Opinion Maximization in Social Networks by Modifying Internal Opinions

Gengyu Wang, Runze Zhang, Zhongzhi Zhang*
College of Computer Science and Artificial Intelligence
Fudan University
{gywang24, rzzhang24}@m.fudan.edu.cn, zhangzz@funda.edu.cn

Abstract

Public opinion governance in social networks is critical for public health campaigns, political elections, and commercial marketing. In this paper, we addresse the problem of maximizing overall opinion in social networks by strategically modifying the internal opinions of a fixed number of nodes. Traditional matrix inversion methods suffer from prohibitively high computational costs, prompting us to propose two efficient sampling-based algorithms. Furthermore, we develop a deterministic asynchronous algorithm that exactly identifies the optimal set of nodes through asynchronous update operations and progressive refinement, ensuring both efficiency and precision. Extensive experiments on real-world datasets demonstrate that our methods outperform baseline approaches. Notably, our asynchronous algorithm delivers exceptional efficiency and accuracy across all scenarios, even in networks with tens of millions of nodes.

1 Introduction

Online social networks have fundamentally transformed the dissemination, evolution, and formation of opinions, serving as a powerful catalyst for accelerating and amplifying modern perspectives [1]. Compared to traditional communication methods, they facilitate faster, broader, and more decentralized information exchange, thereby enhancing the universality, criticality, and complexity of information propagation [2]. Within this intricate interplay between network structure and human behavior, the concept of overall opinion emerges as a key quantitative metric, representing the focal point of public sentiment on contentious issues [3]. This quantified equilibrium of public opinion has been applied in fields such as commercial marketing, political elections, and public health campaigns.

The optimization of overall opinions has garnered significant attention in recent times. Various methods have been explored to optimize collective opinions, including modifying resistance coefficients [4–6], adjusting expressed opinions [7], and altering network structures [8, 9]. Meanwhile, research [10] has highlighted the significant correlation between node topological positions and the evolution of global opinions, revealing that changes in the internal opinions of nodes can have a nonlinear amplification effect on opinion propagation. This provides the possibility of optimizing public opinion at low cost by modifying the internal opinions of key nodes.

In this paper, we address the following optimization problem: given a social network with n nodes and m edges (whether directed or undirected), along with an integer k, how can we strategically identify the k nodes and modify their internal opinions to maximize the overall opinion? Existing exact solution methods require a time complexity of $O(n^3)$, rendering them impractical for large-scale networks. We propose two sampling approaches to approximate the solution, but these approaches face the challenge of balancing between extensive sampling requirements and accuracy. Inspired

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

by the random walk interpretation, we further introduce a asynchronous update algorithm that exactly identifies the optimal set of nodes through asynchronous update operations and progressive refinement. We conducted extensive experiments on various real-world networks to evaluate algorithm performance. The experimental results demonstrate that all three proposed algorithms significantly outperform baseline methods in terms of effectiveness. Moreover, our asynchronous algorithm exhibits both high efficiency and exact precision, while maintaining excellent scalability for networks with tens of millions of nodes.

2 Related Work

We review the related literature from the following two perspectives, including modeling opinion dynamics and optimization problems in opinion dynamics.

Opinion Dynamics Models. Opinion dynamics has been the subject of intense recent research to model social learning processes in various disciplines [11–13]. These models capture the mechanisms and factors influencing opinion formulation, shedding light on understanding the whole process of opinion shaping and diverse phenomena taking place in social media. In the past decades, numerous relevant models have been proposed [14–18]. Among various existing models, the DeGroot model [19] and the Friedkin-Johnson (FJ) model [20] are two popular ones. After their establishment, the DeGroot model and the FJ model have been extended in a variety of ways [11, 21, 8, 22, 4], by incorporating different factors affecting opinion dynamics, such as peer pressure [23], susceptibility to persuasion [4, 8], and opinion leader [7]. Under the formalism of these models, some relevant quantities, properties and explanations have been broadly studied, including the equilibrium expressed opinions [24–26], sufficient condition for the stability [27], the average internal opinion [24], interpretations [28, 25], and so on.

Optimization Problems in Opinion Dynamics. Recently, several optimization problems related to opinion dynamics have been formulated and studied for different objectives. For example, a long line of work has been devoted to maximizing the overall opinion by using different strategies, such as identifying a fixed number of individuals and setting their expressed opinions to 1 [7], changing agent's initial opinions [29, 8, 30], as well as modifying individuals' susceptibility to persuasion [4–6]. [31] studies the problem of allocating seed users to opposing campaigns with a goal to maximize the expected number of users who are co-exposed to both campaigns. In additon, [32] studies the problem of balancing the information exposure. These studies have far-reaching implications in product marketing, public health campaigns, and political candidates. Another major and increasingly important focus of research is optimizing some social phenomena, such as maximizing the diversity [33, 34], minimizing conflict [35, 36], disagreement [37, 38, 9], and polarization [39, 38, 9].

3 Preliminaries

This section is devoted to a brief introduction to some useful notations and tools, in order to facilitate the description of problem formulation and algorithms.

Notations. We denote scalars in $\mathbb{R}$ by normal lowercase letters like a, b, c, sets by normal uppercase letters like A, B, C, vectors by bold lowercase letters like $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, and matrices by bold uppercase letters like $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$. We use $\mathbf{1}$ to denote the vector of appropriate dimensions with all entries being ones, and use $\boldsymbol{e}_i$ to denote the i^{th} standard basis vector of appropriate dimension. Let $\boldsymbol{a}^\top$ and $\boldsymbol{A}^\top$ denote, respectively, transpose of vector $\boldsymbol{a}$ and matrix $\boldsymbol{A}$. We write $A(i, j)$ to denote the entry at row i and column j of $\boldsymbol{A}$ and we use $\boldsymbol{a}(i)$ to denote the i^{th} element of vector $\boldsymbol{a}$. Let $\boldsymbol{a}_{\max}$, $\boldsymbol{a}_{\min}$, $\bar{\boldsymbol{a}}$ and $\boldsymbol{a}_{\text{sum}}$ denote the maximum element, the minimum element, the mean of the elements in vector $\boldsymbol{a}$ and the sum of all elements in vector $\boldsymbol{a}$, respectively.

Graph and Related Matrices. Let $\mathcal{G} = (V, E)$ denote an directed graph with $n = |V|$ nodes and $m = |E|$ edges. The existence of $(v_i, v_j) \in E$ means that there is an edge from v_i to v_j. In what follows, v_i and i are used interchangeably to represent node v_i, when it is clear from the context. For a node $v \in V$, the in-neighbors of v are given by $N_{\text{in}}(v) = \{u|(u, v) \in E\}$, and the out-neighbors of v are given by $N_{\text{out}}(v) = \{u|(v, u) \in E\}$. The connections of graph $\mathcal{G} = (V, E)$ are encoded in its adjacency matrix $\boldsymbol{A} = (a_{i,j})_{n\times n}$, with the element $a_{i,j}$ being 1 if $(v_i, v_j) \in E$ and 0 otherwise.

For a node $i \in V$, its out-degree d_i^+ is defined as $d_i^+ = \sum_{j=1}^n a_{i,j}$, and its in-degree d_v^- is defined as $d_i^- = \sum_{j=1}^n a_{j,i}$. The diagonal degree matrix of $\mathcal{G}$ is defined as $\boldsymbol{D} = \text{diag}(d_1^+, d_2^+, \ldots, d_n^+)$. We define $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ and $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$ as the Laplacian matrix and the transition matrix of graph $\mathcal{G}$.

Opinion Dynamic Model. In this work, we adopt an opinion formation model introduced by the work of DeGroot [19] and Friedkin and Johnsen [40], which has been used in [4, 6, 24]. In this model, each agent i is endowed with an internal opinion s_i in $[0, 1]$, where 0 and 1 are polar opposites of opinions about a certain topic. Each agent also has a parameter that represents the susceptibility to persuasion, which we call the resistance coefficient $\alpha_i \in (0, 1]$. The internal opinion s_i reflects the intrinsic position of the agent i on a certain topic. A higher value on the resistance coefficient α_i means that the agent is less willing to conform to the opinions of the neighbors in the social network. According to the opinion dynamics model, the final opinion of each agent i is a function of the social network, the set of internal opinions, and the resistance coefficients, determined by computing the equilibrium state of a dynamic opinion updating process. The social network is represented as a graph where edges capture influence relationships, with an edge from i to j indicating that agent i is influenced by the expressed opinion of agent j. During the process of opinion evolution, the internal opinion s_i remains constant, while the expressed opinion z_i^t evolves at time $t + 1$ as follows:

$$z_i^{t+1} = \alpha_i s_i + (1 - \alpha_i) \cdot \frac{\sum_{j \in N_{\text{out}}(i)} z_j^t}{d_i^+},$$

which can also be expressed in matrix form as $\boldsymbol{z}^{t+1} = \boldsymbol{R}\boldsymbol{s} + (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P}\boldsymbol{z}^t$. This dynamic converges to a unique equilibrium if $\alpha_i > 0$ for all $i \in V$ [14]. The equilibrium opinion vector $\boldsymbol{z}$ is the solution to a linear system of equations:

$$\boldsymbol{z} = (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^{-1}\boldsymbol{R}\boldsymbol{s}, \tag{1}$$

where $\boldsymbol{R} = \text{Diag}(\alpha)$ is a diagonal matrix called resistance matrix and entry $\boldsymbol{R}(i, i)$ corresponds to α_i. We call $\boldsymbol{z}(i)$ the expressed opinion of agent i. Let $\boldsymbol{M} = (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^{-1}\boldsymbol{R}$, we have $\boldsymbol{z} = \boldsymbol{M}\boldsymbol{s}$. Note that $\boldsymbol{M}$ is a row-stochastic matrix such that $\boldsymbol{M}\mathbf{1} = \mathbf{1}$.

4 Problem Formulation

An important quantity for opinion dynamics is the overall expressed opinion or the average expressed opinion at equilibrium, the optimization problem for which has been addressed under different constraints [7, 41, 4, 6, 8, 29, 30]. In this section, we propose a problem of maximizing overall expressed opinion in a graph, and introduce an exact algorithm optimally solving the problem.

Overall Opinion and Structural Centrality. For the opinion dynamic model in graph $\mathcal{G} = (V, E)$, the overall expressed opinion is defined as the sum $\boldsymbol{z}_{\text{sum}}$ of expressed opinions z_i of every node $i \in V$ at equilibrium. By Eq. (1), $\boldsymbol{z}_{\text{sum}} = \mathbf{1}^\top \boldsymbol{M}\boldsymbol{s}$. Given the internal opinion vector $\boldsymbol{s}$ and the resistance matrix $\boldsymbol{R}$, we use $f(\boldsymbol{R}, \boldsymbol{s})$ to denote the overall expressed opinion. By definition,

$$f(\boldsymbol{R}, \boldsymbol{s}) = \mathbf{1}^\top \boldsymbol{z} = \mathbf{1}^\top \boldsymbol{M}\boldsymbol{s} = \sum_{u \in V} \sum_{v \in V} \boldsymbol{M}(u, v)\boldsymbol{s}(v). \tag{2}$$

Eq. (2) tells us that the overall expressed opinion $f(\boldsymbol{R}, \boldsymbol{s})$ is determined by three factors: the internal opinion and the resistance coefficient of every node, as well as the network structure characterizing interactions between nodes, all of which constitute the social structure of the opinion system. The first two are intrinsic property of each node, while the last one is a structure property of the network, both of which together determine the opinion dynamics system. Concretely, for the equilibrium expressed opinion $\boldsymbol{z}_u = \sum_{v \in V} \boldsymbol{M}(u, v)\boldsymbol{s}(v)$ of node u, $\boldsymbol{M}(u, v)$ indicates the convex combination coefficient or contribution of the internal opinion for node v. And the average value of the v-th column elements of $\boldsymbol{M}$, denoted by $\rho_v \triangleq \sum_{u \in V} \boldsymbol{M}(u, v)$, measures the contribution of the internal opinion of node v to $f(\boldsymbol{R}, \boldsymbol{s})$. We call ρ_v as the structural centrality [42] of node v in the opinion dynamics model, since it catches the long-run structure influence of node v on the overall expressed opinion. Note that matrix $\boldsymbol{M}$ is row stochastic and $0 \leq \boldsymbol{M}(u, v) \leq 1$ for any pair of nodes u and v, $0 \leq \rho_v \leq n$ holds for every node $v \in V$, and $\sum_{v \in V} \rho_v = n$.

Using structural centrality, the overall expressed opinion $f(\boldsymbol{R}, \boldsymbol{s})$ is expressed as $f(\boldsymbol{R}, \boldsymbol{s}) = \sum_{v \in V} \rho_v \boldsymbol{s}(v)$, which shows that the overall expressed opinion $f(\boldsymbol{R}, \boldsymbol{s})$ is a convex combination of the internal opinions of all nodes, with the weight for $\boldsymbol{s}_v$ being the structural centrality ρ_v of node v.

Problem Statement. As shown above, for a given graph $\mathcal{G} = (V, E)$, its node centrality remains fixed. For the FJ opinion dynamics model on $\mathcal{G} = (V, E)$ with internal opinion vector $\boldsymbol{s}$ and resistance matrix $\boldsymbol{R}$, if we choose a set $T \subseteq V$ of k nodes and persuade them to change their internal opinions to 1, the overall equilibrium opinion, denoted by $f_T(\boldsymbol{R}, \boldsymbol{s})$, will increase. It is clear that for $T = \emptyset$, $f_\emptyset(\boldsymbol{R}, \boldsymbol{s}) = f(\boldsymbol{R}, \boldsymbol{s})$. Moreover, for two node sets H and T, if $H \subseteq T \subseteq V$, then $f_T(\boldsymbol{R}, \boldsymbol{s}) \geq f_H(\boldsymbol{R}, \boldsymbol{s})$. Then the problem OPINIONMAX of opinion maximization arises naturally: How to optimally select a set T with a fixed number of k nodes and change their internal opinions to 1, so that their influence on the overall equilibrium opinion is maximized. Let the vector $\boldsymbol{\Delta}$ be the potential influence vector, where $\boldsymbol{\Delta}(i) = \rho_i(1 - \boldsymbol{s}(i))$ defines the potential influence of node i on the growth of the overall equilibrium opinion. Mathematically, it is formally stated as follows.

Problem 1. *(*OPINIONMAX*) Given an unweighted graph $\mathcal{G} = (V, E)$, an internal opinion vector $\boldsymbol{s}$, a resistance matrix $\boldsymbol{R}$, and an integer parameter $k \ll n$, we aim to find the set $T \subset V$ with $|T| = k$ nodes, and change the internal opinions of these chosen k nodes to 1, so that the overall equilibrium opinion is maximized. That is,*

$$T = \arg \max_{U \subset V, |U|=k} f_U(\boldsymbol{R}, \boldsymbol{s}) = \arg \max_{U \subset V, |U|=k} \sum_{i \in U} \boldsymbol{\Delta}(i). \tag{3}$$

Similarly, we can define the problem OPINIONMIN for minimizing the overall equilibrium opinion by optimally selecting a set T of k nodes and changing their internal opinions to 0. The goal of problem OPINIONMIN is to drive the overall equilibrium opinion $f_T(\boldsymbol{R}, \boldsymbol{s})$ towards the polar value 0, while the goal of problem OPINIONMAX is to drive $f_T(\boldsymbol{R}, \boldsymbol{s})$ towards polar value 1. Although the definitions and formulations of problems OPINIONMAX and OPINIONMIN are different, we can prove that they are equivalent to each other. In the sequel, we only consider the OPINIONMAX problem in this paper.

Optimal Solution. The most naive and straightforward method for solving Problem 1 involves directly computing $\boldsymbol{z}$ by inverting the matrix $\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P}$, which has a complexity of $O(n^3)$. Identifying the top k elements using a max-heap has a complexity of $O(n \log k)$. Therefore, the overall time complexity of the algorithm involving matrix inversion is $O(n^3)$. This impractical time complexity makes it infeasible for networks with only tens of thousands of nodes on a single machine. In the following sections, We propose a new interpretation and attempt to propose a new precise algorithm based on this explanation.

5 Sampling Methods

In this section, apart from the algebraic definition, we give two novel interpretations and propose corresponding sampling algorithms to approximately solve Problem 1.

Random Walk-Based Algorithm. Observing that the expression for the overall equilibrium opinion in Eq. (2) can be expanded as $\mathbf{1}^\top \sum_{i=0}^{\infty}((\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^i \boldsymbol{R}\boldsymbol{s}$ via the Neumann series, we introduce the *absorbing random walk*. For a absorbing random walk starting from node s, at each step where the current node is j, the walk either (i) is absorbed by node j with probability α_j, or (ii) moves uniformly at random to a neighboring node with probability $1 - \alpha_j$, where the resistance coefficient α_j of node j is represented as the absorption probability of the random walk at node j.

Lemma 1. *For an unweighted graph $\mathcal{G} = (V, E)$, let $\boldsymbol{p}_i \in \mathbb{R}^{|V|}$ be the absorption probability vector of absorbing random walks starting at node $i \in V$. The structural centrality of node v is $\rho_v = \sum_{i \in V} \boldsymbol{p}_i(v)$.*

Leveraging this connection between structural centrality and termination probabilities, we propose a random walk-based algorithm RWB to efficiently estimate structural centrality for all nodes and compute an approximate solution to Problem 1. In algorithm RWB, we first simulate N runs of the absorbing random walk $\{X_i\}_{i \geq 0}$, where each realization starts from a node uniformly chosen from V. We then estimate the structural centrality ρ_j for each node $j \in V$ by scaling the empirical absorption frequency at j by n.

Lemma 2. *Let $\mathcal{G} = (V, E)$ be an unweighted graph with internal opinion vector $\boldsymbol{s}$, and resistance matrix $\boldsymbol{R}$. For any error tolerance $\epsilon \in (0, 1)$, if algorithm* RWB *simulates $N = O\left(\frac{n}{\epsilon^2} \log n\right)$*

independent random walks, then the estimated structural centrality $\hat{\rho}_i$ of any node $i \in V$ satisfies $\Pr(|\hat{\rho}_i - \rho_i| \geq \epsilon) \leq \frac{1}{n}$.

Theorem 1. *Consider a graph $\mathcal{G} = (V, E)$ with internal opinion vector $\boldsymbol{s}$ and resistance matrix $\boldsymbol{R}$. Let $\alpha_{\min} = \min_{i \in V} R(i,i)$ and $\alpha_{\max} = \max_{i \in V} R(i,i)$. Under the error guarantee of Lemma 2, algorithm* RWB *achieves a time complexity of $O(\frac{\alpha_{\max}(1-\alpha_{\min})}{\epsilon^2 \alpha_{\min}^2} \cdot n \log n)$.*

We now establish that algorithm RWB provides provable approximation guarantees for OPINIONMAX. While Lemma 2 bounds the error of individual $\hat{\rho}_i$ estimates, the following result demonstrates that the *collective quality* of the selected set $\hat{T}$ is near-optimal:

Corollary 1. *Let $\hat{\rho}_i$ be the estimator of ρ_i from algorithm* RWB *with absolute error parameter ϵ, and T^* be the optimal solution to Problem 1. For the set $\hat{T}$ consisting of the k nodes achieving $\arg\max_{|T|=k} \sum_{i \in T} \hat{\rho}_i(1 - s_i)$, we have: $\sum_{i \in \hat{T}} \rho_i(1 - s_i) \geq \sum_{i \in T^*} \rho_i(1 - s_i) - 2k\epsilon$.*

Forest Sampling Algorithm. For matrix $\boldsymbol{Q} + \boldsymbol{L}$, where $\boldsymbol{Q}$ is a diagonal matrix, the elements of its inverse can be interpreted combinatorially in terms of spanning converging forests in a graph [43, 44]. In particular, by setting $\boldsymbol{Q} = \boldsymbol{D}(\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{R}$, we can reformulate Eq. (2) in terms of the fundamental matrix, thereby establishing a direct correspondence between the structural centrality and spanning converging forests.

Lemma 3. *Let $\mathcal{F}$ be the set of all spanning converging forests of graph $\mathcal{G}$, and let $\mathcal{F}^{ij} \subseteq \mathcal{F}$ be the subset where nodes i and j are in the same converging tree rooted at node i. For a forest $F \in \mathcal{F}$, let $r(F)$ be the set of roots of F. Then the structural centrality of node i is $\rho_i = \frac{\sum_{j \in V} \sum_{F \in \mathcal{F}^{ij}} \prod_{u \in r(F)} Q(u,u)}{\sum_{F \in \mathcal{F}} \prod_{u \in r(F)} Q(u,u)}$.*

This theoretical insight connects significantly with an extension of Wilson's algorithm. As shown in [30], Wilson's algorithm, based on loop-erased random walks, effectively simulates the probability distribution of rooted spanning trees when each node $i \in V$ is assigned a probability $\boldsymbol{Q}(i,i)$ of being the root. Building upon this theoretical framework, we propose an algorithm FOREST. The experiment in [30] demonstrates that the algorithm can maintain good effectiveness with a small number of samplings. More details are presented in Appendix B. The following theorem provide its time complexity:

Theorem 2. *When the number of samplings is l, the time complexity of algorithm* FOREST *is $O(\frac{1}{\alpha_{\min}} ln)$.*

6 Fast Exact-Selection Method via Asynchronous Updates

Sampling methods face challenges in identifying optimal size-k node sets with maximal potential influence, as their sample complexity scales as ϵ^{-2}. In this section, we present a deterministic asynchronous algorithm that provides rigorous error guarantees, enabling exact computation of the highest-influence node set without reliance on stochastic samplings. To maintain consistency, we present our derivation using unweighted graphs, noting that the corresponding algorithms can be readily extended to weighted graphs.

6.1 Asynchronous Update-Based Approximation

Let $\boldsymbol{r}^t$ denote the residual vector at time t, initialized as $\boldsymbol{r}^0 = \mathbf{1}$, and updated recursively via $\boldsymbol{r}^{t+1} = \boldsymbol{P}^\top(\boldsymbol{I} - \boldsymbol{R})\boldsymbol{r}^t$. By unrolling this recurrence, we obtain $\boldsymbol{r}^t = (\boldsymbol{P}^\top(\boldsymbol{I} - \boldsymbol{R}))^t\mathbf{1}$, which captures the distribution probability of a t-step absorbing random walk originating from a uniformly chosen node. This probabilistic interpretation motivates our deterministic algorithm employing asynchronous computation. The asynchronous paradigm provides two fundamental advantages: First, it enables local computation where each node's residual evolves independently based on its neighborhoods; second, it supports node-specific termination through local convergence monitoring of individual node states. These properties collectively overcome the synchronization constraints of global methods while eliminating the sampling overhead of probabilistic approaches.

Global Influence Approximation. We propose an efficient asynchronous algorithm to compute the potential influence vector $\boldsymbol{\Delta}$ for the OPINIONMAX problem. The algorithm maintains a residual vector $\boldsymbol{r}_a$ initialized to $\mathbf{1}$, an estimated potential influence vector $\hat{\boldsymbol{\Delta}}$ initialized to $\mathbf{0}$, and uses a

boundary vector $\boldsymbol{h} = \epsilon\mathbf{1}$ (which $\epsilon \in (0,1)$) as the termination criterion. At each iteration, for every node i satisfying $\boldsymbol{r}_a(i) > \boldsymbol{h}(i)$, the algorithm performs three key operations: (i) distributing $(1-\alpha_i)/d_i^+ \cdot \boldsymbol{r}_a(i)$ to each out-neighbor's residual $\boldsymbol{r}_a(j)$ for $j \in N_{\text{out}}(i)$, (ii) accumulating $(1-s_i)\alpha_i \cdot \boldsymbol{r}_a(i)$ to the node's own potential influence estimate $\hat{\boldsymbol{\Delta}}(i)$, and (iii) resetting $\boldsymbol{r}_a(i)$ to 0. These updates are managed asynchronously through a first-in-first-out queue Q, processing nodes when they meet the residual threshold condition. The algorithm terminates when the residual condition $\boldsymbol{r}(v) \leq \boldsymbol{h}(v)$ holds for all nodes $v \in V$. Each push operation maintains strict locality by only accessing direct neighbors, while the asynchronous execution enables efficient computation of the potential influence scores needed for opinion maximization. The pseudo code is provided in Algorithm 1.

Algorithm 1: GLOBALINFAPPROX$(\mathcal{G}, \boldsymbol{R}, \boldsymbol{s}, \epsilon)$

Input : Graph $\mathcal{G} = (V, E)$, resistance matrix $\mathbf{R}$, internal opinion vector $\mathbf{s}$, error parameter ϵ.

Output : Estimated potential influence vector $\hat{\boldsymbol{\Delta}}$ and residue vector $\boldsymbol{r}_a$.

Initialize : $\hat{\boldsymbol{\Delta}} = \mathbf{0};\ \boldsymbol{r}_a = \mathbf{1}$
while $\exists v \in V$ *s.t.* $\boldsymbol{r}_a(v) > \epsilon$ **do**
  $\hat{\boldsymbol{\Delta}}(v) = \hat{\boldsymbol{\Delta}}(v) + (1 - \boldsymbol{s}(v))\alpha_v \boldsymbol{r}_a(v)$
  for *each* $u \in N_{out}(v)$ **do**
    $\boldsymbol{r}_a(u) = \boldsymbol{r}_a(u) + \frac{1-\alpha_v}{d_v^+}\boldsymbol{r}_a(v)$
  $\boldsymbol{r}_a(v) = 0$
return $\hat{\boldsymbol{\Delta}},\ \boldsymbol{r}_a$

The correctness of the algorithm relies on the relationship between the residual vector and the estimation error, which is guaranteed by the following lemma.

Lemma 4. *For any node $i \in V$ during the execution of Algorithm 1, the equality $\boldsymbol{\Delta}(i) - \hat{\boldsymbol{\Delta}}(i) = (1 - \boldsymbol{s}(i)) \cdot \boldsymbol{e}_i^\top \boldsymbol{M}^\top \boldsymbol{r}_a$ holds.*

Using Lemma 4, we can further establish the relative error guarantees for the results returned upon termination of the Algorithm 1, as shown in the following lemma.

Lemma 5. *For any parameter $\epsilon \in (0,1)$, the estimator $\hat{\boldsymbol{\Delta}}$ returned by Algorithm 1 satisfies the following relation: $(1-\epsilon)\boldsymbol{\Delta}(v) \leq \hat{\boldsymbol{\Delta}}(v) \leq \boldsymbol{\Delta}(v), \forall v \in V$.*

Targeted Node Refinement. Algorithm 1 provides solutions with rigorous relative error guarantees. As established in Lemma 8, these precise error bounds allow us to determine whether a given node must necessarily belong to the size-k set with maximal potential influence. Furthermore, when the error tolerance ϵ becomes sufficiently small, we can completely identify the exact size-k node set that maximizes opinion influence. However, since each error threshold setting generates a distinct candidate set of boundary nodes, repeatedly computing global error bounds through Algorithm 1 would incur unnecessary computational overhead. This motivates our key optimization: Can we focus computations exclusively on a reduced candidate set identified by Algorithm 1 to improve efficiency?

Algorithm 2: TARGETEDNODEREFINE$(\mathcal{G}, \boldsymbol{r}_a, \boldsymbol{r}^0, \epsilon)$

Input : Graph $\mathcal{G} = (V, E)$, forward residual vector $\boldsymbol{r}_a$, initial residual vector $\boldsymbol{r}^0$, error parameter ϵ.

Output : Estimator $\tilde{\Delta}$ and residue vector $\boldsymbol{r}_s$.

Initialize : $\tilde{\Delta} = 0;\ \boldsymbol{r}_s = \boldsymbol{r}^0$
while $\exists v \in V$ *s.t.* $\boldsymbol{r}_s(v) > \epsilon\alpha_v$ **do**
  $\tilde{\Delta} = \tilde{\Delta} + \boldsymbol{r}_a(v) \cdot \boldsymbol{r}_s(v)$
  for *each* $u \in N_{in}(v)$ **do**
    $\boldsymbol{r}_s(u) = \boldsymbol{r}_s(u) + \frac{1-\alpha_u}{d_u^+}\boldsymbol{r}_s(v)$
  $\boldsymbol{r}_s(v) = 0$
return $\tilde{\Delta},\ \boldsymbol{r}_s$

The theoretical foundation comes from Lemma 4, which provides the exact decomposition for all nodes in the network. This decomposition directly motivates Algorithm 2, designed to approximate the influence propagation term $(1 - \boldsymbol{s}(i)) \cdot \boldsymbol{e}_i^\top \boldsymbol{M}^\top \boldsymbol{r}_a$ for any given node i. The algorithm initializes with estimate $\tilde{\Delta} = 0$ and residual vector $\boldsymbol{r}_s = \boldsymbol{r}^0$, using $\boldsymbol{h} = \epsilon \boldsymbol{R}\mathbf{1}$ as the boundary vector. For each node v satisfying $\boldsymbol{r}_s(v) > \boldsymbol{h}(v)$, the algorithm updates $\tilde{\Delta}$ by adding $\boldsymbol{r}_a(v) \cdot \boldsymbol{r}_s(v)$ and propagating residuals to in-neighbors before resetting $\boldsymbol{r}_s(v) = 0$. The process repeats until $\boldsymbol{r}_s(v) \leq \boldsymbol{h}(v)$ for all nodes.

While Algorithm 2's asynchronous operations do not maintain the random walk interpretation, its correctness is established through the following analysis.

Lemma 6. *When Algorithm 2 is initialized with $\boldsymbol{r}^0 = \alpha_v(1 - \boldsymbol{s}(v))\boldsymbol{e}_v$, the equality $\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) = \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}_s$ holds throughout execution.*

Building upon Lemma 6, we can derive the following absolute error bound when specific initialization conditions are met.

Lemma 7. *For Algorithm 2 with initial residual $\boldsymbol{r}^0 = \alpha_v(1 - \boldsymbol{s}(v))\boldsymbol{e}_v$, given any parameter $\epsilon \in (0, 1)$, the estimator $\tilde{\Delta}$ satisfies the absolute error bound: $0 < \boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) \leq \epsilon \cdot (\boldsymbol{r}_a)_{\text{sum}}$.*

6.2 Fast Exact-Selection Algorithm

Let $T \subseteq V$ be the set of nodes guaranteed to be contained in the optimal size-k node set with maximal value of sequence $\boldsymbol{a}$, and let $C \subseteq V \setminus T$ denote the candidate nodes that may belong to this optimal set when considering estimation errors. Given an estimator $\hat{\boldsymbol{a}}$ with uniform error bounds (either absolute or relative), we can formally characterize these sets through the following lemma.

Lemma 8. *Let $\hat{\boldsymbol{a}}$ approximate $\boldsymbol{a}$ with uniform error $0 < \boldsymbol{a}(i) - \hat{\boldsymbol{a}}(i) \leq \epsilon_a$ or $0 < \boldsymbol{a}(i) - \hat{\boldsymbol{a}}(i) \leq \epsilon_b \boldsymbol{a}(i)$ for all $i \in V$, and let $\hat{\boldsymbol{a}}_{<i>}$ denote the i-th largest value in $\hat{\boldsymbol{a}}$. Then,*

- *Element $i \in T$ if either $\hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{<k+1>} + \epsilon_a$ or $\hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{<k+1>}/(1 - \epsilon_b)$;*
- *Element $i \in C$ if $i \notin T$ and either $\hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{<k>} - \epsilon_a$ or $\hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{<k>}(1 - \epsilon_b)$.*

Let k_{gap} denote the difference between the k-th and $(k + 1)$-th largest values in the true potential influence vector $\boldsymbol{\Delta}$. When the absolute error $\epsilon < k_{\text{gap}}$, we can guarantee exact identification of the optimal size-k node set. However, since k_{gap} is typically unknown a priori, we employ an iterative refinement approach that progressively tightens the error bound ϵ across successive iterations. This process leverages residual vectors from previous iterations as initial conditions, thereby reducing recomputation overhead through warm-start optimization. The algorithm operates through two computational phases. First, global relative error bounds are applied to identify a small candidate set C. Then, Algorithm 2 computes absolute error guarantees specifically for nodes in C while

Algorithm 3: MAXINFLUENCESELECTOR$(\mathcal{G}, \boldsymbol{R}, \boldsymbol{s}, \epsilon, k)$

Input : Graph $\mathcal{G} = (V, E)$, opinions $\boldsymbol{s}$, parameter k.
Output : Optimal node set T.

Initialize : $T = \emptyset$; $C = V$
$\hat{\boldsymbol{\Delta}}, \boldsymbol{r}_a =$ GLOBALINFAPPROX$(\mathcal{G}, \boldsymbol{R}, \boldsymbol{s}, \epsilon)$
Update T and C via Lemma 8 with $\{\hat{\boldsymbol{\Delta}}(v)\}_{v \in V}$ and error ϵ
Initialize $\boldsymbol{r}_v = \alpha_v(1 - s_v) \cdot \boldsymbol{e}_v$, $\forall v \in C$
for $\epsilon' = 1, \frac{1}{2}, \ldots, \frac{1}{2n}, \ldots$ **do**
  for $v \in C$ **do**
    $\tilde{\Delta}, \boldsymbol{r}_v =$ TARGETEDNODEREFINE$(\mathcal{G}, \boldsymbol{r}_a, \boldsymbol{r}_v, \epsilon'/(\boldsymbol{r}_a)_{\text{sum}})$
    $\hat{\boldsymbol{\Delta}}(v) = \hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}$
  Update T and C via Lemma 8 with $\{\hat{\boldsymbol{\Delta}}(v)\}_{v \in C}$ and error ϵ'
  if $|T| = k$ **then**
    return T

progressively shrinking the candidate set size. The procedure terminates when the error bound satisfies $\epsilon < k_{\text{gap}}$ and the candidate set becomes empty, at which point we obtain the exact optimal size-k set with maximal potential influence. The pseudo code is provided in Algorithm 3.

Theoretical guarantees of this approach are established in the following theorem, which provides an upper bound on the time complexity.

Theorem 3. *For sufficiently small ϵ, the upper bound on the time complexity of Algorithm 3 is* $O(\frac{d^+_{\max} n}{\alpha_{\min}} \log \frac{1}{\epsilon} + \frac{m}{k_{gap} \cdot \alpha_{\min}})$.

7 Experiments

In this section, we experimentally evaluate our proposed three algorithms. Additional experimental results and analyses are presented in Appendix C.

Machine. Our extensive experiments were conducted on a Linux server equipped with 28-core 2.0GHz Intel(R) Xeon(R) Gold 6330 CPU and 1TB of main memory. All the algorithms we proposed are implemented in *Julia v1.10.7* using single-threaded execution.

Table 1: Datasets

	DBLP	Google	YoutubeSnap	Pokec	Flixster	LiveJournal	Twitter	SinaWeibo
Nodes	317,080	875,713	1,134,890	1,632,803	2,523,386	4,847,571	41,652,230	58,655,849
Edges	1,049,866	5,105,039	2,987,624	30,622,564	7,918,801	68,993,773	1,468,365,182	261,321,071
$d^+_{\max}$	343	456	28,754	8,763	1,474	20,296	770,155	278,491
Type	undirected	directed	undirected	directed	undirected	directed	directed	undirected

Datasets and Metrics. We use 8 benchmark datasets that are obtained from the Koblenz Network Collection [45], SNAP [46] and Network Repository [47]. Table 1 summarizes the key characteristics of the networks used in our experiments, including network name, number of nodes, number of edges, maximum out-degree, and network type. The networks are listed in ascending order based on node count. A more detailed description of the datasets is provided in Appendix D. On each dataset, we employ the algorithm GLOBALINFAPPROX with a relative error parameter 10^{-12} to compute the ground-truth structural centrality scores. This ensures that each ground-truth value has at most 10^{-12} relative error. We evaluate the maximum influence node selection problem for varying set sizes $k \in \{1, 2, 4, \ldots, 1024\}$. We evaluate the accuracy of each method using three metrics: overall opinion, along with two standard ranking measures, *precision* and *Normalized Discounted Cumulative Gain (NDCG)* [48]. We initialize internal opinions with uniform distribution as their configuration does not significantly affect experimental results.

Methods. We present numerical results to evaluate the performance of our proposed algorithms, RWB, FOREST and MAXINFLUENCESET against existing baselines. For consistency, we abbreviate MAXINFLUENCESET as MIS in the following text. In all experiments, we set the absolute error parameters of RWB to 10^{-2}. For algorithm FOREST, we set the number of samplings $l = 4000$. We set the initial error parameter of algorithm MIS to 10^{-3}. To further demonstrate the effectiveness of our approach, we compare against five widely-used benchmark algorithms: TOPRANDOM, TOPDEGREE, TOPCLOSENESS, TOPBETWEENNESS, and TOPPAGERANK [49] across all networks.

Resistance Coefficient Distributions. In our experiments, resistance coefficient are generated using three distinct distributions: uniform, normal, and exponential. They are abbreviated as Unif., Norm., and Exp. in the figures. We set the minimum value of the resistance coefficient $\alpha_{\min} = 0.01$ because a zero resistance coefficient would lead to non-convergent results. For the uniform distribution, each node i is assigned an opinion s_i uniformly sampled from the interval $[\alpha_{\min}, 1]$. For the normal distribution, each node i is assigned a sample z_i drawn from the standard normal distribution $z_i \sim \mathcal{N}(0, 1)$. These values are then normalized to the interval $[\alpha_{\min}, 1]$. For the exponential distribution, we generate n positive numbers x using the probability density function $f(x) = \mathrm{e}^{x_{\min}} \mathrm{e}^{-x}$, where $x_{\min} > 0$. These values are similarly normalized to the range $[\alpha_{\min}, 1]$ to represent resistance coefficients.

Efficiency. Table 2 compares the execution time of three algorithms: RWB, FOREST, and our proposed MIS (with $k = 64$), under various resistance coefficient distributions. The direct matrix inversion based EXACT algorithm was excluded from the comparison as it failed to complete

Table 2: Running time with $k = 64$.

Name	Time(s)								
	RWB			FOREST			MIS		
	Unif.	Norm.	Exp.	Unif.	Norm.	Exp.	Unif.	Norm.	Exp.
DBLP	120.61	101.72	703.31	66.02	58.65	104.41	0.28	0.51	1.81
Google	344.02	326.83	1368.66	162.87	154.62	223.67	0.39	0.40	2.91
YoutubeSnap	896.66	900.97	10495.42	196.53	208.92	312.03	1.03	0.87	6.85
Pokec	1037.87	1028.07	6562.19	467.81	493.82	784.69	3.72	3.21	24.68
Flixster	2342.12	2142.68	-	326.52	321.75	376.91	2.59	2.14	13.56
LiveJournal	3275.20	2580.90	-	1454.37	1472.17	2147.34	8.61	13.17	56.49
Twitter	-	-	-	12115.80	13230.32	13852.17	279.73	270.61	1841.55
SinaWeibo	-	-	-	10538.27	10749.41	11743.13	156.83	171.75	1712.25

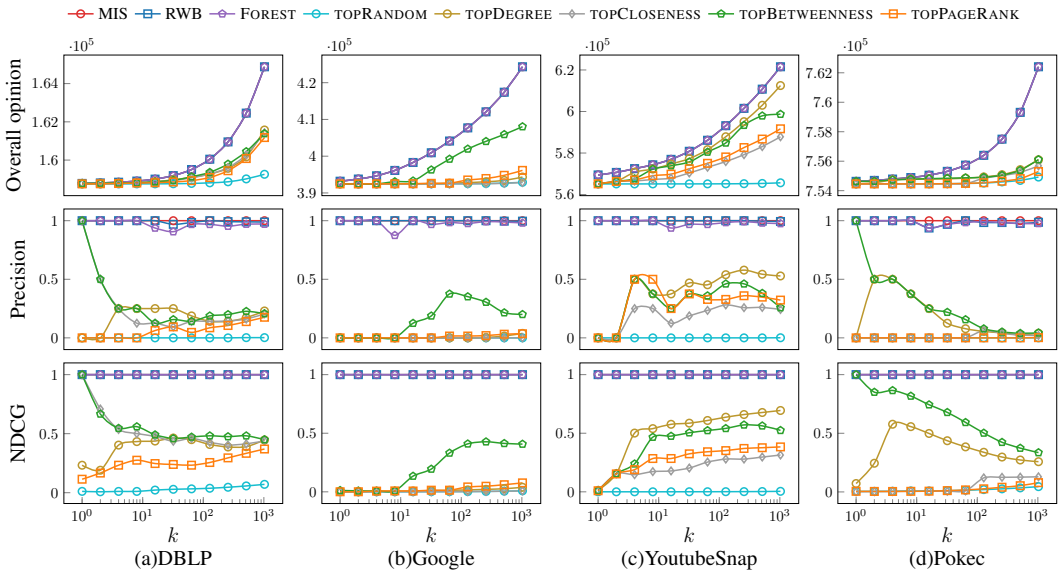


Figure 1: Performance of algorithms in accuracy across four smaller graphs.

even on the smallest DBLP network within the 6-hour time limit. Similarly, RWB executions exceeding 6 hours on large-scale networks with strict resistance distributions were terminated due to impractical runtime requirements. In contrast, both FOREST and MIS demonstrated robust scalability, successfully processing networks with tens of millions of nodes across all distribution scenarios. The data reveals that FOREST consistently achieves faster execution than RWB, while MIS outperforms both algorithms by a substantial margin in all test conditions. Notably, the execution time of RWB and MIS shows considerable sensitivity to resistance coefficient distributions, whereas FOREST maintains relatively stable performance in most cases despite distribution variations. As shown in Table 2, the excellent efficiency of our MIS algorithm can be easily applied in large-scale networks containing tens of millions of nodes, whether directed or undirected. This performance advantage makes MIS a feasible solution for large-scale network analysis tasks in the real world.

Accuracy. Figures 1 and 2 presents a comprehensive evaluation of opinion optimization performance, precision, and NDCG scores for all methods under uniformly distributed resistance coefficients. Due to excessively long running times, we omitted the performance of the RWB algorithm on the Twitter and SinaWeibo networks in Figure 2. Our experimental evaluation reveals that the proposed algorithms (RWB, FOREST, and MIS) consistently outperform all baseline methods across all evaluation metrics. This superior performance demonstrates the effectiveness of our approach in opinion optimization tasks. The result shows that while RWB and FOREST exhibit show measurable variations in precision under certain conditions, these fluctuations do not substantially affect their overall opinion optimization performance. The robustness of these algorithms across

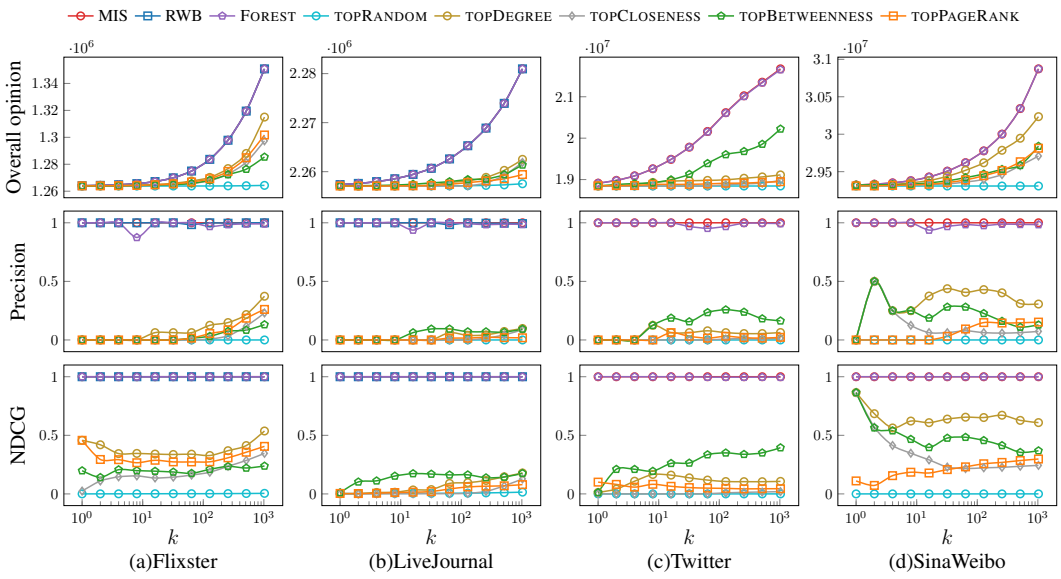


Figure 2: Performance of algorithms in accuracy across four larger graphs.

different network structures further confirms their practical utility. Notably, the MIS algorithm achieves perfect computational precision (exactly 1) while maintaining NDCG scores approaching 1, indicating both optimal opinion optimization results and accurate sequence ordering. When combined with its exceptional efficiency demonstrated in Table 2, these results further underscore the superiority of the MIS algorithm in both accuracy and computational performance.

8 Limitations

Despite its advantages, our MIS algorithm has certain limitations. As formally established in Theorem 3, the algorithm remains sensitive to resistance coefficients—particularly exhibiting prolonged runtime when these values are small. Furthermore, in exact optimization scenarios where boundary nodes contribute equally, the algorithm may encounter termination issues, though such cases are practically negligible. We note that this edge case can be effectively addressed by relaxing constraint conditions.

9 Conclusion

This paper proposes novel solutions for optimizing overall opinions in social networks by modifying the internal opinions of key nodes. As traditional matrix inversion methods face computational limitations in large-scale networks, we introduce two sampling-based algorithms. Building upon a random walk interpretation, we further develop a exact asynchronous update algorithm. This deterministic asynchronous approach provides guaranteed error bounds, leveraging asynchronous update operations and progressive refinement to efficiently and exactly identify nodes with the greatest potential influence. Extensive experiments demonstrate that compared to baseline methods and our sampling approaches, this method achieves superior efficiency and accuracy while effectively scaling to networks with tens of millions of nodes. Future research directions include extensions to dynamic network configurations and multi-opinion optimization scenarios.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62372112).

References

[1] H. Ledford, "How facebook, twitter and other data troves are revolutionizing social science," *Nature*, vol. 582, no. 7812, pp. 328–331, 2020.

[2] D. Notarmuzi, C. Castellano, A. Flammini, D. Mazzilli, and F. Radicchi, "Universality, criticality and complexity of information propagation in social media," *Nature Communications*, vol. 13, no. 1, p. 1308, 2022.

[3] G. Pasi and R. R. Yager, "Modeling the concept of majority opinion in group decision making," *Information Sciences*, vol. 176, no. 4, pp. 390–414, 2006.

[4] R. Abebe, J. Kleinberg, D. Parkes, and C. E. Tsourakakis, "Opinion dynamics with varying susceptibility to persuasion," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1089–1098.

[5] T. H. Chan, Z. Liang, and M. Sozio, "Revisiting opinion dynamics with varying susceptibility to persuasion via non-convex local search," in *Proceedings of the World Wide Web Conference*, 2019, pp. 173–183.

[6] R. Abebe, T.-H. H. Chan, J. Kleinberg, Z. Liang, D. Parkes, M. Sozio, and C. E. Tsourakakis, "Opinion dynamics optimization by varying susceptibility to persuasion via non-convex local search," *ACM Transactions on Knowledge Discovery from Data*, vol. 16, no. 2, pp. 1–34, 2021.

[7] A. Gionis, E. Terzi, and P. Tsaparas, "Opinion maximization in social networks," in *Proceedings of the 2013 SIAM International Conference on Data Mining*, 2013, pp. 387–395.

[8] Z. Zhang, W. Xu, Z. Zhang, and G. Chen, "Opinion dynamics incorporating higher-order interactions," in *2020 IEEE International Conference on Data Mining*, 2020, pp. 1430–1435.

[9] L. Zhu, Q. Bao, and Z. Zhang, "Minimizing polarization and disagreement in social networks via link recommendation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2072–2084, 2021.

[10] J. Klausen, C. E. Marks, and T. Zaman, "Finding extremists in online social networks," *Operations Research*, vol. 66, no. 4, pp. 957–976, 2018.

[11] P. Jia, A. MirTabatabaei, N. E. Friedkin, and F. Bullo, "Opinion dynamics and the evolution of social power in influence networks," *SIAM Review*, vol. 57, no. 3, pp. 367–397, 2015.

[12] Y. Dong, M. Zhan, G. Kou, Z. Ding, and H. Liang, "A survey on the fusion process in opinion dynamics," *Information Fusion*, vol. 43, pp. 57–65, 2018.

[13] B. D. Anderson and M. Ye, "Recent advances in the modelling and analysis of opinion dynamics on influence networks," *International Journal of Automation and Computing*, vol. 16, no. 2, pp. 129–149, 2019.

[14] A. V. Proskurnikov and R. Tempo, "A tutorial on modeling and analysis of dynamic social networks. part I," *Annual Reviews in Control*, vol. 43, pp. 65–79, 2017.

[15] H. Hassani, R. Razavi-Far, M. Saif, F. Chiclana, O. Krejcar, and E. Herrera-Viedma, "Classical dynamic consensus and opinion dynamics models: A survey of recent trends and methodologies," *Information Fusion*, vol. 88, pp. 22–40, 2022.

[16] H. Noorazar, K. R. Vixie, A. Talebanpour, and Y. Hu, "From classical to modern opinion dynamics," *International Journal of Modern Physics C*, vol. 31, no. 07, p. 2050101, 2020.

[17] M. Okawa and T. Iwata, "Predicting opinion dynamics via sociologically-informed neural networks," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 1306–1316.

[18] A. De, S. Bhattacharya, and N. Ganguly, "Shaping opinion dynamics in social networks," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, 2018, pp. 1336–1344.

[19] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[20] N. E. Friedkin and E. C. Johnsen, "Social influence and opinions," *Journal of Mathematical Sociology*, vol. 15, no. 3-4, pp. 193–206, 1990.

[21] G. He, W. Zhang, J. Liu, and H. Ruan, "Opinion dynamics with the increasing peer pressure and prejudice on the signed graph," *Nonlinear Dynamics*, vol. 99, pp. 3421–3433, 2020.

[22] U. Chitra and C. Musco, "Analyzing the impact of filter bubbles on social network polarization," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020, pp. 115–123.

[23] J. Semonsen, C. Griffin, A. Squicciarini, and S. Rajtmajer, "Opinion dynamics in the presence of increasing agreement pressure," *IEEE Transactions on Cybernetics*, vol. 49, no. 4, pp. 1270–1278, 2018.

[24] A. Das, S. Gollapudi, R. Panigrahy, and M. Salek, "Debiasing social wisdom," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 500–508.

[25] D. Bindel, J. Kleinberg, and S. Oren, "How bad is forming your own opinion?" *Games and Economic Behavior*, vol. 92, pp. 248–265, 2015.

[26] G. Wang, R. Zhang, and Z. Zhang, "Efficient algorithms for relevant quantities of friedkin-johnsen opinion dynamics model," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 2915–2926.

[27] C. Ravazzi, P. Frasca, R. Tempo, and H. Ishii, "Ergodic randomized algorithms and dynamics over networks," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 1, pp. 78–87, 2014.

[28] J. Ghaderi and R. Srikant, "Opinion dynamics in social networks with stubborn agents: Equilibrium and convergence rate," *Automatica*, vol. 50, no. 12, pp. 3209–3215, 2014.

[29] Y. Yi, T. Castiglia, and S. Patterson, "Shifting opinions in a social network through leader selection," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 3, pp. 1116–1127, 2021.

[30] H. Sun and Z. Zhang, "Opinion optimization in directed social networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, 2023, pp. 4623–4632.

[31] S. Tu, C. Aslay, and A. Gionis, "Co-exposure maximization in online social networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3232–3243, 2020.

[32] K. Garimella, A. Gionis, N. Parotsidis, and N. Tatti, "Balancing information exposure in social networks," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[33] E. Mackin and S. Patterson, "Maximizing diversity of opinion in social networks," in *2019 American Control Conference*, 2019, pp. 2728–2734.

[34] A. Matakos, S. Tu, and A. Gionis, "Tell me something my friends do not know: Diversity maximization in social networks," *Knowledge and Information Systems*, vol. 62, pp. 3697–3726, 2020.

[35] X. Chen, J. Lijffijt, and T. De Bie, "Quantifying and minimizing risk of conflict in social networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1197–1205.

[36] Y. Wang and J. Kleinberg, "On the relationship between relevance and conflict in online social link recommendations," *Advances in Neural Information Processing Systems*, vol. 36, pp. 36 708–36 725, 2023.

[37] J. Gaitonde, J. Kleinberg, and E. Tardos, "Adversarial perturbations of opinion dynamics in networks," in *Proceedings of the 21st ACM Conference on Economics and Computation*, 2020, pp. 471–472.

[38] C. Musco, C. Musco, and C. E. Tsourakakis, "Minimizing polarization and disagreement in social networks," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 369–378.

[39] A. Matakos, E. Terzi, and P. Tsaparas, "Measuring and moderating opinion polarization in social networks," *Data Mining and Knowledge Discovery*, vol. 31, pp. 1480–1505, 2017.

[40] N. E. Friedkin and E. C. Johnsen, "Social influence networks and opinion change," *Advances in group processes*, vol. 16, no. 1, pp. 1–29, 1999.

[41] A. Ahmadinejad, S. Dehghani, M. Hajiaghayi, H. Mahini, S. Seddighin, and S. Yazdanbod, "Forming external behaviors by leveraging internal opinions," in *2015 IEEE Conference on Computer Communications*, 2015, pp. 1849–1857.

[42] N. E. Friedkin, "A formal theory of reflected appraisals in the evolution of power," *Administrative Science Quarterly*, vol. 56, no. 4, pp. 501–529, 2011.

[43] W. Xu, L. Zhu, J. Guan, Z. Zhang, and Z. Zhang, "Effects of stubbornness on opinion dynamics," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 2321–2330.

[44] P. Chebotarev and E. Shamis, "Matrix-forest theorems," *arXiv preprint math/0602575*, 2006.

[45] J. Kunegis, "Konect: the koblenz network collection," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 1343–1350.

[46] J. Leskovec and R. Sosič, "Snap: A general-purpose network analysis and graph-mining library," *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 1, pp. 1–20, 2016.

[47] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[48] K. Järvelin and J. Kekäläinen, "Ir evaluation methods for retrieving highly relevant documents," in *ACM SIGIR Forum*, vol. 51, no. 2, 2017, pp. 243–250.

[49] L. Vassio, F. Fagnani, P. Frasca, and A. Ozdaglar, "Message passing optimization of harmonic influence centrality," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 109–120, 2014.

[50] R. P. Agaev and P. Y. Chebotarev, "Spanning forests of a digraph and their applications," *Automation and Remote Control*, vol. 62, pp. 443–466, 2001.

[51] P. Chebotarev and R. Agaev, "Forest matrices around the laplacian matrix," *Linear Algebra and its Applications*, vol. 356, no. 1-3, pp. 253–274, 2002.

[52] D. B. Wilson, "Generating random spanning trees more quickly than the cover time," in *Proceedings of the twenty-eighth annual ACM Symposium on Theory of Computing*, 1996, pp. 296–303.

[53] L. Avena and A. Gaudillière, "Two applications of random spanning forests," *Journal of Theoretical Probability*, vol. 31, no. 4, pp. 1975–2004, 2018.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions, including proposing efficient sampling-based algorithms (RWB, FOREST) and a deterministic asynchronous algorithm (MIS) for opinion maximization in large-scale networks. The claims align with theoretical analysis and experiments on real-world datasets (Section 4-7).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper acknowledges limitations in scalability of traditional methods and mentions future extensions to dynamic networks or multi-opinion scenarios (Section 8).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Theoretical assumptions (e.g., resistance coefficients $\alpha_i \in (0, 1]$) are clearly stated. Proofs for lemmas and theorems are provided in Appendix A with logical derivations.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper details the algorithm implementations (e.g., RWB, FOREST, MIS), parameter settings, dataset sources, and hardware configurations in the Experiments section (Section 7). This information is sufficient for independent researchers to reproduce the main experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The experimental code has been packaged into ZIP files and submitted as supplementary materials with the paper. Sufficient instructions are provided in the supplemental material to faithfully reproduce the main experimental results, including details on dataset preparation, parameter settings, and execution steps.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental setup includes dataset statistics, machine specifications, parameter choices, and baseline methods. See Table 1 and Section 7.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The deterministic nature of the MIS algorithm ensures zero variance in its outputs. Furthermore, we provide rigorous mathematical definitions for all test distributions (uniform/normal/exponential) and their parameters, with comprehensive experimental validation demonstrating consistent performance.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides complete computational resource specifications: (1) Dataset scales (nodes/edges) are detailed in Table 1; (2) Execution times for all algorithms are reported in Table 1 and Figure 2; (3) All experiments were conducted in single-threaded mode on a 28-core 2.0GHz Xeon server with 1TB RAM, as specified in Section 7.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work is purely algorithmic and does not involve any research with human participants, crowdsourcing, or data collection from human subjects. All experiments are conducted on publicly available benchmark datasets.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses both positive applications (e.g., public health campaigns, political elections) and potential negative societal impacts (e.g., opinion manipulation risks) in Sections 1 and 8, fulfilling the requirement for broader impact analysis.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This study belongs to basic algorithm research and does not involve the publication of high-risk models or data (such as large language models or sensitive datasets), therefore, it is not subject to safeguard measures requirements.

    Guidelines:

    Justification: The paper introduces new algorithmic implementations (e.g., RWB, FOREST, and MIS) as part of its contributions. These assets are well-documented in the Methods and Experiments sections (Sections 5–7), with detailed descriptions of the algorithms, parameter settings, and computational workflows. The code is released under the MIT License, and documentation is provided in the supplemental materials to ensure reproducibility.

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The public datasets (Koblenz, SNAP, Network Repository) have been sourced, but specific licenses have not been cited. The data source references (such as [42], [43], [44]) have been provided (Section 7).

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces new algorithmic implementations (e.g., RWB, FOREST, and MIS) as part of its contributions. These assets are well-documented in the Methods and Experiments sections (Sections 5–7), with detailed descriptions of the algorithms, parameter settings, and computational workflows. The code is released under the MIT License, and documentation is provided in the supplemental materials to ensure reproducibility.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

Justification: The work does not involve human subjects or crowdsourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects are involved.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs are not used in the core methodology. Writing/editing uses standard tools without LLM assistance.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

A Omitted Proofs

A.1 Proof of Lemma 1

According to Neumann series, we have $\boldsymbol{M} = (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^{-1}\boldsymbol{R} = \sum_{t=0}^{\infty}((\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^t\,\boldsymbol{R}$. Let $\boldsymbol{p}_i = \boldsymbol{e}_i^\top \sum_{t=0}^{\infty}((\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^t\,\boldsymbol{R}$ represent the absorption probability vector of absorbing random walks starting from node i. Hence, we have:

$$\rho_v = \sum_{i\in V}\boldsymbol{M}(i,v) = \sum_{i\in V}\boldsymbol{e}_i^\top\sum_{t=0}^{\infty}((\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^t\,\boldsymbol{R}\boldsymbol{e}_v = \sum_{i\in V}\boldsymbol{p}_i(v).$$

A.2 Proof of Lemma 2

Algorithm RWB simulates N independent runs of the absorbing random walk $\{X_i\}_{i=1}^N$, where:

$$X_i = \begin{cases} n, & \text{if the } i\text{-th walk is absorbed by } v,\\ 0, & \text{otherwise.}\end{cases}$$

The estimator $\hat{\rho}_v = \frac{1}{N}\sum_{i=1}^N X_i$ has the following properties:

$$\begin{aligned}\mathbb{E}[\hat{\rho}_v] &= \frac{1}{N}\sum_{i=1}^N\mathbb{E}[X_i] = n\cdot\frac{1}{n}\sum_{u\in V}\boldsymbol{p}_u(v) = \rho_v.\\ \mathrm{Var}[\hat{\rho}_v] &= \frac{1}{N^2}\sum_{i=1}^N\mathrm{Var}[X_i] = \frac{1}{N^2}\sum_{i=1}^N\left(\mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2\right)\\ &= \frac{1}{N^2}\sum_{i=1}^N\left(n^2\cdot\frac{\rho_v}{n} - \rho_v^2\right) = \frac{\rho_v(n-\rho_v)}{N} \le \frac{n^2}{4N}.\end{aligned}$$

We now apply the following Chernoff bound for bounded variables.

Lemma 9 (Chernoff Bound)**.** *Let $X_i (1 \le i \le N)$ be independent random variables satisfying $X_i \le E[X_i] + M$ for $1 \le i \le N$. Let $X = \frac{1}{N}\sum_{i=1}^N X_i$. Assume that $E[X]$ and $Var[X]$ are respectively the expectation and variance of X. Then we have*

$$\Pr(|X - E[X]| \ge \lambda) \le 2\exp(-\frac{\lambda^2 N}{2Var[X] + 2M\lambda/3}).$$

Applying this lemma with $M = n$ and $\lambda = \epsilon$, we obtain

$$\Pr\left(|\hat{\rho}_v - \rho_v| \ge \epsilon\right) \le 2\exp\left(-\frac{\epsilon^2 N}{2\cdot\frac{n^2}{4N} + 2n\epsilon/3}\right) = 2\exp\left(-\frac{\epsilon^2 N}{\frac{n^2}{2N} + 2n\epsilon/3}\right).$$

When $N = O\left(\frac{n}{\epsilon^2}\log n\right)$, we have $\Pr\left(|\hat{\rho}_v - \rho_v| \ge \epsilon\right) \le \frac{1}{n}$.

A.3 Proof of Theorem 1

We first prove the expected length of a random walk. Since $\sum_{l=1}^{\infty} l \cdot x^{l-1} = \frac{1}{(1-x)^2}$ for $|x| < 1$, we have

$$\begin{aligned}
\mathbb{E}[\text{Walk length starting from node } u] &= \sum_{l=0}^{\infty} l \cdot \Pr(\text{Walk length starting from node } u \text{ is } l) \\
&= \sum_{l=0}^{\infty} l \cdot \boldsymbol{e}_u^\top \left((\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P}\right)^l \boldsymbol{R}\boldsymbol{1} \leq \sum_{l=0}^{\infty} l(1-\alpha_{\min})^l \alpha_{\max} \\
&= \alpha_{\max}(1-\alpha_{\min}) \sum_{l=1}^{\infty} l(1-\alpha_{\min})^{l-1} \\
&= \frac{\alpha_{\max}(1-\alpha_{\min})}{\alpha_{\min}^2}.
\end{aligned}$$

According to Lemma 1, we have $N = \frac{n}{\epsilon^2}\log n$, hence the upper bound of time complexity is $O(\frac{\alpha_{\max}(1-\alpha_{\min})}{\epsilon^2 \alpha_{\min}^2} \cdot n \log n)$.

A.4 Proof of Lemma 3

We begin by establishing the representation $\boldsymbol{M} = (\boldsymbol{Q} + \boldsymbol{L})^{-1}\boldsymbol{Q}$. Starting from the definition of $\boldsymbol{M}$, we derive

$$\begin{aligned}
\boldsymbol{M} &= (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{P})^{-1}\boldsymbol{R} = (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{D}^{-1}\boldsymbol{A})^{-1}\boldsymbol{R} = (\boldsymbol{D}(\boldsymbol{I} - \boldsymbol{R})^{-1} - \boldsymbol{A})^{-1}\boldsymbol{D}(\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{R} \\
&= (\boldsymbol{D}\sum_{i=0}^{\infty}\boldsymbol{R}^i - \boldsymbol{A})^{-1}\boldsymbol{Q} = (\boldsymbol{D}\sum_{i=0}^{\infty}\boldsymbol{R}^i \cdot \boldsymbol{R} + \boldsymbol{D} - \boldsymbol{A})^{-1}\boldsymbol{Q} \\
&= (\boldsymbol{D}(\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{R} + \boldsymbol{L})^{-1}\boldsymbol{Q} = (\boldsymbol{Q} + \boldsymbol{L})^{-1}\boldsymbol{Q}.
\end{aligned}$$

To prove Lemma 3, we invoke a result from [44] concerning spanning converging forests:

Lemma 10 ([44])**.** *Let $\boldsymbol{L}_{-S}$ be the matrix obtained from $\boldsymbol{L}$ by deleting the rows and columns corresponding to the nodes in $S \subseteq V$, and let $\mathcal{F}_S$ be the set of spanning converging forests of graph $\mathcal{G}$ with $|S|$ components that diverge from the nodes of S. Then, the determinant $\det(\boldsymbol{L}_{-S}) = \sum_{F\in\mathcal{F}_S} w(F)$, where $w(F)$ represents the product of edge weights in forest F. Furthermore, for any nodes $i, j \in V \setminus S$, the (i,j)-cofactor $\boldsymbol{L}_{-S}^{ij} = \sum_{F\in\mathcal{F}_{S\cup\{i\}}^{ij}} w(F)$.*

Let $\boldsymbol{L}_q = \boldsymbol{M}^{-1} = \boldsymbol{Q}^{-1}(\boldsymbol{L} + \boldsymbol{Q})$, We analyze $\det(\boldsymbol{L}_q)$ as follows:

$$\begin{aligned}
\det(\boldsymbol{L}_q) = \det(\boldsymbol{Q}^{-1})\det(\boldsymbol{L} + \boldsymbol{Q}) &= \frac{1}{\prod_{v\in V} Q(v,v)}\det(\boldsymbol{L} + \boldsymbol{Q}) \\
&= \frac{1}{\prod_{v\in V} Q(v,v)} \sum_{t=0}^{n} \sum_{\substack{S\subseteq V \\ |S|=t}} \det(\boldsymbol{L}_{-S}) \prod_{u\in S} \boldsymbol{Q}(u,u) \\
&= \frac{1}{\prod_{v\in V} Q(v,v)} \sum_{t=0}^{n} \sum_{\substack{S\subseteq V \\ |S|=t}} \sum_{F\in\mathcal{F}_S} w(F) \prod_{u\in S} \boldsymbol{Q}(u,u) \\
&= \frac{1}{\prod_{v\in V} Q(v,v)} \sum_{S\subseteq V} \sum_{F\in\mathcal{F}_S} w(F) \prod_{u\in S} \boldsymbol{Q}(u,u) \\
&= \frac{1}{\prod_{v\in V} Q(v,v)} \sum_{F\in\mathcal{F}} w(F) \prod_{u\in r(F)} \boldsymbol{Q}(u,u).
\end{aligned}$$

For the (i,j)-cofactor $\boldsymbol{L}_q^{ij}$, we similarly obtain

$$\begin{aligned}
\det(\boldsymbol{L}_q^{ij}) &= \det((\boldsymbol{Q}^{-1})^{ii})\det((\boldsymbol{L}+\boldsymbol{Q})^{ij}) = \frac{Q(i,i)}{\prod_{v\in V}Q(v,v)}\det((\boldsymbol{L}+\boldsymbol{Q})^{ij}) \\
&= \frac{Q(i,i)}{\prod_{v\in V}Q(v,v)}\sum_{t=0}^{n-1}\sum_{\substack{S\subseteq V\setminus\{i,j\}\\|S|=t}}\det(\boldsymbol{L}_{-S}^{ij})\prod_{u\in S}\boldsymbol{Q}(u,u) \\
&= \frac{Q(i,i)}{\prod_{v\in V}Q(v,v)}\sum_{t=0}^{n-1}\sum_{\substack{S\subseteq V\setminus\{i,j\}\\|S|=t}}\sum_{F\in\mathcal{F}^{ij}_{S\cup\{i\}}}w(F)\prod_{u\in S}\boldsymbol{Q}(u,u) \\
&= \frac{Q(i,i)}{\prod_{v\in V}Q(v,v)}\sum_{S\subseteq V\setminus\{i,j\}}\sum_{F\in\mathcal{F}^{ij}_{S\cup\{i\}}}w(F)\prod_{u\in S}\boldsymbol{Q}(u,u) \\
&= \frac{1}{\prod_{v\in V}Q(v,v)}\sum_{F\in\mathcal{F}^{ij}}w(F)\prod_{u\in r(F)}\boldsymbol{Q}(u,u).
\end{aligned}$$

Since all edge weights in $\mathcal{G}$ are 1, the inverse $\boldsymbol{L}_q^{-1}$ simplifies to

$$\boldsymbol{L}_q^{-1}(i,j) = \frac{\det(\boldsymbol{L}_q^{ji})}{\det(\boldsymbol{L}_q)} = \frac{\sum_{F\in\mathcal{F}^{ji}}\prod_{u\in r(F)}Q(u,u)}{\sum_{F\in\mathcal{F}}\prod_{u\in r(F)}Q(u,u)}.$$

Finally, as $\rho_i = \sum_{v\in V}\boldsymbol{M}(v,i) = \sum_{v\in V}\boldsymbol{L}_q^{-1}(v,i)$, the expression for node i's structual centrality ρ_i follows

$$\rho_i = \sum_{j\in V}\frac{\sum_{F\in\mathcal{F}^{ij}}\prod_{u\in r(F)}Q(u,u)}{\sum_{F\in\mathcal{F}}\prod_{u\in r(F)}Q(u,u)}.$$

This completes the proof of Lemma 3.

A.5 Proof of Theorem 2

The time complexity of the algorithm depends on the times nodes are visited in loop-erased random walks. Thus, the time complexity of FOREST is

$$l\cdot\sum_{v\in V}\sum_{i=0}^{\infty}((\boldsymbol{I}-\boldsymbol{R})\boldsymbol{P})^i(v,v) = l\cdot\mathrm{Tr}((\boldsymbol{I}-(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{P})^{-1}) = l\cdot\mathrm{Tr}(\boldsymbol{M}\boldsymbol{R}^{-1}) \le \frac{l}{\alpha_{\min}}\mathrm{Tr}(\boldsymbol{M}).$$

Since $\boldsymbol{M}(i,j)\le 1$ for any $i,j\in V$, the final time complexity is obtained as $O(\frac{1}{\alpha_{min}}ln)$.

A.6 Proof of Corollary 1

Define the *ground-truth utility* of a set T as $U(T)=\sum_{i\in T}\rho_i(1-s_i)$ and the *estimated utility* as $\hat{U}(T)=\sum_{i\in T}\hat{\rho}_i(1-s_i)$.

By the error condition $|\rho_i-\hat{\rho}_i|\le\epsilon$, for any set T with $|T|=k$, we have:

$$\left|\hat{U}(T)-U(T)\right| = \left|\sum_{i\in T}(\hat{\rho}_i-\rho_i)(1-s_i)\right| \le \sum_{i\in T}\epsilon(1-s_i)\le k\epsilon,$$

where the last inequality holds because $0\le 1-s_i\le 1$. This implies

$$U(T)+k\epsilon \ge \hat{U}(T) \ge U(T)-k\epsilon. \tag{4}$$

Since we select $\hat{T}$ to maximize $\hat{U}(T)$, it satisfies

$$\hat{U}(\hat{T}) \ge \hat{U}(T^*) \ge U(T^*)-k\epsilon, \tag{5}$$

Combining (4) and (5), we bound $U(\hat{T})$ as:

$$U(\hat{T}) \geq \hat{U}(\hat{T}) - k\epsilon \geq (U(T^*) - k\epsilon) - k\epsilon = U(T^*) - 2k\epsilon.$$

Thus, $\sum_{i\in\hat{T}} \rho_i(1-s_i) \geq \sum_{i\in T^*} \rho_i(1-s_i) - 2k\epsilon$, as required.

A.7 Proof of Lemma 4

We demonstrate that the invariant holds by using induction. First, we verify that before any computation has begun, the invariant is satisfied by the initialized values:

$$\boldsymbol{\Delta} - \hat{\boldsymbol{\Delta}}^{(0)} = \boldsymbol{\Delta} - \mathbf{0} = (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \mathbf{1}.$$

Let $\boldsymbol{r}_a^{(t)}$ denote the residual vector before an update task for any node i, and let $\boldsymbol{r}_a^{(t+1)}$ denote the residual vector after the update task. Then a single update task corresponds to the following steps:

$$\begin{aligned}
\hat{\boldsymbol{\Delta}}^{(t+1)} &= \hat{\boldsymbol{\Delta}}^{(t)} + \boldsymbol{r}_a^{(t)}(i)(\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{R}\boldsymbol{e}_i, \\
\boldsymbol{r}_a^{(t+1)} &= \boldsymbol{r}_a^{(t)} - \boldsymbol{r}_a^{(t)}(i)\boldsymbol{e}_i + \boldsymbol{r}_a^{(t)}(i)\boldsymbol{P}^\top(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{e}_i = \boldsymbol{r}_a^{(t)} - \boldsymbol{r}_a^{(t)}(i)(\boldsymbol{I}-\boldsymbol{P}^\top(\boldsymbol{I}-\boldsymbol{R}))\boldsymbol{e}_i
\end{aligned}$$

Assuming that $\boldsymbol{\Delta} - \hat{\boldsymbol{\Delta}}^{(t)} = (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \boldsymbol{r}_a^{(t)}$, it follows that

$$\begin{aligned}
\boldsymbol{\Delta} - \hat{\boldsymbol{\Delta}}^{(t+1)} &= \boldsymbol{\Delta} - \hat{\boldsymbol{\Delta}}^{(t)} - \boldsymbol{e}_i^\top(\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{R}\boldsymbol{r}_a^{(t)} \\
&= (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \boldsymbol{r}_a^{(t)} - \boldsymbol{r}_a^{(t)}(i)(\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{R}\boldsymbol{e}_i \\
&= (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \left(\boldsymbol{r}_a^{(t+1)} + \boldsymbol{r}^{(t)}(i)(\boldsymbol{I}-\boldsymbol{P}^\top(\boldsymbol{I}-\boldsymbol{R}))\boldsymbol{e}_i\right) - \boldsymbol{r}_a^{(t)}(i)(\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{R}\boldsymbol{e}_i \\
&= (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \boldsymbol{r}_a^{(t+1)},
\end{aligned}$$

which completes the proof.

A.8 Proof of Lemma 5

On the one hand, since $\boldsymbol{r}_a \geq \mathbf{0}$, by Lemma 4, we have $\boldsymbol{\Delta}(v) \geq \hat{\boldsymbol{\Delta}}(v), \forall v \in V$; on the other hand, the condition in Line 2 of Algorithm 1 indicates that, after termination, the vector $\boldsymbol{r}_a \leq \epsilon\mathbf{1}$, then we have that:

$$\boldsymbol{\Delta} - \hat{\boldsymbol{\Delta}} = (\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \boldsymbol{r}_a \leq \epsilon(\mathbf{1}-\boldsymbol{s}) \odot \boldsymbol{M}^\top \mathbf{1} = \epsilon\boldsymbol{\Delta}. \tag{6}$$

The equation above implies that $(1-\epsilon)\boldsymbol{\Delta}(v) \leq \hat{\boldsymbol{\Delta}}(v), \forall v \in V$, hence the estimator $\hat{\boldsymbol{\Delta}}$ returned by Algorithm 1 satisfies $(1-\epsilon)\boldsymbol{\Delta}(v) \leq \hat{\boldsymbol{\Delta}}(v) \leq \boldsymbol{\Delta}(v), \forall v \in V$.

A.9 Proof of Lemma 6

We demonstrate that the invariant holds by using induction. First, we verify that before any computation has begun, the invariant is satisfied by the initialized values for any node $v \in V$:

$$\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) = \boldsymbol{\Delta}(v) - \hat{\boldsymbol{\Delta}}(v) = (1-\boldsymbol{s}(v)) \cdot \boldsymbol{e}_v^\top \boldsymbol{M}^\top \boldsymbol{r}_a = \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}^0.$$

Let $\boldsymbol{r}_s^{(t)}$ denote the residual vector before an update task for any node u, and let $\boldsymbol{r}_s^{(t+1)}$ denote the residual vector after the update task. Then a single update task corresponds to the following steps:

$$\begin{aligned}
\tilde{\Delta}^{(t+1)} &= \tilde{\Delta}^{(t)} + \boldsymbol{r}_a(u) \cdot \boldsymbol{r}_s^{(t)}(u), \\
\boldsymbol{r}_s^{(t+1)} &= \boldsymbol{r}_s^{(t)} - \boldsymbol{r}_s^{(t)}(u)\boldsymbol{e}_u + \boldsymbol{r}_s^{(t)}(u)(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{P}\boldsymbol{e}_u = \boldsymbol{r}_s^{(t)} - \boldsymbol{r}_s^{(t)}(u)(\boldsymbol{I}-(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{P})\boldsymbol{e}_u
\end{aligned}$$

Assuming that $\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}^{(t)}) = \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}_s^{(t)}$, it follows that

$$\begin{aligned}
\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}^{(t+1)}) &= \boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}^{(t)} + \boldsymbol{r}_a(u) \cdot \boldsymbol{r}_s^{(t)}(u)) \\
&= \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}_s^{(t)} - \boldsymbol{r}_a(u) \cdot \boldsymbol{r}_s^{(t)}(u) \\
&= \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\left(\boldsymbol{r}_s^{(t+1)} + \boldsymbol{r}_s^{(t)}(u)(\boldsymbol{I}-(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{P})\boldsymbol{e}_u\right) - \boldsymbol{r}_a(u) \cdot \boldsymbol{r}_s^{(t)}(u) \\
&= \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}_a^{(t+1)},
\end{aligned}$$

which completes the proof.

A.10 Proof of Lemma 7

On the one hand, since $\boldsymbol{r}_s \geq \mathbf{0}$, by Lemma 6, we have $\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) \geq 0$; on the other hand, the condition in Line 2 of Algorithm 2 indicates that, after termination, the vector $\boldsymbol{r}_s \leq \epsilon \boldsymbol{R}\mathbf{1}$, then we have that

$$\boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) = \boldsymbol{r}_a^\top \boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{r}_a \leq \epsilon \boldsymbol{r}_a^\top \boldsymbol{M}\mathbf{1} = \epsilon \cdot (\boldsymbol{r}_a)_{\text{sum}}.$$

Hence the estimator $\tilde{\Delta}$ returned by Algorithm 2 satisfies $0 < \boldsymbol{\Delta}(v) - (\hat{\boldsymbol{\Delta}}(v) + \tilde{\Delta}) \leq \epsilon \cdot (\boldsymbol{r}_a)_{\text{sum}}$.

A.11 Proof of Lemma 8

We present the proof for absolute errors; the relative error case follows analogously through error bound transformations and is thus omitted.

Let $\mathcal{S}^*$ denote the optimal size-k node set. We first show that any node $i \in T$ must belong to $\mathcal{S}^*$. By definition of T, we have $\hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{\langle k+1 \rangle} + \epsilon$. Applying the error bound yields $\boldsymbol{a}(i) \geq \hat{\boldsymbol{a}}(i) \geq \hat{\boldsymbol{a}}_{\langle k+1 \rangle} + \epsilon$. For any node $j \notin \mathcal{S}^*$, it holds that

$$\boldsymbol{a}(j) \leq \boldsymbol{a}_{\langle k+1 \rangle} \leq \hat{\boldsymbol{a}}_{\langle k+1 \rangle} + \epsilon \leq \boldsymbol{a}(i).$$

This shows $\boldsymbol{a}(i) \geq \boldsymbol{a}(j)$ for all $j \notin \mathcal{S}^*$, which implies $i \in \mathcal{S}^*$.

Next, we prove that $\mathcal{S}^* \setminus T \subseteq C$. Suppose for contradiction that there exists $i \in \mathcal{S}^*$ with $i \notin T \cup C$. By definition of C, this requires $\hat{\boldsymbol{a}}(i) < \hat{\boldsymbol{a}}_{\langle k \rangle} - \epsilon$. Using the error bound, we obtain

$$\boldsymbol{a}(i) \leq \hat{\boldsymbol{a}}(i) + \epsilon < \hat{\boldsymbol{a}}_{\langle k \rangle}.$$

However, since $i \in \mathcal{S}^*$, optimality implies $\boldsymbol{a}(i) \geq \boldsymbol{a}_{\langle k \rangle} \geq \hat{\boldsymbol{a}}_{\langle k \rangle}$, which contradicts the above inequality. Therefore, $\mathcal{S}^* \setminus T \subseteq C$ must hold.

A.12 Proof of Theorem 3

We first proof Lemma 11, which explains the upper bound of the time complexity of Algorithm 1.

Lemma 11. *An upper bound on the running time of Algorithm 1 is* $O(\frac{d_{\max}}{\alpha_{\min}} n \log \frac{1}{n})$.

Proof. During the processing, we add a dummy node that does not actually exist. This node is initially placed at the head of the queue and is re-appended to the queue each time it is popped. The set of nodes in the queue when this dummy node is processed for the $(i+1)$-th time is regarded as $S^{(i)}$, and the residue vector at this time is regarded as $\boldsymbol{r}_a^{(i)}$. The sum of the vector $\boldsymbol{r}_a^{(i)}$ is denoted as $(\boldsymbol{r}_a)_{\text{sum}}^{(i)}$. The process of handling this set is considered the $(i+1)$-th iteration. In the context of the $(i+1)$-th iteration, when node $v \in S^{(i)}$ is about to be processed, it holds that $\boldsymbol{r}_a(v) \geq \boldsymbol{r}_a^{(i)}(v)$. Upon the completion of this operation, the sum of residual vector is decreased by $\alpha_v \boldsymbol{r}_a(v)$. Consequently, by the conclusion of the $(i+1)$-th iteration, the total reduction in the sum of residue vector amounts to:

$$(\boldsymbol{r}_a)_{\text{sum}}^{(i)} - (\boldsymbol{r}_a)_{\text{sum}}^{(i+1)} = \sum_{v \in S^{(i)}} \alpha_v \boldsymbol{r}_a(v) \geq \sum_{v \in S^{(i)}} \alpha_v \boldsymbol{r}_a^{(i)}(v). \tag{7}$$

Given that the bound for any node v is ϵ, we obtain:

$$\frac{\sum_{v \in S^{(i)}} \boldsymbol{r}_a^{(i)}(v)}{|S^{(i)}|} \geq \epsilon \quad \text{and} \quad \frac{\sum_{v \notin S^{(i)}} \boldsymbol{r}_a^{(i)}(v)}{|V \setminus S^{(i)}|} \leq \epsilon.$$

Therefore, it follows that:

$$\frac{\sum_{v \in S^{(i)}} \boldsymbol{r}_a^{(i)}(v)}{|S^{(i)}|} \geq \frac{\sum_{v \in S^{(i)}} \boldsymbol{r}_a^{(i)}(v) + \sum_{v \notin S^{(i)}} \boldsymbol{r}_a^{(i)}(v)}{|S^{(i)}| + |V \setminus S^{(i)}|} = \frac{(\boldsymbol{r}_a)_{\text{sum}}^{(i)}}{n}.$$

Substituting the above expression into Eq. (7), we obtain

$$\begin{aligned}(\boldsymbol{r}_a)_{\text{sum}}^{(i+1)} &\leq (\boldsymbol{r}_a)_{\text{sum}}^{(i)} - \sum_{v\in S^{(i)}} \alpha_v \boldsymbol{r}_a^{(i)}(v) \leq (\boldsymbol{r}_a)_{\text{sum}}^{(i)} - \alpha_{\min} \sum_{v\in S^{(i)}} \boldsymbol{r}_a^{(i)}(v) \\ &\leq (1 - \frac{\alpha_{\min}}{n}|S^{(i)}|)(\boldsymbol{r}_a)_{\text{sum}}^{(i)} \\ &\leq \prod_{t=0}^{i}(1 - \frac{\alpha_{\min}}{n}|S^{(t)}|)(\boldsymbol{r}_a)_{\text{sum}}^{(0)}.\end{aligned}$$

By utilizing the fact that $1 - x \leq e^{-x}$, we obtain

$$(\boldsymbol{r}_a)_{\text{sum}}^{(i+1)} \leq \exp\left(-\frac{\alpha_{\min}}{n}(\sum_{t=0}^{i}|S^{(t)}|)\right) n. \tag{8}$$

Let $T^{(i+1)} = \sum_{t=0}^{i}|S^{(t)}|$ be defined as the total number of updates before the start of the $(i+1)$-th iteration. According to Eq. (8), to satisfy $(\boldsymbol{r}_a)_{\text{sum}}^{(i+1)} \leq \epsilon n$, it suffices to find the minimum number of updates that meets the following conditions:

$$\exp\left(-\frac{\alpha_{\min}}{n}T^{(i+1)}\right) \leq \epsilon \leq \exp\left(-\frac{\alpha_{\min}}{n}T^{(i)}\right).$$

Thus, we obtain $T^{(i)} \leq \frac{1}{\alpha_{\min}} n \log\frac{1}{\epsilon} \leq T^{(i+1)}$, Given the fact that $T^{(i+1)} - T^{(i)} = |S^{(i)}| \leq n$, we further derive

$$T^{(i+1)} \leq T^{(i)} + n \leq \frac{1}{\alpha_{\min}} n \log\frac{1}{\epsilon} + n.$$

For node v, the push operation reduces $(\boldsymbol{r}_a)_{\text{sum}}$ by $\alpha_v \boldsymbol{r}_a(v)$. Consequently, after incurring a total number of updates T, the reduction in $(\boldsymbol{r}_a)_{\text{sum}}$ is at least $\epsilon\alpha_v T$. Hence starting from the state of $(\boldsymbol{r}_a)_{\text{sum}} \leq \epsilon n$, the time cost T is bounded by $O(n/\alpha_{\min})$, thereby constraining the overall time complexity of Algorithm 1 to $O(\frac{d_{\max}^+}{\alpha_{\min}} n \log\frac{1}{\epsilon})$. □

For the refinement stage, we first proof the time complexity of Line 7 in Algorithm 3, when the absolute error parameter $\epsilon = 1$. Assuming that estimating node is $v \in V$, the contribution of $\boldsymbol{r}_s$ at node $u \in V$ each time is $\boldsymbol{r}_s(u)$. Due to the existence of boundary $\boldsymbol{h} = \frac{1}{(\boldsymbol{r}_a)_{\text{sum}}}\boldsymbol{R}\boldsymbol{1}$, its minimum value is α_u, and the upper bound on the total contribution is $(1 - \boldsymbol{s}(v))\boldsymbol{M}(u, v)$. Therefore, the upper bound on the number of updates at node u is $\frac{(\boldsymbol{r}_a)_{\text{sum}} \cdot (1-\boldsymbol{s}(v))\boldsymbol{M}(u,v)}{\alpha_u}$. Therefore, assuming that the candidate set returned in Line 3 of Algorithm 3 is C_a, the upper bound of the time complexity during the first round of execution in the refinement stage is

$$\begin{aligned}\sum_{v\in C_a}\sum_{u\in V} \frac{(\boldsymbol{r}_a)_{\text{sum}} \cdot (1 - \boldsymbol{s}(v))M(u, v)d_u^-}{\alpha_u} &\leq \frac{(\boldsymbol{r}_a)_{\text{sum}} \cdot d_{\max}^-}{\alpha_{\min}} \sum_{u\in V}\sum_{v\in C_a} M(u, v) \\ &= \frac{(\boldsymbol{r}_a)_{\text{sum}} \cdot d_{\max}^-}{\alpha_{\min}} \cdot \sum_{v\in C_a} \rho_v \leq \frac{\epsilon' \cdot d_{\max}^- n}{\alpha_{\min}} \cdot \sum_{v\in C_a} \rho_v.\end{aligned}$$

Now we consider when absolute error parameter $\epsilon = \frac{1}{2}, \frac{1}{4}, \ldots, \frac{1}{2n}, \ldots$. We assume that the error in one round of the process is ϵ', so the error in the previous round of the process is $2\epsilon'$. The upper bound of time complexity of this round is

$$\sum_{v\in C_a}\sum_{u\in V} \frac{2\epsilon'/(\boldsymbol{r}_a)_{\text{sum}} \cdot (1 - \boldsymbol{s}(v))(\boldsymbol{M}\boldsymbol{R}^{-1}\boldsymbol{R}\boldsymbol{1})(u)d_u^-}{\epsilon'/(\boldsymbol{r}_a)_{\text{sum}}\alpha_u} \leq \frac{2}{\alpha_{\min}} \sum_{v\in C_a}\sum_{u\in V} (\boldsymbol{M}\boldsymbol{1})(u)d_u^- = |C_a| \cdot \frac{2m}{\alpha_{\min}}$$

When the current error ϵ' is $\frac{1}{2i}$, the upper bound of the total time complexity from $\epsilon' = \frac{1}{2}$ to $\epsilon' = \frac{1}{2i}$ is $|C_a| \cdot \frac{2i \cdot m}{\alpha_{\min}}$. Hence, when $\epsilon' < k_{\text{gap}}$, the upper bound of the total time complexity is $|C_a| \cdot \frac{m}{k_{\text{gap}} \cdot \alpha_{\min}}$.

Hence, the upper bound of time complexity of Algorithm 3 is

$$O(\frac{d^+_{\max} n}{\alpha_{\min}}(\log\frac{1}{\epsilon} + \epsilon \sum_{v \in C_a} \rho_v) + \frac{|C_a| m}{k_{\text{gap}} \cdot \alpha_{\min}}).$$

Under normal circumstances, when ϵ is sufficiently small, we obtain $|C_a|$ as either 0 or a relatively small constant. Therefore, we can derive the upper bound of time complexity as $O(\frac{d^+_{\max} n}{\alpha_{\min}} \log\frac{1}{\epsilon} + \frac{m}{k_{\text{gap}} \cdot \alpha_{\min}})$ in the general case.

B Forest Sampling Algorithm

For a graph $\mathcal{G} = (V, E)$, a *spanning subgraph* of $\mathcal{G}$ is a subgraph with node set V and edge set being a subset of E. A *converging tree* is a weakly connected graph where exactly one node, called the *root*, has out-degree 0, while all other nodes have out-degree 1. An isolated node is considered a trivial converging tree with itself as the root. A *spanning converging forest* of $\mathcal{G}$ is a spanning subgraph in which every weakly connected component is a converging tree. This structure coincides with the notion of an *in-forest* as introduced by [50] and further studied by [51]. Spanning converging forests are closely related to the fundamental matrix of our model. Following the Lemma 3 and its Proof A.4, it can be shown that the any entry of the fundamental matrix can be represented in the form of spanning converging forest.

Wilson's algorithm [52] provides an efficient method for generating uniform spanning trees by leveraging loop-erased absorbing random walks. For any graph $\mathcal{G} = (V, E)$, Wilson's algorithm can be adapted to generate a uniform spanning converging forest, by using the method similar to that in [53, 30]. The details of the algorithm are presented in 4.

Algorithm 4: RANDOMFOREST($\mathcal{G}$, $\boldsymbol{R}$)

Input : graph $\mathcal{G}$, resistance matrix $\boldsymbol{R}$.
Output : root index vector RootIndex.

1 **for** $i \leftarrow 1$ **to** n **do**
2 InForest[i] $\leftarrow$ false; Next[i] $\leftarrow$ -1; RootIndex[i] $\leftarrow$ 0;
3 **for** $i \leftarrow 1$ **to** n **do**
4 $u \leftarrow i$;
5 **while** *not InForest[u]* **do**
6 **if** RAND() $\leq \alpha_u$ **then**
7 InForest[u] $\leftarrow$ true
8 Next[u] $\leftarrow -1$
9 RootIndex[u] $\leftarrow u$
10 **else**
11 $u \leftarrow$ Next[u] $\leftarrow$ RANDOMSUCCESSOR(u, $\mathcal{G}$)
12 RootNow $\leftarrow$ RootIndex[u]
13 **for** $u \leftarrow i$; *not InForest[u];* $u \leftarrow$ *Next[u]* **do**
14 InForest[u] $\leftarrow$ true
15 RootIndex[u] $\leftarrow$ RootNow
16 **return** RootIndex

The Algorithm 4 initializes three vectors: *InForest* (marks nodes added to the spanning forest), *Next* (tracks random walk steps), and *RootIndex* (records root assignments), all set to false, -1, and 0 respectively (Line 1). For each node in order (Line 3), it performs a loop-erased absorbing random walk that terminates either: (1) with probability α_u (making u a new root, Line 6-7), or (2) upon hitting existing forest nodes (Line 5). The walk's path is recorded in *Next*. After termination, the algorithm backtracks along the walk path (Line 13-14), adding all nodes to the forest and assigning them the current root index. This continues until all nodes are processed (Line 3), returning the *RootIndex* vector representing the sampled rooted spanning forest (Line 15). Then, based on Lemma 3, we obtained the implementation of our algorithm FOREST, and the pseudocode is shown in Algorithm 5.

Algorithm 5: Forest($\mathcal{G}$, $\boldsymbol{R}$, $\boldsymbol{s}$, l)

Input : graph $\mathcal{G}$, resistance matrix $\boldsymbol{R}$, internal opinion vector $\boldsymbol{s}$, sample count l.

Output : the target set $\hat{T}$.

Initialize : $\hat{T} \leftarrow \emptyset$; $\hat{\boldsymbol{\rho}} \leftarrow \mathbf{0}$

for $t \leftarrow 1$ **to** l **do**

  RootIndex $\leftarrow$ RANDOMFOREST($\mathcal{G}$, $\boldsymbol{R}$)

  for $i \leftarrow 1$ **to** n **do**

    $u \leftarrow$ RootIndex[i]

    $\hat{\boldsymbol{\rho}}_u \leftarrow \hat{\boldsymbol{\rho}}_u + 1$

$\hat{\boldsymbol{\rho}} \leftarrow \hat{\boldsymbol{\rho}}/l$

for $i = 1$ **to** n **do**

  $u \leftarrow \arg\max_{v \in V \setminus \hat{T}} \hat{\boldsymbol{\rho}}_v (1 - \boldsymbol{s}_v)$

  $\hat{T} \leftarrow \hat{T} \cup \{u\}$

return $\hat{T}$

C Additional Experiments

C.1 Efficiency

Figure 3 and Table 3 collectively present a comprehensive performance analysis of our proposed MIS algorithm under different parameters and network conditions. Figure 3 demonstrates the algorithm's runtime performance across all tested networks under various resistance coefficient distributions as parameter k varies. The results indicate that although fluctuations in the potential influence distribution near the k-th most influential node cause minor variations, MIS consistently maintains superior performance compared to both RWB and FOREST algorithms in all test cases.

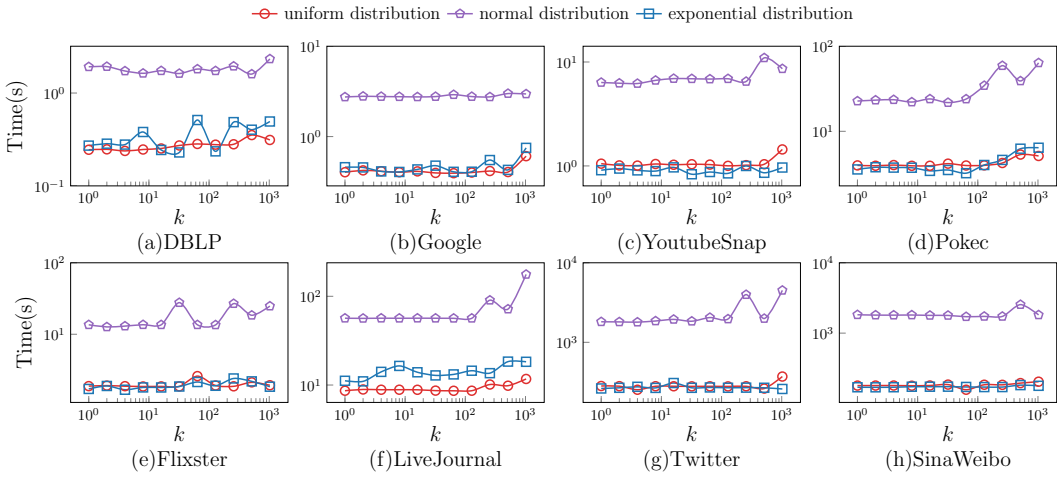


Figure 3: Running time of algorithm MIS under different resistance coefficient distributions.

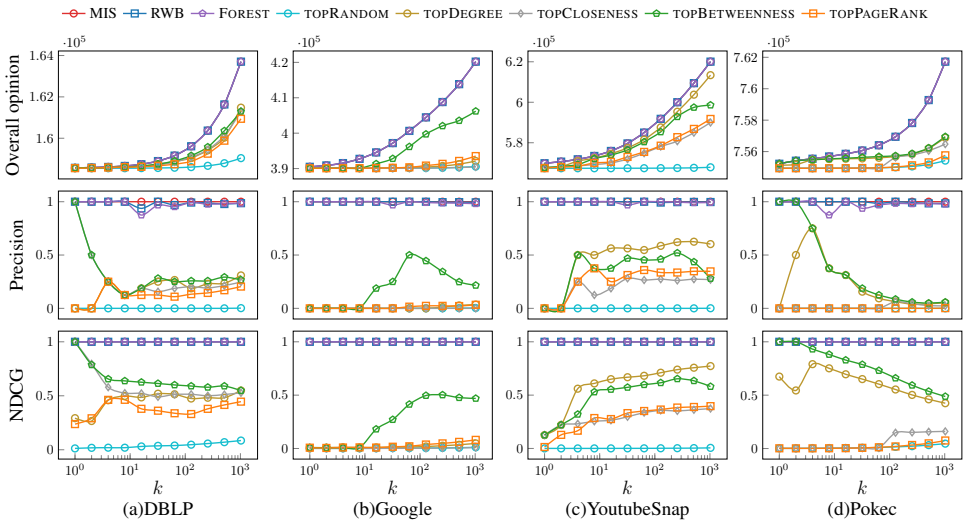
Complementing the runtime analysis, Table 3 evaluates computational efficiency by examining the average number of updates per node during algorithm convergence under the specific parameter setting of k=64 and uniform resistance coefficients. The update ratios for all networks remain within a practical range of 27.06 to 249.28, with Twitter maintaining controllable levels despite its largest scale while Google achieves optimal efficiency. These results confirm that the algorithm achieves near-linear time complexity in practice, with stable constant factors that do not increase significantly with network size, demonstrating strong scalability for large-scale network applications.

Table 3: Updates per node across networks.

Network	DBLP	Google	YoutubeSnap	Pokec	Flixster	LiveJournal	Twitter	SinaWeibo
Ratio	49.93	27.06	49.34	126.86	114.84	106.05	249.28	78.77

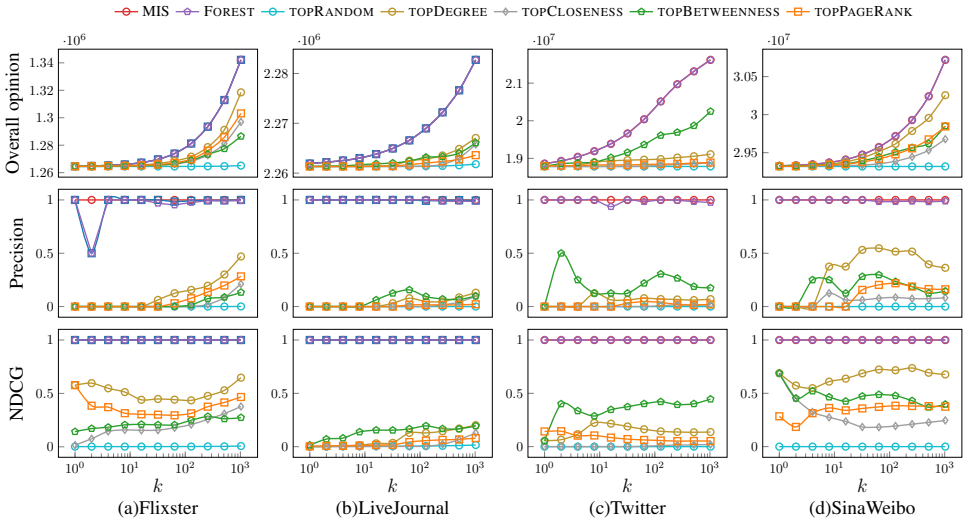
C.2 Accuracy

Extending the analysis from Figures 1 and 2, we conduct a comprehensive evaluation of accuracy performance across multiple experimental configurations. Figures 4 and 5 examine performance under normal distribution conditions. Similarly, Figures 6 and 7 demonstrate algorithm behavior under exponential distribution. The complete set of experimental results reveals several important findings. First, our proposed algorithms (RWB, FOREST, and MIS) consistently outperform all baseline methods across every tested condition. While the RWB algorithm fails to complete execution within practical time limits for certain cases due to its computational complexity, the algorithms FOREST and MIS successfully deliver results in all scenarios. More significantly, the MIS algorithm achieves exact solutions in every case while simultaneously maintaining near-perfect accuracy (NDCG ≈ 1) in sequence ordering. This combination of guaranteed precision and optimal ordering performance clearly establishes the superiority of our MIS approach over both baseline methods and our own alternative proposals.



(a)DBLP (b)Google (c)YoutubeSnap (d)Pokec

Figure 4: Performance of algorithms in accuracy under normal distribution.



(a)Flixster (b)LiveJournal (c)Twitter (d)SinaWeibo

Figure 5: Performance of algorithms in accuracy under normal distribution.

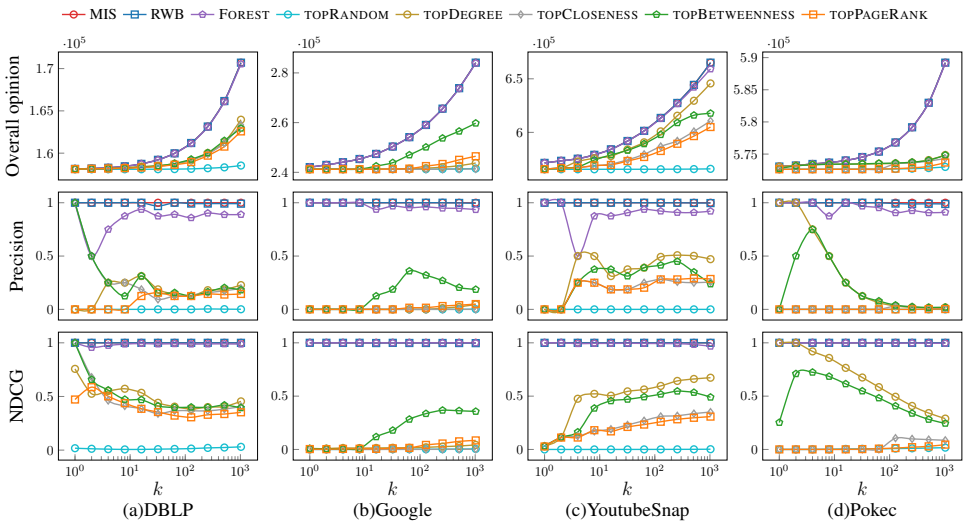


Figure 6: Performance of algorithms in accuracy under exponential distribution.

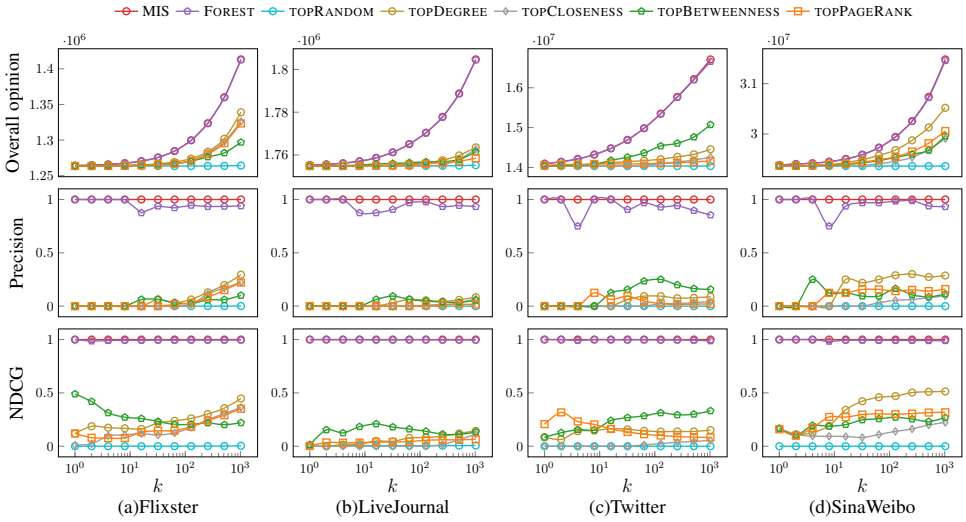


Figure 7: Performance of algorithms in accuracy under exponential distribution.

D Dataset Details

The study utilizes eight benchmark datasets obtained from the Koblenz Network Collection [45], SNAP [46], and Network Repository [47].

DBLP is an undirected co-authorship network where nodes represent authors and edges indicate co-authorship relationships. The dataset uses the largest connected component, with publication venues (conferences or journals) defining ground-truth communities, retaining the top 5,000 high-quality communities each containing at least three nodes. Google is a directed network where nodes represent web pages and edges represent hyperlinks, originating from the 2002 Google Programming Contest. YouTube is an undirected social network where nodes represent users and edges represent friendships, with communities defined by user-created groups, similarly retaining the top 5,000 communities with at least three nodes and using the largest connected component. Pokec is a Slovak social network where nodes represent users and directed edges represent friendships, containing anonymized user attributes such as gender and age. Flixster is an undirected movie social network where nodes represent users and edges represent social connections. LiveJournal is a directed online community network where nodes represent users and edges represent friend relationships. Twitter is a directed social network where nodes represent users and edges represent follower relationships. SinaWeibo is a directed microblogging social network where nodes represent users and edges represent social connections. Detailed statistical characteristics of each network are provided in Table 1.