Efficient Influence Minimization via Node Blocking

Jinghao Wang
Zhejiang Gongshang University
University of Technology Sydney
jinghaow.au@gmail.com

Yanping Wu
University of Technology Sydney
yanping.wu@student.uts.edu.au

Xiaoyang Wang
The University of New South Wales
xiaoyang.wang1@unsw.edu.au

Ying Zhang
Zhejiang Gongshang University
ying.zhang@zjgsu.edu.cn

Lu Qin
University of Technology Sydney
lu.qin@uts.edu.au

Wenjie Zhang
The University of New South Wales
wenjie.zhang@unsw.edu.au

Xuemin Lin
Shanghai Jiaotong University
xuemin.lin@sjtu.edu.cn

ABSTRACT

Given a graph G, a budget k and a misinformation seed set S, *Influence Minimization* (IMIN) via node blocking aims to find a set of k nodes to be blocked such that the expected spread of S is minimized. This problem finds important applications in suppressing the spread of misinformation and has been extensively studied in the literature. However, existing solutions for IMIN still incur significant computation overhead, especially when k becomes large. In addition, there is still no approximation solution with non-trivial theoretical guarantee for IMIN via node blocking prior to our work. In this paper, we conduct the first attempt to propose algorithms that yield data-dependent approximation guarantees. Based on the Sandwich framework, we first develop submodular and monotonic lower and upper bounds for our non-submodular objective function and prove the computation of proposed bounds is #P-hard. In addition, two advanced sampling methods are proposed to estimate the value of bounding functions. Moreover, we develop two novel martingale-based concentration bounds to reduce the sample complexity and design two non-trivial algorithms that provide $(1-1/e-\epsilon)$-approximate solutions to our bounding functions. Comprehensive experiments on 9 real-world datasets are conducted to validate the efficiency and effectiveness of the proposed techniques. Compared with the state-of-the-art methods, our solutions can achieve up to two orders of magnitude speedup and provide theoretical guarantees for the quality of returned results.

PVLDB Reference Format:
Jinghao Wang, Yanping Wu, Xiaoyang Wang, Ying Zhang, Lu Qin, Wenjie Zhang, and Xuemin Lin. Efficient Influence Minimization via Node Blocking. PVLDB, 17(10): 2501 - 2513, 2024.
doi:10.14778/3675034.3675042

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/wjh0116/IMIN.

Proceedings of the VLDB Endowment, Vol. 17, No. 10 ISSN 2150-8097.
doi:10.14778/3675034.3675042

1 INTRODUCTION

With the rapid development of the Internet, various online social networks (OSNs) have thrived, immensely satisfying and facilitating the need for individuals to share their perspectives and acquire information. Leveraging the established connections between individuals, information and opinions can spread through word-of-mouth effects across OSNs [42–44]. However, immense user bases and rapid sharing abilities also make OSNs effective channels for spreading misinformation, which leads to significant harm, such as economic damages and societal unrest [3, 28]. Therefore, it is necessary to implement a series of strategies to minimize the spread of misinformation. In the literature, strategies for addressing this problem can be categorized into three types: *i*) positive information spreading [6, 39], which selects a set of nodes to trigger the spread of positive information to fight against the spread of misinformation; *ii*) edge blocking [20, 22], which removes a set of edges to decrease the spread of misinformation; *iii*) node blocking [41, 46], which removes a set of critical nodes to limit the spread of misinformation.

In this paper, we consider the problem of *Influence Minimization* (IMIN) via node blocking [41, 46]. Specifically, given a graph G, a seed set S of misinformation and a budget k, IMIN aims to find a set of k nodes to be removed such that the expected spread of S is minimized. These removed nodes are called blockers. Note that, removing a node causes some nodes previously reachable by misinformation to become unreachable. We call that these nodes are *protected* by the blocker. In such a condition, IMIN equals to identify a blocker set with at most k nodes so that the expected number of protected nodes is maximized.

The IMIN problem is NP-hard and APX-hard unless P=NP [46]. Wang et al. [41] first study IMIN via node blocking under the IC model. They use Monte-Carlo simulations to estimate the expected decreased spread of misinformation seed set and provide a greedy algorithm to select blockers, i.e., iteratively select the node that leads to the largest decreased spread. However, the proposed solution is prohibitively expensive on large social networks due to the inefficiency of Monte-Carlo simulations. Recently, Xie et al. [46] propose a novel approach based on the dominator tree (formal definition can be found in Section 2.2) that can effectively and efficiently estimate the decreased spread of misinformation seed set. They also adopt the greedy algorithm, but the difference is that they

use the newly proposed estimation method. They observe that the greedy method may miss selecting some critical nodes. Therefore, they propose a more effective modified greedy algorithm, which prioritizes the outgoing neighbors of misinformation seed nodes.

However, the solutions in [46] require re-estimating the value of decreased spread of misinformation seed set after selecting a blocker in each iteration, which results in prohibitive computation overhead, especially when the budget becomes large. Besides, since the objective function of IMIN is non-submodular [46], directly adopting the greedy method cannot provide $(1-1/e)$-approximate solutions [29]. Moreover, based on curvature and submodularity ratio [4], we show that the result returned by greedy does not provide any non-trivial guarantees. Therefore, it remains an open problem to devise efficient algorithms for IMIN with non-trivial theoretical guarantees.

In this paper, we address this problem based on the Sandwich approximation strategy [27, 45], which is a widely used framework for non-submodular maximization problems. The general idea of Sandwich framework is to first develop monotone nondecreasing and submodular lower and upper bounds for the objective function studied, and then produce solutions with approximation guarantees (breads of the Sandwich) for bounding functions maximization (i.e., maximize the lower and upper bounding functions). The actual effectiveness of the Sandwich-based approach relies on how close the proposed bounds are to the objective function. In other words, a loose submodular bound w.r.t. the objective function can also be applied to solve our problem, but it cannot produce satisfactory results in terms of effectiveness, and would only yield trivial data-dependent approximation factor. For example, a constant function can serve as a trivial submodular upper bound (e.g., an upper bound that equals the number of nodes in the graph). However, it is apparent that this bounding function may adversely affect our results since the solution to the constant function can be arbitrary. Thus, to provide high-quality results, tight bounds with submodularity property for our objective function are required.

Following the Sandwich framework, we first propose appropriate lower and upper bounds for our functions, and prove that the computation of them is #P-hard. Additionally, the widely used technique for influence estimation, *Reverse Influence Sampling* (RIS) [5], cannot directly extend to the proposed bounding functions estimation. This is because, different from the classic RIS, which treats all the nodes equally and samples the node uniformly, we need to focus on the nodes who are prone to be affected by the misinformation, since only those nodes can contribute to the bounding functions. That is, if we estimate the bounding functions using a similar manner of RIS, we need to sample the node based on its probability of being activated by the misinformation. However, this probability is #P-hard to compute [8]. To overcome this issue, we propose two novel unbiased estimators based on two new proposed sample sets, i.e., CP sequence and LRR set, to estimate the value of lower and upper bounding functions, respectively.

For maximizing the bounding functions with theoretical guarantees, a straightforward approach is to employ OPIM-C [33], which is RIS-based and the state-of-the-art method for *Influence Maximization* (IM). However, given that RIS cannot be applied to estimate the bounding functions, we cannot inherit the sample complexity from OPIM-C. To tackle this challenge, we first design two novel martingale-based concentration bounds tailored to the new proposed unbiased estimators. By utilizing these, the sample complexity required to make an unbiased estimate of the bounding functions is significantly reduced, in comparison to the previous concentration bounds used in OPIM-C. Moreover, to avoid the new derived sample complexity depending on the expected spread of the misinformation $\mathbb{E}[I_G(S)]$, whose computation is #P-hard, we resort to the generalized stopping rule algorithm in [47], to obtain the value of estimated $\mathbb{E}[I_G(S)]$. Based on the above analysis, we design two non-trivial algorithms, LSBM and GSBM, to maximize lower and upper bounding functions with a provable approximation guarantee of $(1-1/e-\epsilon)$ with high probability, respectively. Finally, we propose a lightweight heuristic LHGA for IMIN, to serve as the filling of the Sandwich. By instantiating the Sandwich with LSBM, GSBM and LHGA, our proposed solution SandIMIN can offer a strong theoretical guarantee for the IMIN problem (details can be found in Section 6.4). Experiments over 9 real-world graphs are conducted to verify the efficiency and effectiveness of proposed techniques compared with the state-of-the-art solutions [46]. The main contributions of the paper are summarized as follows.

- In this paper, based on the Sandwich search framework, we propose a novel solution SandIMIN for the influence minimization problem via node blocking. To the best of our knowledge, we are the first to propose algorithms that yield approximation guarantees for the problem. Submodular and monotonic lower and upper bounds are designed for the objective function, and we prove the computation of bounding functions is #P-hard.
- To estimate the bounds proposed, two novel sample sets and the corresponding sampling techniques are proposed. In addition, new martingale-based concentration bounds are developed to reduce the sample complexity and improve the overall performance. Furthermore, we propose two non-trivial algorithms to maximize lower and upper bounding functions, which provide $(1-1/e-\epsilon)$ approximation guarantee with high probability.
- We conduct extensive experiments on 9 real-world graphs to verify the efficiency and effectiveness of proposed techniques. Compared with the state-of-the-art algorithms [46], our solutions show better scalability in terms of dataset size and parameters, and can achieve up to two orders of magnitude speedup.

Note that, due to the limited space, all the proofs are omitted and can be found in the full version [40].

2 PRELIMINARIES

In this section, we first formally define the *Influence Minimization* (IMIN) problem, and then we present an overview of existing solutions for the IMIN problem.

2.1 Problem Definition

We consider a directed graph $G = (V, E)$ with a node set V and a directed edge set E, where $|V| = n$ and $|E| = m$. Given an edge $\langle u, v\rangle \in E$, we refer to u as an incoming neighbor of v and v as an outgoing neighbor of u. Each edge $\langle u, v\rangle$ is associated with a propagation probability $p(u, v) \in [0, 1]$, representing the probability that u influences v. Table 1 summarizes the notations frequently used.

Diffusion model. In this paper, we focus on the *independent cascade (IC)* model, which is widely used to simulate the information

Table 1: Frequently used notations

Notation	Description
$G = (V, E)$	a social network with node set V and edge set E
S, B	the seed set of misinformation and blocker set
$\mathbb{E}[I_G(S)]$	the expected spread of seed set S
$G[V']$	the subgraph in G induced by node set V'
ϕ, Ω	a realization and the set of all possible realizations
$D_S(B)$	the expected decreased spread of seed set S after blocking nodes in B
$D_S^L(\cdot), D_S^U(\cdot)$	the submodular and monotonic lower bound and upper bound of $D_S(\cdot)$
B_L^o, B_U^o, B^o	the optimal solution to the lower bounding function, upper bounding function and objective function
$C^s, \mathbb{C}^s$	a CP sequence and the set of CP sequences
$L(v), \mathbb{L}$	a LRR set of v and the set of LRR sets

diffusion in the literature [5, 19, 33, 43, 44]. Given a seed set $S \subseteq V$, the diffusion process of S under the IC model unfolds in discrete timestamps, whose details are shown in the following.

- At timestamp 0, the nodes in the seed set S are activated, while all other nodes are inactive. Each activated node will remain active in the subsequent timestamps.
- If a node u is activated at timestamp t, for each of its inactive outgoing neighbor v, u has a single chance to activate v with probability $p(u, v)$ at timestamp $t + 1$.
- The propagation process stops when no more nodes can be activated in the graph G.

Given a seed set $S \subseteq V$, let $I_G(S)$ be the number of active nodes in G when the propagation process stops. Alternatively, the diffusion process can also be characterized as the *live edge* procedure [19]. Specifically, by removing each edge $\langle u, v\rangle \in E$ with $1 - p(u, v)$ probability, the remaining graph is referred to as a *realization*, denoted as ϕ. Let $I_\phi(S)$ denote the number of nodes that are reachable from S in ϕ. For any seed set S, its expected spread $\mathbb{E}[I_G(S)]$ can be defined as follows.

$$\mathbb{E}[I_G(S)] = \mathbb{E}_{\Phi\sim\Omega}[I_\Phi(S)] = \sum_{\phi\in\Omega} I_\phi(S) \cdot p(\phi), \tag{1}$$

where Ω is the set of all possible realizations of G, $\Phi \sim \Omega$ denotes that Φ is a random realization sampled from Ω and $p(\phi)$ is the probability for realization ϕ to occur.

In this paper, we study the problem of minimizing the spread of misinformation. One strategy for influence minimization problem is to *block critical nodes* on social networks [41, 46]. When a node u is blocked, we set the probability of all edges pointing to u as 0 and refer to u as a blocker. We can obtain that the activation probability of a blocker is 0. Additionally, we assume that a blocker cannot be a seed node for propagating misinformation. Note that, after blocking a node v, the status of some nodes changes from active to inactive. We call these nodes are *protected* by v. Given a seed set $S \subseteq V$ and a blocker set $B \subseteq (V\backslash S)$, we denote $D_S(B) = \mathbb{E}[I_G(S)] - \mathbb{E}[I_{G[V\backslash B]}(S)]$ as the expected decreased spread of seed set S after blocking nodes in B, where $G[V\backslash B]$ denotes the subgraph in G induced by node set $V\backslash B$.

Problem statement. Given a directed social network $G = (V, E)$, a seed set S for propagating misinformation and a budget k, *Influence Minimization* (IMIN) via node blocking is to find a blocker set B^* with at most k nodes such that the influence (i.e., expected spread) of seed set S is minimized after blocking nodes in B^*. In other words, IMIN aims to identify a blocker set B^* with at most k nodes such that the expected number of protected nodes is maximized, i.e.,

$$B^* = \underset{B\subseteq(V\backslash S),|B|\le k}{\arg\max} D_S(B).$$

As shown in [46], IMIN is proved to be NP-hard and APX-hard unless P=NP. In addition, given a seed set S, $D_S(\cdot)$ is monotonic, but not submodular. Due to the non-submodularity property of the IMIN objective, the direct use of the greedy framework cannot return a result with an approximation ratio of $(1 - 1/e)$ [29].

2.2 Existing Solutions Revisited

Here we first abstract the greedy framework from recent studies [41, 46], and then we introduce the state-of-the-art approaches for addressing the IMIN problem and show their limitations.

Greedy framework for IMIN. Suppose $D_S(u|B) = D_S(B \cup \{u\}) - D_S(B)$ as the marginal gain of adding u to the set B. In a nutshell, the greedy framework [41] starts from an empty blocker set $B = \emptyset$. The subsequent part of the algorithm consists of k iterations. At each iteration, it iteratively selects the node v from $V\backslash S$ that leads to the largest $D_S(v|B)$ and adds it into B. After selecting one blocker, the probability of all edges pointing to it is set as 0.

Due to $D_S(B) = \mathbb{E}[I_G(S)] - \mathbb{E}[I_{G[V\backslash B]}(S)]$, one feasible way for calculating $D_S(B)$ is to compute $\mathbb{E}[I_G(S)]$. However, the computation of $\mathbb{E}[I_G(S)]$ is proved as #P-hard [8], which means that the computation of $D_S(B)$ is also #P-hard. In [41], Wang et al. use Monte-Carlo simulations to estimate the influence spread $\mathbb{E}[I_G(S)]$ and adopt the greedy framework. It can solve the IMIN problem effectively but due to the inefficiency of Monte-Carlo simulations, it incurs significant computation overhead.

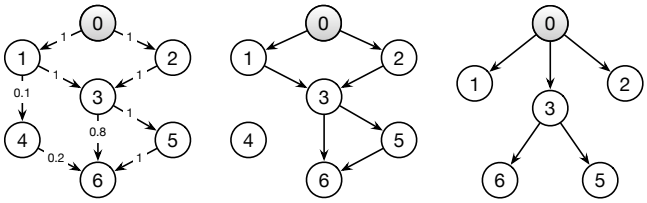
The state-of-the-art approach. Compared to the Monte-Carlo based estimation method under the greedy framework, the state-of-the-art approach [46] optimizes the estimation for $D_S(\cdot)$. Instead of estimating the expected spread (i.e., $\mathbb{E}[I_G(S)]$), Xie et al. [46] directly estimate the expected decreased spread (i.e., $D_S(\cdot)$) based on the *dominator tree* (DT) [2, 26]. To explain how this estimation algorithm works, we first introduce three concepts as follows.

Definition 2.1 (Dominator). Given a realization ϕ and a source s, a node u is called a dominator of a node v if and only if all paths from s to v pass through u.

Definition 2.2 (Immediate Dominator). Given a realization ϕ and a source s, a node u is said to be an immediate dominator of a node v, denoted as $idom(v) = u$, if and only if u dominates v and every other dominator of v dominates u.

Definition 2.3 (Dominator Tree (DT)). Given a realization ϕ and a source s, the dominator tree of ϕ is induced by the edge set $\{\langle idom(u), u\rangle : u \in V\backslash\{s\}\}$ with root s.

According to the concept of DT, each DT has one root. Xie et al. [46] first propose to create a unified seed node s to replace the given seed set S under the IC model. For each node $u \in (V\backslash S)$, if there are h distinct seed nodes pointing to node u, and each edge has a probability p_i $(1 \le i \le h)$, they will remove all edges from the seed nodes to node u, and add an edge from node s to node u



(a) Social network G **(b) A realization ϕ** **(c) DT of ϕ**

Figure 1: Example of estimating $D_s(\cdot)$

with the probability $(1 - \prod_{i=1}^{h}(1 - p_i))$. Correspondingly, $D_S(\cdot)$ is replaced by $D_s(\cdot)$. In addition, Xie et al. [46] prove that for each node $u \in (V \backslash S)$, the expected number of protected nodes by u (i.e., $D_s(u)$) equals the expected size of the subtree rooted at u in the DT. Based on these, the estimation algorithm of [46] runs in the following steps, where $\hat{D}_s(\cdot)$ is the estimated value of $D_s(\cdot)$.

- Generate a certain number of random realizations $\mathcal{G}$ from G.
- For each generated realization $\phi \in \mathcal{G}$, apply Lengauer-Tarjan algorithm [23] to construct the DT of ϕ, which roots at s. For each node $v \in (V \backslash S)$, measure the size of subtree with the root v in DT, which is denoted as $c_\phi(v)$.
- For each node $v \in (V \backslash S)$, the estimated value of $D_s(v)$ is the average value of $c_\phi(v)$ in all realizations that are generated, i.e., $\hat{D}_s(v) = (\sum_{\phi \in \mathcal{G}} c_\phi(v))/|\mathcal{G}|$.

Example 2.1. Here we illustrate an example of the above estimation procedure with one realization. Figure 1(a) shows a social network $G = (V, E)$, where v_0 is the misinformation seed node. The number associated with each edge is its corresponding propagation probability. Figure 1(b) shows a realization ϕ obtained from G. In Figure 1(c), a DT rooted at s of ϕ is constructed by Lengauer-Tarjan algorithm [23]. Then, we can get the estimated value of $D_s(v)$ for each node $v \in (V \backslash \{v_0\})$ by measuring the size of subtree with root v in DT, i.e., $\hat{D}_s(v_1) = \hat{D}_s(v_2) = \hat{D}_s(v_5) = \hat{D}_s(v_6) = 1$, $\hat{D}_s(v_3) = 3$, and $\hat{D}_s(v_4) = 0$ since v_4 cannot be activated by v_0 in ϕ.

By utilizing the greedy framework but with one difference, Xie et al. [46] propose AdvancedGreedy (AG). That is, they use the above estimation method to obtain the estimated value of $D_s(\cdot|B)$. However, some critical nodes may be missed by AG, e.g., some outgoing neighbors of the seed nodes. Reconsider Figure 1(a), suppose $k = 2$. v_0 has two outgoing neighbors and if we directly select these two nodes as blockers, v_0 cannot activate any node. However, if we block the blockers returned by AG (i.e., v_3 and v_1), v_2 cannot be protected. To address this issue, Xie et al. [46] further propose GreedyReplace (GR), which consists of two stages. In the first stage, the outgoing neighbors of seed nodes are stored in the candidate set. They iteratively select the node v from the candidate set with the largest $\hat{D}_s(v|B)$ and add it into B, until $|B| = \min\{d_s^{out}, k\}$, where d_s^{out} denotes the number of nodes in the candidate set. In the second stage, they consider processing the blockers in B according to the reverse order of their insertion order. For each blocker in B, they first remove it from B, called the replaced node. Then they select the node v from $V \backslash S$ with the largest $\hat{D}_s(v|B)$ and add it into B, called the current best blocker. If the replaced node is the current best blocker, they return B directly. Otherwise, continue the replacement process. Compared with AG, GR can achieve a better result quality. However, in most cases, AG is more efficient since GR requires two stages to select nodes, and often requires multiple rounds of replacement process in the second stage before it terminates, resulting in additional time cost for GR.

Limitations. Despite the efficiency of dominator tree based estimation method, AG and GR still incur significant computation overhead in practice. This is because when AG/GR selects a node as a blocker, it needs to remove that node from the graph. Therefore, AG/GR cannot reuse the realizations generated in the last round. Correspondingly, Xie et al. need to regenerate realizations and construct the corresponding DTs, which incurs significant time cost for large values of k. Moreover, there is no theoretical analysis provided for AG and GR, which are both based on the greedy framework. Based on the curvature and submodularity ratio [4], in this paper, we analyze the approximation guarantee for the greedy strategy on our non-submodular objective. The submodularity ratio serves as a metric to assess how closely the objective approximates being submodular. Formally, for all $A, B \subset V$, the submodularity ratio of $D_s(\cdot)$ is the largest scalar ψ such that,

$$\sum_{\omega \in A \backslash B}[D_s(B \cup \{\omega\}) - D_s(B)] \geq \psi[D_s(B \cup A) - D_s(B)]. \quad (2)$$

Considering an example with the graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and $E = \{\langle v_0, v_1\rangle, \langle v_0, v_2\rangle, \langle v_1, v_3\rangle, \langle v_2, v_3\rangle, \langle v_3, v_4\rangle, \langle v_3, v_5\rangle, \ldots, \langle v_3, v_{n-1}\rangle\}$. The probability on each edge is set to 1 and v_0 is the misinformation seed node. When $B = \emptyset$ and $A = \{v_1, v_2\}$, the left side of Eq. (2) is equal to 2, and the right side of Eq. (2) is equal to $n - 1$, i.e., $2 \geq \psi \cdot (n - 1)$. As n gradually becomes larger, the submodularity ratio ψ approaches 0 infinitely, consequently using the greedy strategy yields approximation guarantee that also approaches 0 infinitely [4]. Thus, the result returned by the state-of-the-art algorithms for IMIN does not provide any non-trivial guarantees. To fill these gaps, in this paper, we design efficient approximation algorithms with theoretical guarantees for IMIN.

3 SANDWICH APPROXIMATION STRATEGY

To solve the IMIN problem, we propose efficient approximation algorithms based on the Sandwich framework [27, 45], which is widely used for non-submodular maximization. In Section 3.1, we first give the general framework of our algorithm. Then we propose the lower and upper bounds in Section 3.2 and 3.3, respectively.

3.1 Overview of SandIMIN

Generally, our SandIMIN algorithm first finds the α_1-approximate solution and α_2-approximate solution (breads of Sandwich) to the lower bound and the upper bound of the objective function, respectively. Then, it finds a solution (filling of Sandwich) to the original problem with a heuristic method. Finally, it returns the best solution among these three results. The pseudocode of the above process is shown in Algorithm 1 and has the following result,

$$D_s(B) \geq \max\left\{\frac{D_s^L(B_L^o)}{D_s(B^o)}\alpha_1, \frac{D_s(B_U)}{D_s^U(B_U)}\alpha_2\right\}\frac{1-\gamma}{1+\gamma}D_s(B^o), \quad (3)$$

where D_s^L, D_s^U are the non-negative, monotonic and submodular set functions defined on V, i.e., $D_s^L : 2^V \to \mathbb{R}_{\geq 0}$ and $D_s^U : 2^V \to \mathbb{R}_{\geq 0}$, such that $\forall B \subseteq (V \backslash S)$, $D_s^L(B) \leq D_s(B) \leq D_s^U(B)$. B_L^o, B_U^o and B^o

Algorithm 1: SandIMIN

Input : The graph $G = (V, E)$, the seed set S, the unified seed node s, the budget k and the error parameters α_1, α_2, δ, γ.
Output : The blocker set B.

// LSBM Algorithm in Section 5.1
1 $B_L \leftarrow$ the α_1-approximate solution for lower bounding function maximization;
// GSBM Algorithm in Section 5.2
2 $B_U \leftarrow$ the α_2-approximate solution for upper bounding function maximization;
// LHGA Algorithm in Section 5.3
3 $B_R \leftarrow$ the heuristic solution for original problem;
4 $\hat{I}_{G[V\setminus\cdot]}(s) \leftarrow$ the (γ, δ)-estimate of $\mathbb{E}[I_{G[V\setminus\cdot]}(s)]$;
5 $B \leftarrow \arg\min_{B^* \in \{B_L, B_U, B_R\}} \hat{I}_{G[V\setminus B^*]}(s)$;
6 **return** B;

are the optimal solutions to maximize the lower bounding function, upper bounding function and objective function, respectively. In addition, we call $\hat{\mu}$ is the (γ, δ)-estimate of μ if $\hat{\mu}$ satisfies:

$$\Pr[(1-\gamma)\mu \le \hat{\mu} \le (1+\gamma)\mu] \ge 1-\delta. \tag{4}$$

As observed, the key to SandIMIN is to find the lower and upper bounds of the objective function, which are both monotonic and submodular. Before presenting our bounds, we first introduce the technique of how to transfer multiple seeds to one seed for presentation simplicity. We create a unified seed node s and then introduce the edges with the propagation probability of 1 from s to every seed node $v \in S$. Note that, s is the virtual node and it does not belong to V. Under such a setting, we can guarantee that s and S have the same spread under the IC model and there is no need to pre-compute $1 - \prod_{i=1}^{h}(1-p_i)$ for each node $u \in (V\setminus S)$ as stated in Section 2.2. In the following, we use $D_s(\cdot)$ to denote $D_S(\cdot)$.

Roadmap of SandIMIN. In Section 3.2 and Section 3.3, we propose submodular and monotonic lower and upper bounds of the objective function, respectively. We then design sampling methods to estimate the value of lower and upper bounds in Section 4.1 and Section 4.2, respectively. In Section 5.1 and 5.2, we devise two approximation algorithms that provide $(1 - 1/e - \epsilon)$-approximate solutions for lower and upper bounding functions maximization, respectively (Lines 1-2 of Algorithm 1). In Section 5.3, we devise a heuristic method for IMIN (Line 3 of Algorithm 1) and show that SandIMIN yields a data-dependent approximation guarantee.

3.2 Lower Bound

A function $f : 2^V \to \mathbb{R}_{\ge 0}$ is submodular if for any $S \subseteq T \subseteq V$ and any $x \in (V\setminus T)$, $f(\cdot)$ satisfies $f(S\cup\{x\}) - f(S) \ge f(T\cup\{x\}) - f(T)$. The reason for the non-submodularity of function $D_s(\cdot)$ is due to the *combination effect* of nodes in the blocker set. That is, to prevent a node from being activated by s, we need to simultaneously block two or more nodes. For example, reconsidering Figure 1, to protect v_3, we need to simultaneously block v_1 and v_2. Motivated by this, we disregard the combination effect to obtain a lower bound that is submodular. Specifically, we only consider nodes that can be protected by blocking only one node in G. Accordingly, given a blocker set B, the lower bound of $D_s(B)$ can be defined as:

$$D_s^L(B) = \mathbb{E}_{\Phi\sim\Omega}[|\cup_{v\in B} N_\Phi(v)|] = \sum_{\phi\in\Omega} p(\phi)\cdot|\cup_{v\in B} N_\phi(v)|, \tag{5}$$

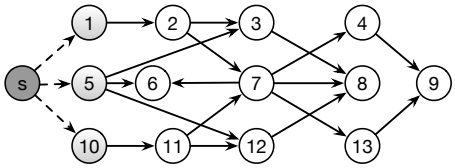


Figure 2: Example of the bounds

where $N_\phi(v)$ denotes the set of nodes whose status changes from active to inactive under ϕ after v is blocked.

Lemma 3.1. *Given a seed set S and its unified seed node s, $D_s^L(\cdot)$ is monotone nondecreasing and submodular under the IC model.*

Due to the space limitation, the detailed proof for Lemma 3.1 and other omitted proofs can be found in the full version [40]. In addition, since $D_s^L(\emptyset) = \mathbb{E}[I_G(s)]$ and computing $\mathbb{E}[I_G(s)]$ is #P-hard [8], the computation of $D_s^L(\cdot)$ is #P-hard.

3.3 Upper Bound

For each node v activated by s, v can be protected if all the paths from s to v are blocked. Intuitively, we can obtain the upper bound of the objective function by relaxing the condition for nodes to be protected. We call that v can be *alternative-protected* if there exists one path from s to v is blocked. Given a realization $\phi = (V(\phi), E(\phi))$, let $\phi^s = (V(\phi^s), E(\phi^s))$ be the graph of misinformation receivers under ϕ. Specifically, $V(\phi^s) = R_\phi(s)\setminus S$, where $R_\phi(s)$ is the set of nodes reachable from s in ϕ and $E(\phi^s) = \{\langle u, v\rangle \in E(\phi) : u \in V(\phi^s), v \in V(\phi^s)\}$. In such a setting, for any node $v \in V(\phi^s)$, v can alternative-protect those nodes that are reachable from v in ϕ^s. Given a blocker set B, the upper bound of $D_s(B)$ can be defined as:

$$D_s^U(B) = E_{\Phi\sim\Omega}[|M_\phi(B)|] = \sum_{\phi\in\Omega} p(\phi)\cdot|M_\phi(B)|, \tag{6}$$

where $M_\phi(B)$ is the set of nodes reachable from B in ϕ^s. Upper bounding function maximization is essentially the influence maximization problem [19], thus it also possesses monotonicity, submodularity and NP-hardness, and the computation of $D_S^U(\cdot)$ is #P-hard.

Example 3.1. Here we first illustrate how to transfer multiple seeds to one seed. As shown in Figure 2, the misinformation seed set $S = \{v_1, v_5, v_{10}\}$. We create a unified seed node s and three edges, $\langle s, v_1\rangle, \langle s, v_5\rangle, \langle s, v_{10}\rangle$, with the propagation probability of 1.

Then we illustrate an example of our proposed bounds. Suppose the blocker set is $B = \{v_3, v_7, v_{12}\}$. For the objective function of IMIN, $D_s(B) = |\{v_3, v_4, v_7, v_8, v_9, v_{12}, v_{13}\}| = 7$. For the lower bound, $D_s^L(B) = |\{v_3, v_4, v_7, v_9, v_{12}, v_{13}\}| = 6$. For the upper bound, $D_s^U(B) = |\{v_3, v_4, v_6, v_7, v_8, v_9, v_{12}, v_{13}\}| = 8$.

4 BOUNDING FUNCTIONS ESTIMATION

To achieve Lines 1-2 in Algorithm 1, we first need to compute $D_s^L(\cdot)$ and $D_s^U(\cdot)$. However, the computation of them is #P-hard. Additionally, the state-of-the-art technique for influence estimation (i.e., RIS [5]) cannot be applied to estimate our bounding functions. Specifically, we only need to consider the nodes that can be reached by misinformation, since only these nodes need to be protected and can contribute to the bounding functions. Besides, the number of these nodes is unknown due to the randomness of propagation.

To address this issue, in Section 4.1 and Section 4.2, we devise two sampling methods named Local Sampling and Global Sampling to estimate the lower bound and upper bound, respectively.

4.1 Lower Bounding Function Estimation

In the following, we first clarify some concepts mentioned in the sampling technique.

Definition 4.1 (Common Path (CP) Set & Common Path (CP) Sequence). Given a graph $G = (V, E)$, a seed set $S \subseteq V$, a unified source node s and a realization ϕ obtained from G, for any node $v \in (V \backslash S)$, a Common Path (CP) set of v, denoted by $C_\phi(s, v)$, is the set of common nodes on all paths from s to v in ϕ (exclude S), i.e., $C_\phi(s, v) = \{u \in (V \backslash S) : u \in \bigcap_{i=1}^{j} P_i(s, v)\}$, where $P_i(s, v)$ $(1 \leq i \leq j)$ denotes the set of nodes on a path from s to v and j is the number of paths from s to v. Let $R_\phi(s)$ be the set of nodes reachable from s in ϕ. A Common Path (CP) sequence in ϕ, denoted by C_ϕ^s, is the set of CP sets of the nodes in $R_\phi(s)$ (exclude S), i.e., $C_\phi^s = \{C_\phi(s, v) : v \in (R_\phi(s) \backslash S)\}$.

Example 4.1. Reconsider the social network G in Figure 1. Given the realization ϕ of G in Figure 1(b) and the misinformation seed node v_0, we first illustrate the CP set of node v_6 as follows. We can find that there are four paths between v_0 and v_6, i.e., $P_1(v_0, v_6) = \{v_1, v_3, v_6\}$, $P_2(v_0, v_6) = \{v_2, v_3, v_6\}$, $P_3(v_0, v_6) = \{v_1, v_3, v_5, v_6\}$ and $P_4(v_0, v_6) = \{v_2, v_3, v_5, v_6\}$. Note that, $P_i(v_0, v_6)(1 \leq i \leq 4)$ does not include v_0. The CP set of v_6 in ϕ is the common nodes on four paths, i.e., $C_\phi(v_0, v_6) = \{v_3, v_6\}$. Due to $R_\phi(v_0) = \{v_1, v_2, v_3, v_5, v_6\}$, a CP sequence in ϕ is the set of CP sets of nodes in $R_\phi(v_0)$.

In this paper, ϕ can be dropped when it is clear from the context. Given a blocker set $B \subseteq (V \backslash S)$ and a set of CP sequences $\mathbb{C}^s$, we use $Cov_{\mathbb{C}^s}(B)$ denote the coverage of B in $\mathbb{C}^s$, i.e., $Cov_{\mathbb{C}^s}(B) = \sum_{C^s \in \mathbb{C}^s} \sum_{C(s,v) \in C^s} \min\{|B \cap C(s, v)|, 1\}$. We can estimate $D_s^L(B)$ by generating a certain number of CP sequences. Lemma 4.2 shows that $\frac{Cov_{\mathbb{C}^s}(B)}{|\mathbb{C}^s|}$ is an unbiased estimator of $D_s^L(B)$.

Lemma 4.2. *Given a misinformation seed set $S \subseteq V$, a unified seed node s and the set of CP sequences $\mathbb{C}^s$, for any blocker set $B \subseteq (V \backslash S)$,*

$$D_s^L(B) = \mathbb{E}[\frac{Cov_{\mathbb{C}^s}(B)}{|\mathbb{C}^s|}], \tag{7}$$

where the expectation is taken over the random choices of $\mathbb{C}^s$.

Based on the above analysis, to accurately estimate $D_s^L(B)$, we need to generate sufficient CP sequences, which consist of numerous CP sets. A straightforward method to generate a CP set is first to find all paths between two nodes and then identify the common nodes on these paths. However, finding all paths between two nodes is time-consuming. To address this issue, we devise a scalable implementation to generate a CP set in polynomial time based on the DT (Definition 2.3 in Section 2.2). Given the source node s and a node $v \in (V \backslash S)$, the construction of the CP set of v is as follows.

- Generate a realization ϕ from G.
- Construct the DT of ϕ by Lengauer-Tarjan algorithm [23], obtain all the nodes on the path from s to v (exclude s) in DT and store them into $C_\phi(s, v)$.

We can observe that the time complexity of generating a CP set is the same as that of the Lengauer-Tarjan algorithm [23], which is $O(m \cdot \alpha(m, n))$, α is the inverse function of Ackerman's function [1]. Since DT is a tree, each node in DT has only one incoming neighbor. Based on this property, we further propose an efficient algorithm for constructing a CP sequence, whose details are shown in Algorithm 2. We first generate a random realization ϕ from G (Line 1). Then we obtain all reachable nodes of s in ϕ by applying DFS, and store them into $R_\phi(s)$ (Line 2). By applying Lengauer-Tarjan algorithm [23], the immediate dominator of each node $u \in R_\phi(s)$ is recorded as $idom[u]$ and we construct the corresponding DT roots at s of ϕ (Line 3). In Line 4, we initialize C_ϕ^s as $\emptyset$ to store the CP sequence and $M[\cdot]$ as 0. If $v \in S$, we set $M[v] = 1$ (Line 5) and we only construct the CP set for the nodes in $R_\phi(s) \backslash S$ (Line 7). Note that, we prioritize constructing the CP set for nodes with the order of DFS traversal from s (Line 6). In such a setting, for each node $u \in (R_\phi(s) \backslash S)$, the CP set of $idom[u]$ will be generated earlier than that of u since $idom[u]$ will be traversed earlier. When constructing the CP set for node u, if $M[idom[u]] = 0$ (i.e., $idom[u] \notin S$), u can inherit the CP set of $idom[u]$ (Lines 8-9). Otherwise, the CP set for u will only consist of u itself (Lines 10-11). It is clear that in Lines 6-12, we only need to process a DFS traversal starting from s. Therefore, the time complexity of Algorithm 2 is $O(m \cdot \alpha(m, n))$.

Algorithm 2: Local Sampling

```
Input  : The graph G = (V, E), the seed set S and the unified seed node s.
Output : The CP sequence C_φ^s.
1  generate a realization φ from G;
2  obtain all reachable nodes of s in φ by DFS and store them into R_φ(s);
3  record the immediate dominator of each node u ∈ R_φ(s) as idom[u] and
   construct the DT roots at s of φ;
4  C_φ^s ← ∅, M[·] ← 0;
5  for each v ∈ S do M[v] ← 1;
6  for each u ∈ R_φ(s) with the order of DFS traversal from s do
7      if M[u] = 0 then
8          if M[idom[u]] = 0 then
9              C_φ(s, u) ← C_φ(s, u) ∪ C_φ(s, idom[u]) ∪ {u};
10         else
11             C_φ(s, u) ← C_φ(s, u) ∪ {u};
12         C_φ^s ← C_φ^s ∪ C_φ(s, u);
13 return C_φ^s;
```

Example 4.2. Here is an example to illustrate the process of Local Sampling. As shown in Figure 1(a), v_0 is the misinformation seed node. A realization obtained from Figure 1(a) and the corresponding DT are shown in Figure 1(b) and Figure 1(c), respectively. Suppose the DFS order is $(v_1, v_3, v_6, v_5, v_2)$. Since the immediate dominator of v_1 and v_3 is both v_0, i.e., $idom[v_1] = idom[v_3] = v_0$, the CP set of them only consists themselves, i.e., $C_\phi(v_0, v_1) = \{v_1\}$ and $C_\phi(v_0, v_3) = \{v_3\}$. Since $idom[v_6] = idom[v_5] = v_3$, we can obtain that $C_\phi(v_0, v_6) = \{v_3, v_6\}$ and $C_\phi(v_0, v_5) = \{v_3, v_5\}$. Similarly, $C_\phi(v_0, v_2) = \{v_2\}$.

4.2 Upper Bounding Function Estimation

To estimate the upper bounding function, in the following, we first propose the concept of LRR set.

Definition 4.3 (Local Reverse Reachable (LRR) Set). Given a graph $G = (V, E)$, a seed set $S \subseteq V$, a unified seed node s, a node $v \in (V \backslash S)$ and a realization $\phi = (V(\phi), E(\phi))$, a Local Reverse Reachable (LRR) set of v, denoted by $L_\phi(v)$, is the set of nodes that can reach v in ϕ^s.

Algorithm 3: Global Sampling

Input : The graph $G = (V, E)$, the seed set S and the unified seed node s.
Output : The random LRR set $L_\phi(v)$.

1 generate a realization ϕ obtained from G and a node v is randomly selected from V'_s with the probability of $\frac{1}{|V'_s|}$;
2 **if** $v \in R_\phi(s)$ **then**
3 $\quad L_\phi(v) \leftarrow$ the set of nodes that can reach v in ϕ^s;
4 **else** $L_\phi(v) \leftarrow \emptyset$;
5 **return** $L_\phi(v)$;

Table 2: Time cost of generating 10K LRR sequences on DBLP

$\|S\|$	10	20	30	40	50
time (s)	26438.5	48907.3	67038.2	83011.5	99088.0

By generating a certain number of random LRR sets $\mathbb{L}$, we can obtain that $|V'_s| \cdot \frac{Cov_{\mathbb{L}}(B)}{|\mathbb{L}|}$ is an unbiased estimate of $D_s^U(B)$, where V'_s denotes the set of nodes that can be reached by s in G (exclude S).

Lemma 4.4. *Given a seed set $S \subseteq V$, a unified seed node s and the set of random LRR sets $\mathbb{L}$, for any blocker set $B \subseteq (V \backslash S)$,*

$$D_s^U(B) = |V'_s| \cdot \mathbb{E}[\frac{Cov_{\mathbb{L}}(B)}{|\mathbb{L}|}], \tag{8}$$

where the expectation is taken over the random choices of $\mathbb{L}$, $Cov_{\mathbb{L}}(B)$ is the coverage of B in $\mathbb{L}$,i.e., $Cov_{\mathbb{L}}(B) = \sum_{L(v)\in\mathbb{L}} \min\{|B \cap L(v)|, 1\}$.

Based on the above lemma, we propose Global Sampling to enable an accurate estimation of $D_s^U(\cdot)$. As shown in Algorithm 3, we first generate a realization ϕ and randomly select a node v from V'_s (Line 1). If the selected node v cannot be influenced by s in ϕ, which implies that there are no nodes that can alternative-protect v within this realization, we set the LRR set to an empty set (Line 4).

Discussion. Actually, the upper bounding function can be estimated in a similar manner to Local Sampling based on the concept of Local Reverse Reachable (LRR) sequence for ϕ, denoted by L^s_ϕ. L^s_ϕ is the sequence of the LRR sets of the nodes that can be reached by s in ϕ (exclude S), i.e., $L^s_\phi = \{L_\phi(v) : v \in R_\phi(s)\backslash S\}$. By generating a certain number of LRR sequences $\mathbb{L}^s$, $D_s^U(B)$ can be unbiasedly estimated via $\frac{Cov_{\mathbb{L}^s}(B)}{|\mathbb{L}^s|}$, where $Cov_{\mathbb{L}^s}(B)$ denotes the coverage of B in $\mathbb{L}^s$, i.e., $Cov_{\mathbb{L}^s}(B) = \sum_{L^s\in\mathbb{L}^s} \sum_{L(v)\in L^s} \min\{|B \cap L(v)|, 1\}$.

However, since ϕ^s is not necessarily a tree structure, we cannot generate an LRR sequence with only one single DFS traversal, as we do when generating a CP sequence. Generally, we need to conduct $|R_\phi(s)\backslash S|$ DFS traversals for each LRR sequence generation, which is rather time-consuming, especially when a substantial number of users can receive the misinformation. Thus, we propose the Global Sampling method to tackle the problem. In Table 2, we present the time cost of generating 10K LRR sequences on the DBLP network, which has more than one million edges (the dataset details can be seen in Section 6). As observed, with the increase of the number of misinformation seed nodes, the time cost of LRR sequence generation grows. In particular, when $|S| = 50$, the time required to produce the samples even exceeds one day, which is practically infeasible. In contrast, based on Global Sampling, our solution for maximizing the upper bounding function only needs 7693.55 seconds when $|S| = 50$ on DBLP, and the sample size is larger than 10K to ensure the desired approximation guarantee.

Table 3: The number of CP sequences generated by OPBM and LSBM on DBLP ($k = 100$, $|S| = 10$, $\beta = 0.1$, $\delta = 1/n$)

ϵ	0.1	0.2	0.3	0.4	0.5
OPBM	2,871,296	1,435,648	717,824	358,912	179,456
LSBM	22,544	11,272	5,636	2,818	2,818

5 BOUNDING FUNCTIONS MAXIMIZATION

Based on the proposed estimation methods in Section 4, we first design two algorithms for bounding functions maximization in Section 5.1 and Section 5.2, which aim to find a set of blockers to maximize the bounding functions with approximation guarantees. We then propose a heuristic algorithm to solve IMIN in Section 5.3.

5.1 Lower Bounding Function Maximization

In general, to get a solution for lower bounding function maximization, we need to obtain the value of $D_s^L(\cdot)$. As we stated in Section 4.1, $Cov_{\mathbb{C}^s}(\cdot)/|\mathbb{C}^s|$ is an unbiased estimator of $D_s^L(\cdot)$. That is, for estimating $D_s^L(\cdot)$ accurately, we need to generate a certain number of samples (i.e., CP sequences). However, the generation of numerous samples may lead to significant computation overhead for the algorithm. Therefore, it is imperative for us to determine an appropriate sample size, so as to strike a balance between efficiency and accuracy. A straightforward approach is to employ the state-of-the-art algorithm for influence maximization (i.e., OPIM-C [33]). Specifically, OPIM-C first generates two independent collections of samples $\mathcal{R}_1$ and $\mathcal{R}_2$ and then generates a solution using the greedy algorithm on $\mathcal{R}_1$. Afterward, OPIM-C assesses whether the solution can meet the stopping criterion. If the criterion is met, the solution is returned; otherwise, the above-mentioned steps are repeated until the algorithm terminates. In particular, the stopping criterion of OPIM-C is $\mathcal{M}(S) \geq 1 - 1/e - \epsilon$, where S is the solution in the current round and $\mathcal{M}(\cdot)$ is the function determined by two martingale-based concentration bounds [36] on $\mathcal{R}_1$ and $\mathcal{R}_2$.

However, OPIM-C cannot be directly applied for our problem. The main reason is that OPIM-C relies on RIS, which is not suitable for the estimation of bounding function as stated before. Besides, since only the nodes eligible to be reached by the misinformation seed set can contribute to the bounding function, the concentration bounds leveraged by OPIM-C are also not tailored to the unbiased estimator proposed in Section 4.1, which results in the stopping criterion of OPIM-C being infeasible for our problem. This motivates us to develop an efficient algorithm for lower bounding function maximization. In this section, we first propose two martingale-based concentration bounds, based on which, we then derive a novel stopping criterion. Combining this with Local Sampling, we propose Local Sampling based Bounding Function Maximization (LSBM). In Table 3, we report the number of CP sequences generated by LSBM and the algorithm, which employs OPIM-C for our problem directly but with the proposed unbiased estimator of $D_s^L(\cdot)$ (termed as OPBM) on DBLP. As can be seen, in most cases, the number of samples generated by OPBM is more than 100x larger than that of LSBM. In addition, on the larger datasets, OPBM cannot even finish due to the memory overflow. In the following, we first propose two novel concentration bounds, which are based on the concept of *martingale* [13].

Algorithm 4: LSBM

Input : The graph $G = (V, E)$, the seed set S, the unified seed node s, the budget k and parameter β, ϵ, δ.
Output : The blocker set B_L with $(1 - 1/e - \epsilon)$-approximation.

1 $ON \leftarrow$ the set of nodes that can be directly activated by S;
2 **if** $|ON| \leq k$ **then**
3 **return** $B_L = ON$;
4 $\hat{I}_G(s) \leftarrow$ the estimated value of $\mathbb{E}[I_G(s)]$ with $(\beta, \frac{\delta}{6})$-approximation;
5 $\mathrm{OPT}^L \leftarrow$ the lower bound of $D_s^L(B_L^o)$;
6 $\theta_{\max} \leftarrow \frac{2\hat{I}_G(s)\left((1-1/e)\sqrt{\ln\frac{12}{\delta}}+\sqrt{(1-1/e)(\ln\binom{n-|S|}{k}+\ln\frac{12}{\delta})}\right)^2}{(1-\beta)\epsilon^2\mathrm{OPT}^L}$;
7 $\theta_0 \leftarrow \theta_{\max} \cdot (1-\beta)\epsilon^2\mathrm{OPT}^L/\hat{I}_G(s)$;
8 $i_{\max} \leftarrow \lceil \log_2 \frac{\theta_{\max}}{\theta_0} \rceil$;
9 generate two sets $\mathbb{C}_1^s, \mathbb{C}_2^s$ of θ_0 random CP sequences, respectively;
10 $a_1 \leftarrow \ln \frac{3i_{max}}{\delta}, a_2 \leftarrow \ln \frac{3i_{max}}{\delta}$;
11 **for** $i \leftarrow 1$ *to* $i_{\max}$ **do**
12 $B_L \leftarrow$ Max-Coverage$(\mathbb{C}_1^s, k)$;
13 $\sigma^L(B_L) \leftarrow 0$;
14 **if** $Cov_{\mathbb{C}_2^s}(B_L) \cdot (1-\beta)/\hat{I}_G(s) \geq 5a_1/18$ **then**
15 $\sigma^L(B_L) \leftarrow \left((\sqrt{\frac{Cov_{\mathbb{C}_2^s}(B_L)\cdot(1-\beta)}{\hat{I}_G(s)} + \frac{2a_1}{9}} - \sqrt{\frac{a_1}{2}})^2 - \frac{a_1}{18}\right) \cdot \frac{1}{|\mathbb{C}_2^s|}$;
16 **else if** $Cov_{\mathbb{C}_2^s}(B_L) \cdot (1+\beta)/\hat{I}_G(s) \leq 5a_1/18$ **then**
17 $\sigma^L(B_L) \leftarrow \left((\sqrt{\frac{Cov_{\mathbb{C}_2^s}(B_L)\cdot(1+\beta)}{\hat{I}_G(s)} + \frac{2a_1}{9}} - \sqrt{\frac{a_1}{2}})^2 - \frac{a_1}{18}\right) \cdot \frac{1}{|\mathbb{C}_2^s|}$;
18 $Cov^u_{\mathbb{C}_1^s}(B_L^o) \leftarrow \min_{0\leq i\leq k}\left(Cov_{\mathbb{C}_1^s}(B_i) + \sum_{v\in maxMC(B_i,k)} Cov_{\mathbb{C}_1^s}(v \mid B_i)\right)$;
19 $\sigma^U(B_L^o) \leftarrow \left(\sqrt{\frac{Cov^u_{\mathbb{C}_1^s}(B_L^o)\cdot(1+\beta)}{\hat{I}_G(s)} + \frac{a_2}{2}} + \sqrt{\frac{a_2}{2}}\right)^2 \cdot \frac{1}{|\mathbb{C}_1^s|}$;
20 **if** $\sigma^L(B_L)/\sigma^U(B_L^o) \geq 1 - 1/e - \epsilon$ ***or*** $i = i_{\max}$ **then**
21 **return** B_L;
22 double the sizes of $\mathbb{C}_1^s$ and $\mathbb{C}_2^s$ with new CP sequences;
23 **Procedure** Max-Coverage$(\mathbb{C}^s, k)$;
24 $B \leftarrow \emptyset$;
25 **for** $i \leftarrow 1$ *to* k **do**
26 $u \leftarrow \arg\max_{v\in(V\setminus S)}(Cov_{\mathbb{C}^s}(B\cup\{v\}) - Cov_{\mathbb{C}^s}(B))$;
27 $B \leftarrow B \cup \{u\}$;
28 **end for**
29 **return** B;

Lemma 5.1 (Concentration Bounds). *Given a blocker set B, a seed node s and a set of θ random CP sequences $\mathbb{C}^s$. For any $\lambda > 0$,*

$$\Pr\left[\frac{Cov_{\mathbb{C}^s}(B)}{\mathbb{E}[I_G(s)]} - \frac{D_s^L(B)\cdot\theta}{\mathbb{E}[I_G(s)]} \geq \lambda\right] \leq \exp\left(-\frac{\lambda^2}{\frac{2D_s^L(B)}{\mathbb{E}[I_G(s)]}\cdot\theta + \frac{2}{3}\lambda}\right), \quad (9)$$

$$\Pr\left[\frac{Cov_{\mathbb{C}^s}(B)}{\mathbb{E}[I_G(s)]} - \frac{D_s^L(B)\cdot\theta}{\mathbb{E}[I_G(s)]} \leq -\lambda\right] \leq \exp\left(-\frac{\lambda^2}{\frac{2D_s^L(B)}{\mathbb{E}[I_G(s)]}\cdot\theta}\right). \quad (10)$$

LSBM algorithm. Based on these concentration bounds, we devise a scalable implementation called LSBM for lower bounding function maximization. The pseudocode of LSBM is shown in Algorithm 4. Let ON be the set of outgoing neighbors of S (Line 1). When the budget k is no less than the number of nodes in ON, LSBM directly returns ON as the blocker set (Lines 2-3). Then we calculate $\hat{I}_G(s)$ with $(\beta, \frac{\delta}{6})$-approximation by employing the generalized stopping rule algorithm introduced in [47]. In addition, we derive the lower bound of $D_s^L(B_L^o)$ and define the constants $\theta_{\max}$ and θ_0 (Lines 4-7). Afterwards, we generate two sets of CP sequences $\mathbb{C}_1^s$ and $\mathbb{C}_2^s$, each of size θ_0 (Line 9). The subsequent part of the algorithm consists of at most i_{max} iterations. In each iteration, we first invoke Procedure Max-Coverage to get a blocker set B_L, i.e., finding a set of k nodes such that B_L intersects with as many CP sequences as possible in $\mathbb{C}_1^s$ (Line 12). Then we derive $\sigma^L(B_L)$ and $\sigma^U(B_L^o)$ from $\mathbb{C}_2^s$ and $\mathbb{C}_1^s$, respectively (Lines 14-19). Specifically, $\sigma^L(B_L)$ is the lower bound of $D_s^L(B_L)/\mathbb{E}[I_G(s)]$ and $\sigma^U(B_L^o)$ is the upper bound of $D_s^L(B_L^o)/\mathbb{E}[I_G(s)]$. If $\sigma^L(B_L)/\sigma^U(B_L^o) \geq 1 - 1/e - \epsilon$ or $i = i_{\max}$, LSBM returns B_L and terminates. Otherwise, the quantities of CP sequences in $\mathbb{C}_1^s$ and $\mathbb{C}_2^s$ will be doubled, and LSBM will proceed to the next iteration (Lines 20-22). Next, we will explain in detail how LSBM can return a solution with an approximation guarantee. Note that, when deriving $\theta_{\max}$, $\sigma^L(B_L)$ and $\sigma^U(B_L^o)$, it is imperative to carefully consider the error associated with estimating $\mathbb{E}[I_G(s)]$, to ensure the desired approximation guarantee.

Deriving $\theta_{\max}$. Based on previous concentration bounds [36], Tang et al. derive an upper bound on the sample size required to ensure $(1 - 1/e - \epsilon)$-approximation holds with probability at least $1 - \delta$ for IM. Similarly, we derive the corresponding upper bound on the sample size for the IMIN problem based on our proposed novel concentration bounds. The following lemma provides the setting of θ_{max}, ensuring the correctness of LSBM when $i = i_{\max}$.

Lemma 5.2. *Let $\mathbb{C}^s$ be a set of random CP sequences, B_L be a size-k blocker set generated by applying Max-Coverage on $\mathbb{C}^s$, B_L^o be the optimal solution with size-k, OPT^L be the lower bound of $D_s^L(B_L^o)$ and $\hat{I}_G(s)$ be the estimated value of $\mathbb{E}[I_G(s)]$ with $(\beta, \frac{\delta}{6})$-approximation. For fixed β, ϵ and δ, let*

$$\theta_{max} = \frac{2\hat{I}_G(s)\left((1-1/e)\sqrt{\ln\frac{12}{\delta}} + \sqrt{(1-1/e)(\ln\binom{n-|S|}{k} + \ln\frac{12}{\delta})}\right)^2}{(1-\beta)\epsilon^2 OPT^L},$$

if $|\mathbb{C}^s| = \theta \geq \theta_{\max}$, then B_L is $(1 - 1/e - \epsilon)$-approximate solution with at least $1 - \delta/3$ probability.

Deriving OPT^L. Due to the different properties of the objective functions of IMIN and IM, we cannot directly replace OPT^L with k as OPIM-C [33]. To address this issue, we set $\mathrm{OPT}^L = \sum_{v\in B^*}\Pr[s \to v]$, where B^* denote the set of k nodes of ON with the k largest probability of being activated by s and $\Pr[s \to v]$ is the probability that s can activate v. Since when we select a node v as a blocker, $D_s^L(\{v\} \cup B) - D_s^L(B) \geq \Pr[s \to v]$. The reason why we set ON as the candidate set is that the value of $\Pr[s \to v]$ of each node $v \in ON$ can be computed efficiently, otherwise it will become the bottleneck of the whole algorithm.

Deriving $\sigma^L(B_L)$ and $\sigma^U(B_L^o)$. Next, we derive the lower bound $\sigma^L(B_L)$ of $\frac{D_s^L(B_L)}{\mathbb{E}[I_G(s)]}$ and the upper bound $\sigma^U(B_L^o)$ of $\frac{D_s^L(B_L^o)}{\mathbb{E}[I_G(s)]}$ such that the approximation ratio $\frac{D_s^L(B_L)}{D_s^L(B_L^o)} \geq \frac{\sigma^L(B_L)}{\sigma^U(B_L^o)}$.

Lemma 5.3. *For any $0 \leq \beta, \epsilon, \delta \leq 1$, we have*

$$\Pr\left[\sigma^L(B_L) \leq \frac{D_s^L(B_L)}{\mathbb{E}[I_G(s)]}\right] \geq 1 - \frac{\delta}{3i_{\max}},$$

$$\Pr\left[\sigma^U(B_L^o) \geq \frac{D_s^L(B_L^o)}{\mathbb{E}[I_G(s)]}\right] \geq 1 - \frac{\delta}{3i_{\max}}.$$

Putting together. The reason that LSBM ensures $(1-1/e-\epsilon)$-approximation with at least $1-\delta$ probability can be explained as follows. First, the algorithm has at most $i_{\max}$ iterations. In each of the first $i_{\max}-1$ iterations, a blocker set B_L is generated and we derive an approximation guarantee $\sigma^L(B_L)/\sigma^U(B_L^o)$ that is incorrect with at most $2\delta/(3i_{\max})$ probability (Lemma 5.3). By the union bound, LSBM has at most $2\delta/3$ to return an incorrect solution in the first $i_{\max}-1$ iterations. Meanwhile, in the last iteration, a blocker set B_L obtained by applying Procedure Max-Coverage on $\mathbb{C}_1^s$, with $|\mathbb{C}_1^s| \geq \theta_{\max}$. This ensures that B_L is an $(1-1/e-\epsilon)$-approximation with at least $1-\delta/3$ probability when $i = i_{\max}$ (Lemma 5.2). Therefore, the probability that LSBM returns an incorrect solution in any iteration is at most δ, leading to the following theorem.

Theorem 5.4. *Given $0 \leq \beta, \epsilon, \delta \leq 1$, B_L^o is the optimal solution of lower bounding function maximization, LSBM returns B_L satisfies:*

$$\Pr[D_s^L(B_L) \geq (1-1/e-\epsilon)D_s^L(B_L^o)] \geq 1-\delta. \tag{11}$$

Moreover, we have the following theorem to guarantee the expected time complexity of LSBM.

Theorem 5.5. *When $\frac{2\beta}{1+\beta} \leq \epsilon$ and $\delta \leq 1/2$, LSBM runs in $O(\frac{(k\ln(n-|S|)+\ln 1/\delta)\mathbb{E}[I_G(s)](m\cdot\alpha(m,n)+D_s^L(B_L^o))}{(\epsilon+\epsilon\beta-2\beta)^2 D_s^L(B_L^o)} + \frac{m\cdot\ln 1/\delta}{\beta^2})$ expected time under the IC model.*

5.2 Upper Bounding Function Maximization

In what follows, we propose Global Sampling based Bounding Function Maximization (GSBM) for upper bounding function maximization. GSBM is similar to the framework of LSBM and the differences between them are that we use Global Sampling to estimate $D_s^U(\cdot)$ with a certain number of random LRR sets and we set:

$$\theta_{\max} = \frac{2|V_s'|\left((1-1/e)\sqrt{\ln\frac{6}{\delta}} + \sqrt{(1-1/e)(\ln\binom{|V_s'|-|S|}{k} + \ln\frac{6}{\delta})}\right)^2}{\epsilon^2 \mathrm{OPT}^L},$$

$$\theta_0 = \theta_{\max}\cdot\epsilon^2\mathrm{OPT}^L/|V_s'|,$$

$$\sigma^L(B_U) = \left((\sqrt{Cov_{\mathbb{L}_2}(B_U) + \frac{2a_1}{9}} - \sqrt{\frac{a_1}{2}})^2 - \frac{a_1}{18}\right)\cdot\frac{|V_s'|}{|\mathbb{L}_2|},$$

$$\sigma^U(B_U^o) \leftarrow \left(\sqrt{Cov_{\mathbb{L}_1}^u(B_U^o) + \frac{a_2}{2}} + \sqrt{\frac{a_2}{2}}\right)^2\cdot\frac{|V_s'|}{|\mathbb{L}_1|}.$$

Similar to the proof of Theorem 5.4, we also show GSBM can return a solution with $(1-1/e-\epsilon)$-approximation.

Theorem 5.6. *Given $0 \leq \epsilon, \delta \leq 1$, B_U^o is the optimal solution of upper bounding function maximization, GSBM returns B_U satisfies:*

$$\Pr[D_s^U(B_U) \geq (1-1/e-\epsilon)D_s^U(B_U^o)] \geq 1-\delta. \tag{12}$$

Besides, the time complexity of GSBM is shown in Theorem 5.7.

Theorem 5.7. *When $\delta \leq 1/2$, the time complexity of GSBM under the IC model is $O(\frac{(k\ln(n-|S|)+\ln(1/\delta))(|V_s'|+m)}{\epsilon^2})$.*

5.3 A Lightweight Heuristic for IMIN

Here we consider the filling of the Sandwich, i.e., the solution for the original problem IMIN. To address this problem, we propose a Lightweight Heuristic algorithm that adopts the greedy framework

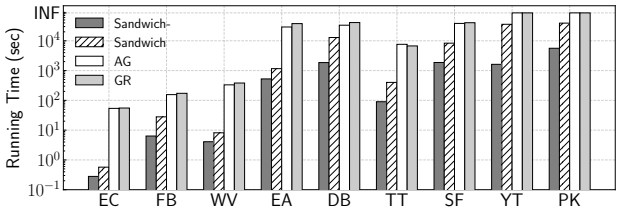


Figure 3: Time cost on all the datasets

(LHGA) combining the following proposed function to evaluate the quality score of each node:

$$s(v) = \Pr[s \to v]\cdot deg[v]. \tag{13}$$

Specifically, we iteratively select the node from ON with the largest quality score $s(\cdot)$. The motivations of our proposed function are that *i*) the outgoing neighbors of the misinformation seeds are more likely to be blockers; *ii*) the nodes with a relatively high probability of being activated by s are more likely to be blockers; *iii*) the nodes with large influence are more likely to be blockers. If the budget is no less than the number of nodes in ON, we directly return ON as the blocker set. Although LHGA does not make any theoretical contribution to SandIMIN as shown in Eq. (3), it can enhance the effectiveness of our algorithm without sacrificing efficiency. More details can be seen in Section 6.

Summary. Based on the above results, i.e., B_L returned by LSBM, B_U returned by GSBM and B_R returned by LHGA, SandIMIN returns the blocker set $B^* \in \{B_L, B_U, B_R\}$ with the smallest (γ,δ)-estimated value of $\mathbb{E}[I_{G[V\setminus B^*]}(s)]$. According to the Theorem 5.4 and 5.6, the constant α_1 and α_2 in Eq. (3) are both set to $(1-1/e-\epsilon)$, by union bound, the blocker set B produced by SandIMIN has the following theoretical guarantees with at least $1-3\delta$ probability,

$$D_s(B) \geq \max\left\{\frac{D_s(B_U)}{D_s^U(B_U)}, \frac{D_s^L(B_L^o)}{D_s(B^o)}\right\}(1-\tfrac{1}{e}-\epsilon)\tfrac{1-\gamma}{1+\gamma}D_s(B^o). \tag{14}$$

6 EXPERIMENTS

In this section, we conduct extensive experiments on 9 real-world datasets to evaluate the performance of our algorithms.

Algorithms. In the experiment, we implement and evaluate the following algorithms. *i*) **AG/GR**: the state-of-the-art algorithms proposed in [46]. *ii*) **SandIMIN**: the Sandwich framework based approach proposed in this paper with tight theoretical guarantee. Note that, both AG and GR are heuristic solutions without theoretical guarantees about the final result. Therefore, in the experiment, we also implement *iii*) **SandIMIN-**, which relaxes the theoretical result of SandIMIN by setting $\alpha_1 = 1-1/e-\epsilon$, $\alpha_2 = 0$ in Algorithm 1, i.e., without GSBM. It can provide better performance in efficiency and competitive quality of results.

Datasets. We use 9 real datasets which are available on SNAP[1] in our experiments, i.e., EmailCore, Facebook, Wiki-Vote, EmailAll, DBLP, Twitter, Stanford, Youtube and Pokec. Due to the limited space, the details of the datasets can be found in the full version [40].

Parameter settings. Following the convention [16, 30, 33, 36, 37], we set the propagation probability $p(u,v)$ of each edge $\langle u,v\rangle$ as

[1] http://snap.stanford.edu

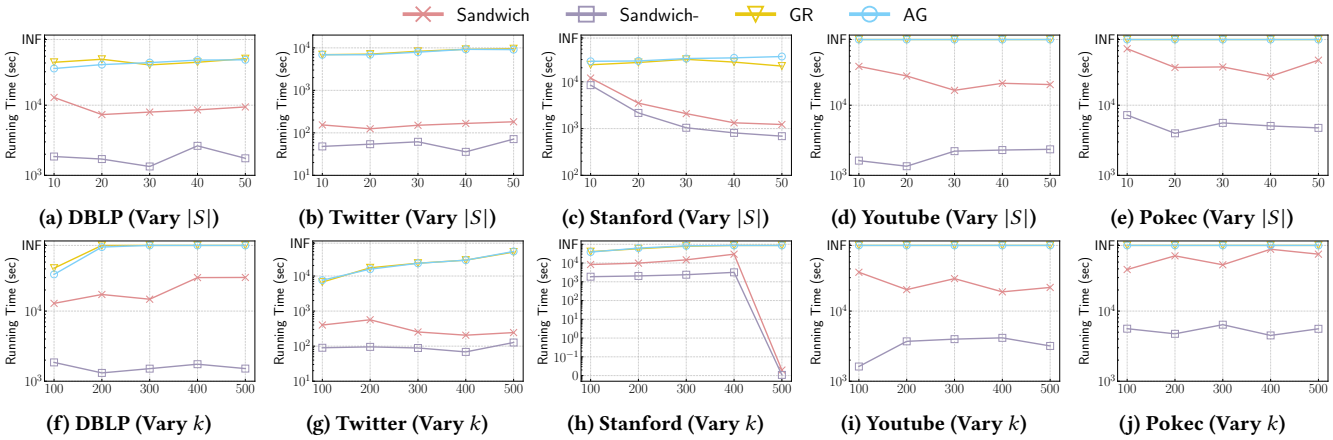


Figure 4: Efficiency evaluation by varying $|S|$ and k

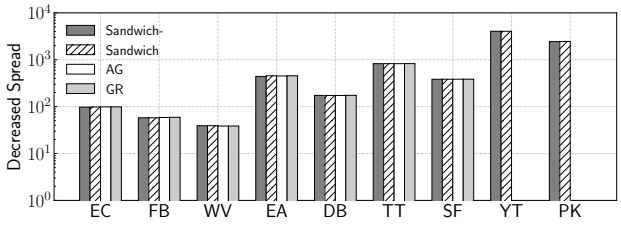


Figure 5: Decreased spread on all the datasets

the inverse of v's in-degree in IC model. By default, we set $\epsilon = 0.2$, $\beta = \gamma = 0.1$ and $\delta = 1/n$ for SandIMIN and SandIMIN-. For AG and GR, we set the number of generated realizations to 10^4, which is recommended in [46]. $|S|$ and k are 10 and 100 by default, respectively. In addition, the nodes in misinformation seed set are randomly selected from the top 200 most influential nodes. Finally, we estimate the expected spread of the seed set by taking the average of its spreads over 10^5 Monte-Carlo simulations. For each parameter setting, we repeat each algorithm 10 times and report the average value. For those experiments that cannot finish within 24 hours, we set them as **INF**. All the programs are implemented in C++ and performed on a PC with an Intel Xeon 2.10GHz CPU and 512GB memory.

6.1 Efficiency Evaluation

Results on all the datasets. In Figure 3, we first evaluate the time cost on all the datasets with the default settings. As can be seen, SandIMIN and SandIMIN- cost less time than AG and GR on all the datasets and they can achieve up to two orders of magnitude speedup. In most cases, AG is more efficient than GR. Besides, AG and GR cannot complete on large datasets (i.e., Youtube and Pokec) in a reasonable time. The primary reasons are that *i*) our methods transform the original non-submodular maximization problem into the submodular maximization scenario. This allows us to avoid generating new samples after selecting each blocker. Furthermore, our heuristic algorithm for IMIN incurs almost no computation overhead. *ii*) Building upon the novel martingale-based concentration bounds, the sample size of our methods can be significantly reduced. As shown in Table 3, on DBLP with $\epsilon = 0.3$, the sample size generated by LSBM is only 5636, even smaller than the sample size produced by AG and GR in each round. In addition, SandIMIN- is more efficient than SandIMIN on all the datasets, which is not surprising given that SandIMIN- does not execute GSBM. Besides, as the dataset becomes large, the gap between the time cost of our solutions and that of AG and GR becomes smaller. This is because, to provide theoretical guarantees for our solutions, the number of samples generated will increase as the dataset becomes large, while AG and GR use a fixed number of realizations each round for graphs of any size and offer no approximation guarantee for the quality of the returned results.

Varying $|S|$ and k. Figures 4(a)-4(e) report the response time by varying $|S|$ on the largest five datasets. As shown, SandIMIN and SandIMIN- always run faster than AG and GR on all five datasets. SandIMIN- is faster than SandIMIN due to the relaxed requirements. Generally, SandIMIN- can achieve at least an order of magnitude speedup compared with AG and GR on all five datasets under all settings except Stanford. In particular, AG and GR cannot complete within a day on Youtube and Pokec, while SandIMIN- takes only a few thousand seconds to complete. In addition, with $|S| = 40$ on Twitter, the response time of SandIMIN- (resp. SandIMIN) is 35.7 (resp. 166.7) seconds while AG (resp. GR) needs 9228.6 (resp. 9240.6) seconds to complete. This is because AG and GR need to regenerate a large number of realizations in each iteration. On Twitter and Stanford, the gap of SandIMIN and SandIMIN- is smaller than that on the other three datasets. The reason is that the large number of nodes in these three datasets may result in a relatively small value for $\mathbb{E}[I_G(s)]/|V'_s|$. Under such circumstances, the Global Sampling is more prone to generating empty LRR sets, which do not contribute to the blocker set construction, and increase computation overhead. Figures 4(f)-4(j) present the response time by varying k, where similar trends can be observed. In addition, AG and GR cannot finish within one day when k becomes larger on all five datasets except Twitter and it is seen that SandIMIN and SandIMIN- are orders of magnitude faster than AG and GR on Twitter dataset. Note that, the response time for our algorithms may either increase or decrease by increasing $|S|$ and k. This is because the time cost of

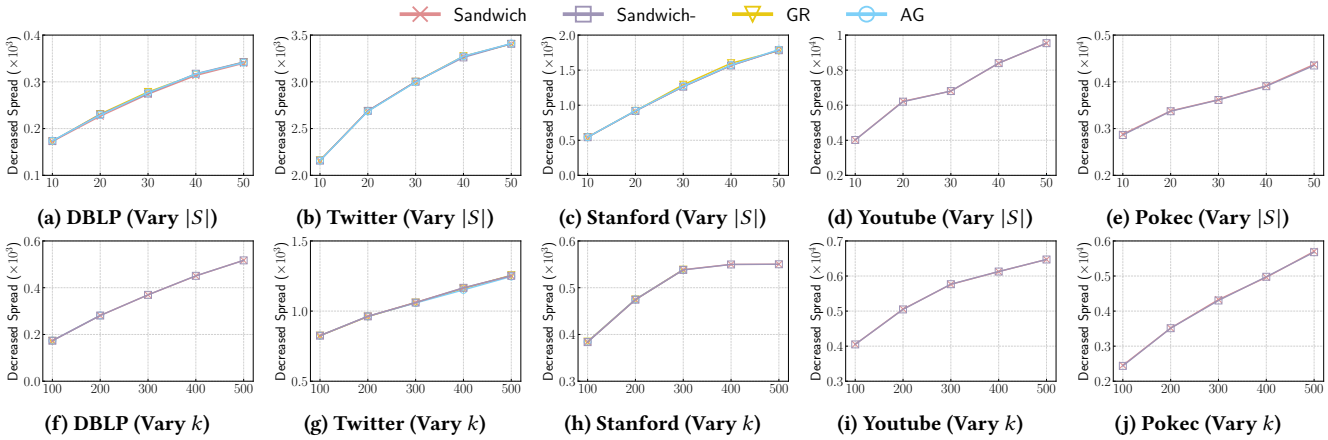


Figure 6: Effectiveness evaluation by varying $|S|$ and k

Table 4: Decreased spread of GSBM, LSBM and LHGA by varying k

	EmailCore			EmailAll			DBLP			Stanford			Youtube			Pokec		
k	GSBM	LSBM	LHGA	GSBM	LSBM	LHGA	GSBM	LSBM	LHGA	GSBM	LSBM	LHGA	GSBM	LSBM	LHGA	GSBM	LSBM	LHGA
$k = 10$	26.726	31.292	**33.140**	**313.66**	307.643	153.96	49.160	57.260	**59.350**	6921.6	**7123.6**	217.35	**1150.5**	705.20	980.40	785.90	656.50	**1000.2**
$k = 20$	**55.870**	52.544	49.614	405.77	**455.15**	240.40	65.790	67.150	**74.980**	7631.3	**8108.0**	689.32	**1962.6**	1795.9	1295.1	**1506.9**	1302.4	1111.6
$k = 30$	66.408	70.629	**72.460**	538.61	**582.23**	332.03	93.000	**98.620**	80.850	8319.7	**8991.2**	940.73	2246.9	2284.5	**2310.5**	**1922.9**	1591.2	1545.3
$k = 40$	81.816	**83.650**	80.379	588.76	**643.87**	455.00	114.45	**116.67**	92.290	8699.7	**9688.3**	1302.6	**2743.6**	2694.1	2406.6	2071.9	**2193.1**	1580.2
$k = 50$	90.907	**102.50**	97.018	664.88	**714.80**	505.38	128.21	136.95	**143.41**	9169.3	**10168**	1991.2	3046.4	**3271.5**	2551.7	**2734.1**	2590.6	2179

LSBM and GSBM mainly depends on when the stopping condition is reached, as we stated in Section 5. The larger $|S|$ (k) may make it easier or harder to reach the stopping condition. Moreover, observe that on Stanford with $k = 500$, SandIMIN and SandIMIN- only take very short time to finish. The reason lies in that our proposed algorithms can return the outgoing neighbors of S as the blocker set directly when k is large enough.

6.2 Effectiveness Evaluation

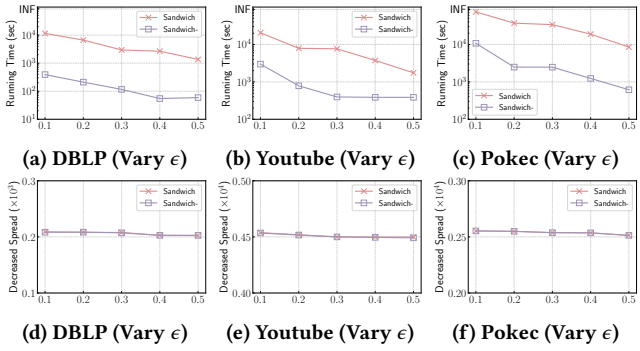
Results on all the datasets. In Figure 5, we demonstrate the effectiveness of the proposed techniques on all the datasets with the default settings. The results indicate that our solutions exhibit similar performance to AG and GR in terms of decreased spread on all the datasets. Note that, on large datasets, i.e., Youtube and Pokec, AG and GR cannot finish in a reasonable time. Therefore, the corresponding value is not shown in the figure.

Varying $|S|$ and k. Figure 6 shows the decreased spread by varying the parameters $|S|$ and k on the largest five datasets. It can be observed that SandIMIN achieves the similar decreased spread to AG and GR under different parameter settings, which reflects the effectiveness of our proposed method. In addition, the performance of SandIMIN and SandIMIN- is also very close. This validates that our proposed lower bound is very tight and close to the objective function, and the relaxation in theoretical parameters does not affect much on the real performance. For all the algorithms, the decreased spread grows with the increase of $|S|$, since more nodes could be protected. Similarly, the decreased spread increases when k becomes larger, since more blockers are selected.

Effectiveness evaluation of LSBM, GSBM and LHGA. In Table 4, we report the decreased spread of LSBM, GSBM, LHGA by varying k on six datasets with different scales. Recall that, SandIMIN returns the best solution regarding the effectiveness among the results obtained by its three components. As can be seen, within SandIMIN, each sub-algorithm possesses the potential to outperform others in achieving the largest decrease in spread. This demonstrates the effectiveness of the three algorithms. In most cases, it can be seen that LSBM performs the best in terms of effectiveness, indicating that the scenario where multiple blockers jointly protect a node is not very common. Therefore, the proposed lower bound is relatively tight w.r.t. the objective function. Additionally, our trivial heuristic method, LHGA, exhibits the best decrease in spread in a few cases. That is, if we remove LHGA from SandIMIN, its empirical accuracy would decrease in such cases. It is worth noting that LHGA almost incurs no time overhead. Therefore, LHGA can enhance the effectiveness of our algorithms without sacrificing efficiency.

6.3 Sensitivity Evaluation by Varying ϵ

Figures 7(a)-7(c) show the response time by varying ϵ on DBLP, Youtube and Pokec. As observed, the time cost of SandIMIN and SandIMIN- is reduced with the increase of ϵ, since a larger ϵ leads to smaller sample size. Figures 7(d)-7(e) present the corresponding decreased spread by varying ϵ. It can be seen that with the increase of ϵ, the decreased spread of SandIMIN and SandIMIN- slightly drops due to the smaller sample size. For example, on Pokec, the decreased spread of SandIMIN (resp. SandIMIN-) is 2553.3 (resp. 2553.3) with $\epsilon = 0.1$, and the reduction in spread of SandIMIN (resp. SandIMIN-) is 2513.6 (resp. 2511.2) with $\epsilon = 0.5$.



(a) DBLP (Vary ϵ) (b) Youtube (Vary ϵ) (c) Pokec (Vary ϵ)

(d) DBLP (Vary ϵ) (e) Youtube (Vary ϵ) (f) Pokec (Vary ϵ)

Figure 7: Efficiency and effectiveness evaluation by varying ϵ

6.4 Approximation Quality Evaluation

The approximation guarantee of SandIMIN can be seen in Eq. (14). Obviously, the exact approximation ratio is intractable to compute, as B^o and B_L^o are unknown. According to [45], $\frac{(1-\gamma)^2}{(1+\gamma)^2} \cdot (1 - 1/e - \epsilon) \cdot \frac{\hat{D}_s(B_U)}{\hat{D}_s^U(B_U)}$ is a computable lower bound of the approximation ratio for SandIMIN. Note that, no computable approximation ratio is provided for SandIMIN-, since it does not return B_U. The average lower bound of the approximation ratio (i.e., empirical approximation ratio) of SandIMIN on all the datasets (averaged over $k = 10, 50$ and 100) is shown in Figure 8. We report the results under two different settings of ϵ and γ. In particular, on all the datasets, the empirical approximation ratio exceeds 20% with $\epsilon = 0.2, \gamma = 0.1$, and exceeds 30% with $\epsilon = 0.1, \gamma = 0.05$. As a data-dependent approximation guaranteed algorithm, SandIMIN performs well in terms of approximation, as the ratios closely approximate the value of $1 - 1/e - \epsilon$ ($\approx 53.2\%$ when $\epsilon = 0.1$).

7 RELATED WORK

Influence Maximization. The Influence Maximization (IM) problem, which aims to find a set of users with the largest expected spread, is a fundamental problem in graph analysis. Kempe et al. [19] first formulate IM problem and propose independent cascade (IC) as well as linear threshold (LT) models. In addition, they utilize a greedy algorithm that returns $(1 - 1/e - \epsilon)$-approximate solution. Afterwards, a large number of work [8–12, 14, 18, 31, 34, 35, 44] focuses on the development of heuristic algorithms to reduce the computation overhead. However, such solutions return the results without theoretical guarantees. To address this issue, Brogs et al. [5] propose the *Reverse Influence Sampling* (RIS) technique, which reduces the time complexity to almost linear to the graph size. Subsequently, many RIS-based algorithms [16, 30, 33, 36, 37] that ensure $(1 - 1/e - \epsilon)$-approximations with reduced computation overhead are proposed. In addition, the variants of IM have also been extensively studied, such as considering time aspect [21, 25] and location aspect [24, 42, 43]. In [27], Lu et al. propose a Sandwich approximation strategy to solve non-submodular competitive and complementary IM. Then Wang et al. [45] extend Sandwich to the case where the objective function is intractable. Huang et al. [17] study influence maximization in closed social networks, which is non-submodular. They resort to the influence lower bounds, which are computed with the Restricted Maximum Probability Path (RMPP) model [7], to preserve submodularity. Recently, Hu et al. [15] study the triangular stability maximization problem, which is also non-submodular. They propose the Joint Baking Algorithmic Framework with theoretical guarantees to solve this problem.

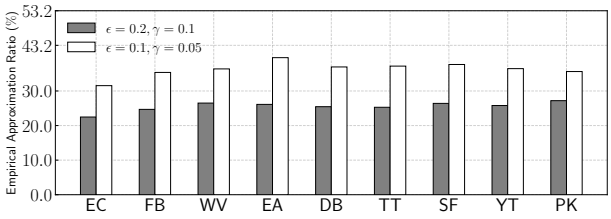


Figure 8: Empirical approximation ratio on all the datasets

Influence Minimization. As an important variant of IM, Influence minimization (IMIN) has attracted great attention due to its wide applications [3, 28]. Generally, the existing solutions for IMIN can be divided into three categories: positive information spreading, edge blocking and node blocking. Budak et al. [6] first propose to spread positive information to achieve IMIN under the IC model. Under this strategy, the objective is shown to be monotonic and submodular. Based on these properties, Tong et al. [38, 39] later design the sampling based methods and present the algorithms that provide $(1 - 1/e - \epsilon)$-approximations. Simpson et al. [32] study a time-sensitive variant of IMIN via spreading positive information. In [22] and [20], IMIN via edge blocking is investigated under the LT and IC model, respectively. Wang et al. [41] first study IMIN through node blocking under the IC model. Recently, Xie et al. [46] propose a novel approach based on dominator trees that can effectively estimate the decrease in influence of misinformation after blocking a specific node. However, the above solutions are all heuristic. Prior to our work, there is still no approximation algorithm with theoretical guarantees for IMIN via node blocking.

8 CONCLUSION

In this paper, we study the influence minimization problem via node blocking. To our best knowledge, we are the first to propose algorithms with approximation guarantees for the problem. Based on the Sandwich framework, we develop submodular and monotonic lower and upper bounds of the objective function and propose two sampling based methods to estimate the value of bounds. Besides, we design novel martingale-based concentration bounds and devise two non-trivial algorithms that provide $(1 - 1/e - \epsilon)$-approximate solutions to maximize our proposed bounding functions. We also present a lightweight heuristic for IMIN. Finally, our algorithm, SandIMIN, returns the best blocker set among these three solutions and yields a data-dependent approximation guarantee to the IMIN objective. Extensive experiments over 9 real-world datasets demonstrate the effectiveness and efficiency of our proposed approaches.

ACKNOWLEDGMENTS

This work is partially supported by ARC DP230101445, ARC DP240101322, ARC FT200100787, ARC DP230101445, ARC FT210100303, NSFC U2241211, NSFC U20B2046, and 23H020101910. Yanping Wu and Xiaoyang Wang are the corresponding authors.

REFERENCES

[1] Wilhelm Ackermann. 1928. Zum hilbertschen aufbau der reellen zahlen. *Math. Ann.* 99, 1 (1928), 118–133.

[2] Alfred V. Aho and Jeffrey D. Ullman. 1972. *The theory of parsing, translation, and compiling. 1: Parsing.* Prentice-Hall.

[3] Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of economic perspectives* (2017).

[4] Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. 2017. Guarantees for Greedy Maximization of Non-submodular Functions with Applications. In *ICML (Proceedings of Machine Learning Research)*, Vol. 70. PMLR, 498–507.

[5] Christian Borgs, Michael Brautbar, Jennifer T. Chayes, and Brendan Lucier. 2014. Maximizing Social Influence in Nearly Optimal Time. In *SODA*. SIAM, 946–957.

[6] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. 2011. Limiting the spread of misinformation in social networks. In *WWW*. ACM, 665–674.

[7] Vineet Chaoji, Sayan Ranu, Rajeev Rastogi, and Rushi Bhatt. 2012. Recommendations to boost content spread in social networks. In *WWW*. ACM, 529–538.

[8] Wei Chen, Chi Wang, and Yajun Wang. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*. ACM, 1029–1038.

[9] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. In *KDD*. ACM, 199–208.

[10] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In *ICDM*. IEEE Computer Society, 88–97.

[11] Suqi Cheng, Huawei Shen, Junming Huang, Wei Chen, and Xueqi Cheng. 2014. IMRank: influence maximization via finding self-consistent ranking. In *SIGIR*. ACM, 475–484.

[12] Suqi Cheng, Huawei Shen, Junming Huang, Guoqing Zhang, and Xueqi Cheng. 2013. StaticGreedy: solving the scalability-accuracy dilemma in influence maximization. In *CIKM*. ACM, 509–518.

[13] Fan R. K. Chung and Lincoln Lu. 2006. Survey: Concentration Inequalities and Martingale Inequalities: A Survey. *Internet Math.* 3, 1 (2006), 79–127.

[14] Amit Goyal, Wei Lu, and Laks V. S. Lakshmanan. 2011. SIMPATH: An Efficient Algorithm for Influence Maximization under the Linear Threshold Model. In *ICDM*. IEEE Computer Society, 211–220.

[15] Zheng Hu, Weiguo Zheng, and Xiang Lian. 2023. Triangular Stability Maximization by Influence Spread over Social Networks. *Proceedings of the VLDB Endowment* 16, 11 (2023), 2818–2831.

[16] Keke Huang, Sibo Wang, Glenn S. Bevilacqua, Xiaokui Xiao, and Laks V. S. Lakshmanan. 2017. Revisiting the Stop-and-Stare Algorithms for Influence Maximization. *Proc. VLDB Endow.* 10, 9 (2017), 913–924.

[17] Shixun Huang, Wenqing Lin, Zhifeng Bao, and Jiachen Sun. 2022. Influence Maximization in Real-World Closed Social Networks. *Proc. VLDB Endow.* 16, 2 (2022), 180–192.

[18] Kyomin Jung, Wooram Heo, and Wei Chen. 2012. IRIE: Scalable and Robust Influence Maximization in Social Networks. In *ICDM*. IEEE Computer Society, 918–923.

[19] David Kempe, Jon M. Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *KDD*. ACM, 137–146.

[20] Elias Boutros Khalil, Bistra Dilkina, and Le Song. 2014. Scalable diffusion-aware optimization of network topology. In *KDD*. ACM, 1226–1235.

[21] Arijit Khan. 2016. Towards Time-Discounted Influence Maximization. In *CIKM*. ACM, 1873–1876.

[22] Masahiro Kimura, Kazumi Saito, and Hiroshi Motoda. 2008. Solving the Contamination Minimization Problem on Networks for the Linear Threshold Model. In *PRICAI (Lecture Notes in Computer Science)*, Vol. 5351. Springer, 977–984.

[23] Thomas Lengauer and Robert Endre Tarjan. 1979. A Fast Algorithm for Finding Dominators in a Flowgraph. *ACM Trans. Program. Lang. Syst.* 1, 1 (1979), 121–141.

[24] Guoliang Li, Shuo Chen, Jianhua Feng, Kian-Lee Tan, and Wen-Syan Li. 2014. Efficient location-aware influence maximization. In *SIGMOD Conference*. ACM, 87–98.

[25] Bo Liu, Gao Cong, Dong Xu, and Yifeng Zeng. 2012. Time Constrained Influence Maximization in Social Networks. In *ICDM*. IEEE Computer Society, 439–448.

[26] Edward S. Lowry and C. W. Medlock. 1969. Object code optimization. *Commun. ACM* 12, 1 (1969), 13–22.

[27] Wei Lu, Wei Chen, and Laks V. S. Lakshmanan. 2015. From Competition to Complementarity: Comparative Influence Diffusion and Maximization. *Proc. VLDB Endow.* 9, 2 (2015), 60–71.

[28] Evgeny Morozov. 2009. Swine flu: Twitter's power to misinform. *Foreign policy* (2009).

[29] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.* 14, 1 (1978), 265–294.

[30] Hung T. Nguyen, My T. Thai, and Thang N. Dinh. 2016. Stop-and-Stare: Optimal Sampling Algorithms for Viral Marketing in Billion-scale Networks. In *SIGMOD Conference*. ACM, 695–710.

[31] Naoto Ohsaka, Takuya Akiba, Yuichi Yoshida, and Ken-ichi Kawarabayashi. 2014. Fast and Accurate Influence Maximization on Large Networks with Pruned Monte-Carlo Simulations. In *AAAI*. AAAI Press, 138–144.

[32] Michael Simpson, Laks V. S. Lakshmanan, and Farnoosh Hashemi. 2022. Misinformation Mitigation under Differential Propagation Rates and Temporal Penalties. *Proc. VLDB Endow.* 15, 10 (2022), 2216–2229.

[33] Jing Tang, Xueyan Tang, Xiaokui Xiao, and Junsong Yuan. 2018. Online Processing Algorithms for Influence Maximization. In *SIGMOD Conference*. ACM, 991–1005.

[34] Jing Tang, Xueyan Tang, and Junsong Yuan. 2017. Influence Maximization Meets Efficiency and Effectiveness: A Hop-Based Approach. In *ASONAM*. ACM, 64–71.

[35] Jing Tang, Xueyan Tang, and Junsong Yuan. 2018. An efficient and effective hop-based approach for influence maximization in social networks. *Social Network Analysis and Mining* 8 (2018), 1–19.

[36] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence Maximization in Near-Linear Time: A Martingale Approach. In *SIGMOD Conference*. ACM, 1539–1554.

[37] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD Conference*. ACM, 75–86.

[38] Guangmo Amo Tong and Ding-Zhu Du. 2019. Beyond Uniform Reverse Sampling: A Hybrid Sampling Technique for Misinformation Prevention. In *INFOCOM*. IEEE, 1711–1719.

[39] Guangmo Amo Tong, Weili Wu, Ling Guo, Deying Li, Cong Liu, Bin Liu, and Ding-Zhu Du. 2017. An efficient randomized algorithm for rumor blocking in online social networks. In *INFOCOM*. IEEE, 1–9.

[40] Jinghao Wang, Yanping Wu, Xiaoyang Wang, Ying Zhang, Lu Qin, Wenjie Zhang, and Xuemin Lin. 2024. Efficient Influence Minimization via Node Blocking. *arXiv preprint arXiv:2405.12871* (2024).

[41] Senzhang Wang, Xiaojian Zhao, Yan Chen, Zhoujun Li, Kai Zhang, and Jiali Xia. 2013. Negative Influence Minimizing by Blocking Nodes in Social Networks. In *AAAI (Late-Breaking Developments) (AAAI Technical Report)*, Vol. WS-13-17. AAAI.

[42] Xiaoyang Wang, Ying Zhang, Wenjie Zhang, and Xuemin Lin. 2016. Distance-aware influence maximization in geo-social network. In *ICDE*. IEEE Computer Society, 1–12.

[43] Xiaoyang Wang, Ying Zhang, Wenjie Zhang, and Xuemin Lin. 2017. Efficient Distance-Aware Influence Maximization in Geo-Social Networks. *IEEE Trans. Knowl. Data Eng.* 29, 3 (2017), 599–612.

[44] Xiaoyang Wang, Ying Zhang, Wenjie Zhang, Xuemin Lin, and Chen Chen. 2017. Bring Order into the Samples: A Novel Scalable Method for Influence Maximization. *IEEE Trans. Knowl. Data Eng.* 29, 2 (2017), 243–256.

[45] Zhefeng Wang, Yu Yang, Jian Pei, Lingyang Chu, and Enhong Chen. 2017. Activity Maximization by Effective Information Diffusion in Social Networks. *IEEE Trans. Knowl. Data Eng.* 29, 11 (2017), 2374–2387.

[46] Jiadong Xie, Fan Zhang, Kai Wang, Xuemin Lin, and Wenjie Zhang. 2023. Minimizing the Influence of Misinformation via Vertex Blocking. In *ICDE*. IEEE, 789–801.

[47] Yuqing Zhu, Jing Tang, Xueyan Tang, Sibo Wang, and Andrew Lim. 2023. 2-hop+ Sampling: Efficient and Effective Influence Estimation. *IEEE Trans. Knowl. Data Eng.* 35, 2 (2023), 1088–1103.