Time-aware Influence Minimization via Blocking Social Networks

Xueqin Chang[♮], Jiajie Fu[♮], Qing Liu[♮], Yunjun Gao[♮], Baihua Zheng[♯]
[♮]*Zhejiang University,* [♯]*Singapore Management University*
{changxq, jiajiefu, qingliucs, gaoyj}@zju.edu.cn, bhzheng@smu.edu.sg

***Abstract*—In this paper, we investigate the Time-aware Influence Minimization (TIMIN) problem in social networks, focusing on minimizing negative influence concerning a critical deadline by temporarily blocking specific nodes in the given social network. First, we introduce the *Temporal Linear Threshold* (TLT) model, a novel framework that incorporates time delay in influence propagation, the decay of influence power over time, and the lifecycle of influence. Building on this model, we formally define the TIMIN problem and prove its NP-hardness, *monotonicity*, and *supermodularity*. To tackle the TIMIN problem, we develop the Timin-Greedy, a greedy algorithm that achieves $(1-1/e)$ approximation. Since exact computation of negative influence spread for any node set in Timin-Greedy is #P-hard, we propose TESTIM, a scalable implementation that provides $(1-1/e-\epsilon)$ approximation. To further enhance the efficiency, we introduce NReplacer, a heuristic algorithm leveraging the insight that potential blocking nodes often cluster near the negative source. Additionally, we explore two variants of the TIMIN problem that incorporate constraints related to time urgency and blocking costs. Our extensive experimental evaluations demonstrate several key findings: (1) TESTIM is up to 10× faster than the baselines while achieving 30%–50% more reductions in negative influence spread, and (2) NReplacer exhibits a 5× speedup compared to TESTIM, with comparable reductions in negative influence spread.**

***Index Terms*—Influence Minimization, Temporal Information Propagation Model, Social Network**

I. INTRODUCTION

Influence minimization, with a wide range of applications, aims to minimize the expected influence of negative phenomena, be it fake news, rumors, or infectious diseases [1]–[5]. For example, in new product launch campaigns, influence minimization can mitigate the adverse impact of fake news on companies and their products; in the realm of epidemiology control, it can curtail the spread of infectious diseases.

Approaches proposed for influence minimization can be categorized as *clarification-based* methods [1], [2], [6], [7] and *blocking-based* methods [3], [8]–[10]. The former promotes positive content to enhance user awareness and reduce the acceptance of negative information. In contrast, the latter removes (or monitors) some critical users or connections from networks to minimize the spread of negative influences. However, clarification-based methods may face limitations in effectiveness in real applications. A study in Science [11] observed that negative information propagates 6× faster than positive information. There may be scenarios where, at *a particular timestamp*, the negative influences have reached a wide audience, while positive influences have just started to spread or have reached only a small audience. This can render clarification-based methods less effective, especially when a critical deadline is imposed for influence minimization, such as the end date of the product launch. In this paper, we consider a critical deadline for influence minimization, and consequently, we employ *blocking techniques* to solve this problem.

Many existing propagation models in influence minimization research assume immediate influence propagation at successive timestamp [8]–[10], [12], and assign fixed influence powers to users [1], [8], [13], [14]. However, real-world influence dynamics are more complex. Firstly, there are often delays in influencing others [1], [2], [13], [15]. For example, users in online social networks may not immediately react to newly received information from friends (only 25% Facebook users open the app daily [16]). Similarly, disease transmission rates vary due to different viral incubation periods [17]. Secondly, influence power can decay over time and is not everlasting, due to factors such as decreasing information freshness, credibility [18], or viral load [19].

To address these complexities, we introduce a novel *Temporal Linear Threshold* (TLT) model to simulate the temporal negative information or disease diffusion. In the TLT model, influence weights decrease over time. Users receive influence weights from their active in-neighbors after certain time delays, and continually accumulate these weights over a survival period. Activation occurs when the accumulated influence *live-weights* exceed a trust threshold. Unlike existing propagation models, TLT integrates time delays in influence propagation, decay of influence power over time, and the lifecycle of influence simultaneously.

Time-aware Influence Minimization Problem. In this paper, we study the Time-aware Influence Minimization (TIMIN) problem via blocking social networks. Based on the TLT model, we formally define the TIMIN problem as follows. Given a graph $G(V, E)$, a set of negative seed nodes $S_n \subseteq V$, a budget k, and a deadline T, the TIMIN problem aims to identify a node set $B \subseteq V \setminus S_n$ with $|B| = k$, such that after blocking B, the *expected negative influence spread* (defined in Definition 2) under the TLT model is minimized at the deadline T. The TIMIN problem has many real-life applications.

Application 1. *During sensitive periods such as product launches, competitors may spread fake news across social platforms. However, the diffusion of such information can be delayed due to various online user behaviors [16], and its impact may diminish over time as the credibility of news decreases [18]. To protect companies and their new products from the detrimental effects of fake news, it is crucial to mitigate its spread promptly. Thus, companies report fake news to social platforms, which can then utilize* TIMIN *to identify* *key users in social networks and temporarily deactivate their accounts until the product launch concludes. This strategy aims to reduce the negative influence of fake news effectively.*

Application 2. *Some infectious diseases exhibit varying transmission times due to viral incubation periods and individual immune responses [17]. Moreover, as the viral load in infectious patients diminishes over time, the potency of transmission decreases [19]. To safeguard public health and relieve the strain on healthcare systems, it is essential to minimize the spread of infectious diseases before effective treatments/vaccines become available. Thus, the center for disease control and prevention can employ* TIMIN *to quarantine critical individuals and restrict the scope of disease transmission before the medical supplies are adequately prepared.*

Challenges and Solutions. Existing works [3], [9], [10], [20], [21] have proposed blocking-based methods to address influence minimization problem but have largely overlooked temporal factors. However, these methods are unsuitable for tackling the TIMIN problem for three key reasons. **(1)** The influence minimization under certain propagation models, such as the *Independent Cascade* (IC) model [22] and *Susceptible-Infected-Recovered* (SIR) model [23], is non-supermodular. Thus, the corresponding algorithms [2], [10], [20] are not theoretically guaranteed and cannot be applied to the TIMIN problem, which is supermodular. **(2)** Existing greedy algorithms for the influence minimization [3], [8], [9], [14] under the *Linear Threshold* (LT) model [22] do not consider temporal factors. As a result, the returned blocking nodes are ineffective in reducing the spread of negative influence when the temporal factors are taken into account. **(3)** Existing approaches [3], [9] that use reverse influence sampling (*RIS*) technique [24] to tackle influence minimization are all based on the IMM algorithm [25], leading to two significant drawbacks. (i) IMM requires applying union bounds to all size-k node sets, thereby increasing the failure probability by a factor of $\binom{n}{k}$ [26]. (ii) IMM necessitates generating a large number of samples to ensure the approximation. Therefore, it is necessary to develop new efficient algorithms for the TIMIN problem.

First, we introduce the concept of *temporal live-edge graph*, and utilize it to prove that the TIMIN problem is **NP**-hard, *monotone*, and *supermodular* under the TLT model. Considering the inherent complexity of the TIMIN problem and the fact that only successful propagation from negative seeds can be potentially reduced by blocking nodes under time constraints, we firstly propose Timin-Greedy, a greedy algorithm that offers a $(1-1/e)$-approximation based on the monotonicity and supermodularity properties of the TIMIN problem. However, the computation of the exact negative influence spread for any node set in Timin-Greedy is $\#$**P**-hard, making it impractical for large graphs. Then, we propose a new *Temporal Reverse Influence Sampling* (*TRIS*) technique. Specifically, *TRIS* first samples two sets, including random temporal-reverse reachable (T-RR) sets and reverse reachable-blocking (RR-B) sets. The T-RR sets are used to estimate the expected negative influence spread, while the RR-B sets are utilized to select the optimal blocking set. Then, along with two new unbiased estimation methods based on T-RR and RR-B sets, we propose TESTIM, a trial-and-error algorithm that leverages the *TRIS* to greedily select k blocking nodes with a $(1-1/e-\epsilon)$-approximation guarantee. Compared to existing *RIS* technique for influence minimization, the novelty of our proposed *TRIS* technique is two-folded: (i) *TRIS* improves efficiency by stopping the sampling process early for invalid nodes, and (ii) *TRIS* samples two sets of samples, T-RR sets and RR-B sets, to estimate negative influence spread and select the blocking set, respectively. Furthermore, motivated by the observation that potential blocking nodes often locate among the nearby neighbors of the negative seed nodes, we design a heuristic algorithm called NReplacer to improve efficiency. Specifically, NReplacer builds an out-neighbor-index H, greedily selects k blocking nodes from H, and replaces them with nodes that have the largest marginal loss from the next hop neighbors of the replaced nodes.

Moreover, to explore the significance of deadline and budget constraints, we study two practical variations of the TIMIN problem, called Deadline-sensitive TIMIN problem and Budget-sensitive TIMIN problem, and extend our proposed algorithms to efficiently handle these two variations.

Contributions. Our contributions are summarized as follows.

- We introduce a new TLT model by considering the temporal aspects. Based on it, we formulate the TIMIN problem, and prove it is **NP**-hard, *monotone*, and *supermodular*.
- We propose a greedy algorithm Timin-Greedy and a scalable version TESTIM, with $(1-1/e)$ and $(1-1/e-\epsilon)$ approximations, respectively. We also devise a heuristic algorithm NReplacer to further improve the efficiency.
- We present two variants of TIMIN problem, i.e., DSTIMIN and BSTIMIN, and extend the TESTIM algorithm to handle them in polynomial time with approximation guarantees.
- Extensive experiments on five real networks and a visualized case study on Facebook demonstrate the effectiveness, efficiency, and scalability of the proposed algorithms.

Roadmap. Section II formally defines the TLT model and TIMIN problem. Section III introduces algorithms for the TIMIN problem. Section IV presents two variants of the TIMIN problem and the corresponding solutions. Section V reports experimental results. Section VI reviews the related work, and Section VII concludes the paper.

II. PRELIMINARIES

In this section, we first propose the Temporal Linear Threshold (TLT) model. Based on it, we formally define and analyze the problem of Time-aware Influence Minimization (TIMIN).

We consider a directed graph $G(V,E)$, where V and $E \subseteq V \times V$ are the sets of n nodes and m edges, respectively. Each edge $e=(u,v) \in E$ is associated with **(1)** an influence weight $w_{u,v}$ quantifying the influence of u on v, and **(2)** a delay time $\varphi_{u,v}$ indicating the delay of influence from u to v. That is, if a node u is active at timestamp t, then v will be influenced by u with influence weight $w_{u,v}$ at timestamp $t+\varphi_{u,v}$. We assume that the delay times of all edges follow *Poisson* or

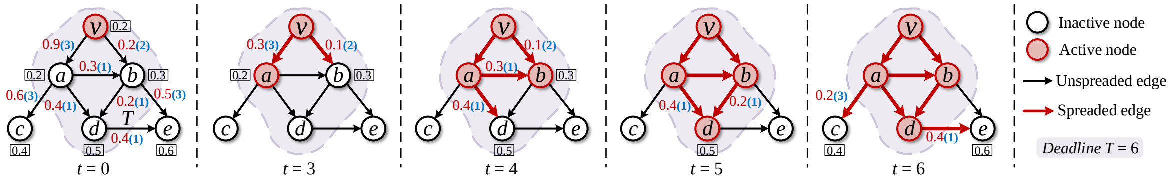


Fig. 1. Illustrating influence propagation under *Temporal Linear Threshold* (TLT) model; red numbers on the edges are influence weights, blue numbers inside the brackets are time delays, and numbers besides nodes are activation thresholds

Geometric distribution, as with [13], [27]. In real applications, *the influence weights decay over time and are not everlasting.* Thus, we introduce an influence decay function $f(\varphi)$ and the maximum survival time φ_{max} for influence weights [28], and define the *live-weight* as follows.

Definition 1. (Live-weight) *Given a graph $G(V, E)$, an edge $(u, v) \in E$, an influence decay function $f(\varphi)$, and a maximum survival time φ_{max}, assume that at timestamp t, the active node u starts to influence v with the influence weight $w_{u,v}$. $w_{u,v}$ remains alive during the time interval $[t, t+\varphi_{max}]$. After time delay $\varphi_{u,v}$, the influence weight $w_{u,v}$ becomes:*

$$w_{u,v}(\varphi_{u,v}) = \begin{cases} f(\varphi_{u,v}) \cdot w_{u,v} & 0 < \varphi_{u,v} \le \varphi_{max}, \\ 0 & \varphi_{u,v} > \varphi_{max}. \end{cases} \quad (1)$$

Definition 1 specifies the life cycle of an influence weight. The influence decay function $f(\varphi_{u,v})$ models the reduction in the influence weight $w_{u,v}$ w.r.t. the time delay $\varphi_{u,v}$. We primarily consider *power-law* and *exponential* functions [29], [30] for this decay. Specifically, for the power-law function, $f(\varphi_{u,v}) = \varphi_{u,v}^{-\alpha}$, and for the exponential function, $f(\varphi_{u,v}) = e^{-\alpha\varphi_{u,v}}$, where $\alpha \ge 0$. Note that, the influence weights are always non-negative, and if the delay time exceeds φ_{max}, the influence weight drops to zero. We employ $\widetilde{w}_{u,v}(t)$ to denote the live-weight at timestamp t. Hereinafter, when the context is clear, we will use "live-weight" and "weight" interchangeably.

A. TLT *Model*

Based on the above settings, we present the TLT model. Given a graph $G(V, E)$, a set of negative seed nodes $S_n \subset V$ being active at timestamp 0, and a *deadline* T, influence propagation under the TLT model occurs at each timestamp t until the deadline T as follows.

Propagation process of the TLT model. Initially, each node $v \in V$ is inactive and is assigned an activation threshold θ_v uniformly at random from the range [0, 1]. The sum of the weights of all incoming edges for each node is normalized to be at most 1 (i.e., $\sum_{u:(u,v)\in E} w_{u,v} \le 1$). At timestamp 0, each node $s \in S_n$ becomes active. At any timestamp $t \in [1, T]$, if a node u is active at $t-1$, it attempts to activate its inactive out-neighbor v with influence weight $w_{u,v}$, and the node v receives the decaying influence weight at timestamp $t + \varphi_{u,v}$. Then, at any timestamp $t' \in [t, T]$, the node v becomes active if the sum of the *live-weights* from v's active in-neighbors exceeds v's activation threshold, i.e., $\sum_{u:(u,v)\in E \wedge u \text{ is activated}} \widetilde{w}_{u,v}(t') \ge \theta_v$. Otherwise, v remains inactive. Once v becomes active, it remains active until the end of the propagation process. The TLT propagation terminates when it meets the deadline T or no nodes can be activated.

Example 1. *Figure 1 shows the influence propagation process under* TLT *model. Let v be the negative seed, the deadline $T = 6$, the decay function $f(\varphi) = 1/\varphi$, and the maximum survival time $\varphi_{max} = 4$. The influence propagates as follows.*

- *$t = 0$, node v becomes active.*
- *$t = 3$, node a receives the influence weight from v (as $\varphi_{v,a} = 3$) and is activated since $\widetilde{w}_{v,a}(3) = 0.9 \times 1/3 = 0.3 > \theta_a = 0.2$. Note that although node b has received the influence weight from v at $t = 2$, b remains inactive since $\widetilde{w}_{v,b}(3) = 0.2 \times 1/2 = 0.1 < \theta_b = 0.3$.*
- *$t = 4$, b receives additional influence weight from a and becomes active since $\widetilde{w}_{v,b}(4) + \widetilde{w}_{a,b}(4) = 0.1 + 0.3 = 0.4 > \theta_b = 0.3$. Meanwhile, node d also receives the influence weight from a. However, since $\widetilde{w}_{a,d}(4) = 0.4 < \theta_d = 0.5$, d remains inactive.*
- *$t = 5$, node d is influenced by node b and its accumulated live-weights is $\widetilde{w}_{a,d}(5) + \widetilde{w}_{b,d}(5) = 0.4 + 0.2 = 0.6 > \theta_d = 0.5$. Thus, d is successfully activated by a and b.*
- *$t = 6$, nodes c and e remain inactive and the influence propagation process terminates with active node set $\{v, a, b, d\}$.*

Comparison with existing time-aware influence propagation models. In particular, the TCIC model [1] considers distinct propagation rates of truth and misinformation, while the IC-M model [13] introduces user-level online delays to capture users' log-in and log-out behavior. Both TCIC and IC-M models are extensions of the IC model, in which the activation of nodes is simply subject to random variables. Additionally, the T-DLT [14] model considers the maximum propagation hops among users, and the TVLT [18] model incorporates time-decaying probabilities during the process of maximizing influence. Both T-DLT and TVLT models are extensions of the LT model. *To our best knowledge, our* TLT *model is the first to simultaneously incorporate connection-level time delays, influence power decays, and the lifecycle of influence into the* LT *model to simulate the temporal propagation process.*

B. *Problem Definition*

Based on the TLT model, we present the formal problem statement of TIMIN. First, we define the *expected influence spread*, which quantifies the influence of a set of seeds by the deadline T on the graph.

Definition 2. (Expected influence spread) *Given a graph $G(V, E)$, a set of negative seed nodes $S_n \subset V$, and a deadline T, the expected influence spread, denoted by $\sigma(S_n, G, T)$, is the expected number of active nodes by the deadline T under the* TLT *model.*

In order to minimize the spread of negative influence, we try to block some key nodes, except the negative seed nodes, to prevent their activation and the subsequent spread of negative influence in the diffusion process. Then, we introduce the notion of *blocking node* in the graph.

Definition 3. (Blocking node) *Given a graph $G(V, E)$ and a node v in V, if v is a blocking node, the influence weight of all v's incoming edges is set to 0, i.e., $\forall\ (u, v) \in E$, $w_{u,v} = 0$.*

As all incoming edges of blocking nodes carry zero influence weight, the accumulated influence weights will be zero as well, rendering the blocking nodes inactive throughout influence propagation. Building upon the aforementioned definitions, we formally define the TIMIN problem.

Problem 1. (TIMIN) *Given a graph $G(V, E)$, a set of negative seed nodes $S_n \subset V$, a budget k, and a deadline T, the objective of* TIMIN *is to find a blocking set $B \subseteq V \setminus S_n$ with k nodes that minimizes the expected influence spread by the deadline T. Formally:*

$$B = \arg\min_{B \subseteq (V \setminus S_n) \wedge |B| = k} \sigma(S_n, G[V \setminus B], T)$$

C. Problem Analyses

In this section, first, we show the **NP**-hardness of the TIMIN problem. Then, we introduce the concept of *temporal live-edge graph* to prove that the TIMIN problem is *monotone* and *supermodular* under the TLT model.

Theorem 1. *The* TIMIN *is* **NP***-hard under the* TLT *model.*

Proof. It has been proved that the influence minimization problem under the classical LT model is **NP**-hard [21]. We construct a case of the TIMIN by setting that (1) the time delay distribution follows the Geometric distribution with probability of 1; (2) the parameter α of the influence decay function is $\alpha = 0$; (3) the maximum survival time φ_{max} is infinite; and (4) the deadline T is infinite. Then, the TIMIN is a special case of influence minimization under the TLT model. Therefore, the TIMIN is **NP**-hard. □

Temporal live-edge graph. Kempe et al. [22] illustrated an alternative description of the LT model using *live-edge graphs*. To extend it to our TLT model, we incorporate the temporal information into the live-edge graphs. The generation process of a random *temporal live-edge graph* $X_t = (V, E_{X_t})$ can be summarized as follows. For each $v \in V$, we randomly sample at most one live-edge $(u, v) \in E$ from its incoming edges with a probability $w_{u,v}(\varphi_{u,v})$. No live-edge is selected with probability $1 - \sum_{u:(u,v)\in E} w_{u,v}(\varphi_{u,v})$. Consequently, each X_t exists with a probability $P(X_t|G) = \prod_{(u,v)\in E_{X_t}} w_{u,v}(\varphi_{u,v}) \cdot \prod_{(u,v)\in E\setminus E_{X_t}} (1 - w_{u,v}(\varphi_{u,v}))$. All the selected live-edges form the temporal live-edge set E_{X_t}, which is a subset of E, i.e., $E_{X_t} \subseteq E$, and these edges are unweighted. Note that, the sampling method for temporal live-edge graphs closely resembles the one employed for live-edge graphs, with the key distinction being the inclusion of time-decay on each edge.

Moreover, in the context of the TLT model, for a node v to become active, *three conditions must be satisfied*: **(1)** there should exist a live-edge path from a seed node $s \in S_n$ to $v \in V$ in the temporal live-edge graph X_t; **(2)** the cumulative time-delay $\varphi_{s,v}$ must not exceed the *deadline* T; and **(3)** most importantly, there must be no blocking nodes in B on this path. Let $A_{X_t}(S_n, G[V \setminus B], T)$ denote the set of nodes in X_t that can be reached from the seed set S_n, and $\sigma(S_n, G[V \setminus B], T)$ represent the expected influence spread of S_n under the TLT model. Then, the expected influence spread can be computed as follows.

$$\begin{aligned} &\sigma(S_n, G[V \setminus B], T) \\ =&\mathbb{E}[|A_{X_t}(S_n, G[V \setminus B], T)|] \\ =&\sum\nolimits_{X_t \in \mathcal{X}^t_{G\setminus B}} P(X_t|G \setminus B) \cdot |A_{X_t}(S_n, G[V \setminus B], T)| \end{aligned} \tag{2}$$

In Eq. (2), $\mathcal{X}^t_{G\setminus B}$ denotes the space of all possible temporal live-edge graphs based on $G \setminus B := (V \setminus B, E)$. The expectation is computed over the distribution of these temporal live-edge graphs. Assuming a blocking set B is already identified, adding another node $b \in V \setminus B$ to B decreases the influence by $\sigma(S_n, G[V \setminus B], T) - \sigma(S_n, G[V \setminus (B \cup \{b\})], T)$. The challenge lies in the *changing set of temporal live-edge graphs and the associated probabilities involved in computing the influence function*, when nodes are removed from the graph. It is not immediately clear whether this function remains monotone, and a similar challenge arises in proving supermodularity. It has been shown in [21] that under the classical LT model without temporal constraints, $\sigma(s, G[V \setminus B])$ for each $s \in S_n$ forms a monotonically decreasing and supermodular function of B, as detailed in following Lemmas [21].

Lemma 1. *Given a graph $G(V, E)$ and a set of negative seeds S_n, for any blocking set $B \subseteq V \setminus S_n$ and any $b \in V \setminus (S_n \cup B)$,*

$$\sigma(s, G[V \setminus B]) - \sigma(s, G[V \setminus (B \cup \{b\})]) \geq 0 \tag{3}$$

Lemma 2. *Given a graph $G(V, E)$ and a set of negative seeds S_n, for any blocking sets $B, B' \subseteq V \setminus S_n$, $B \subseteq B'$ and any $b \in V \setminus (S_n \cup B')$, let $R(b|B) = \sigma(s, G[V \setminus B]) - \sigma(s, G[V \setminus (B \cup \{b\})])$ (resp. $R(b|B') = \sigma(s, G[V \setminus B']) - \sigma(s, G[V \setminus (B' \cup \{b\})])$) denote the reduction in influence when node b is added into B (B'),*

$$R(b|B) - R(b|B') \geq 0 \tag{4}$$

However, in [21], the total influence is estimated by directly *summing* the influence of individual nodes in the seed sets. In contrast, our problem defines total influence as the *union* of influence from individual nodes in the seed sets. Thus, we propose the following theorems to demonstrate that our TIMIN problem retains the properties of being monotonically non-increasing and supermodular under the TLT model.

Theorem 2. *The expected influence spread $\sigma(S_n, G[V \setminus B], T)$ is monotonically non-increasing under the* TLT *model.*

Proof. In Lemma 1, it is clear that when a node is newly blocked from the graph without considering the temporal factors, the set of live-edge graphs and the associated probabilities involved in computing the influence function are not affected. Thus, we need to demonstrate that the above analysis still

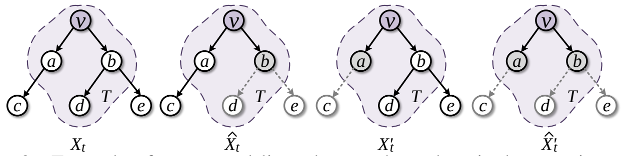


Fig. 2. Example of a temporal live-edge graph; node v is the negative seed node; grey nodes are blocked nodes

holds within the temporal factors. (1) The time-delays on the sampled live-edges and the deadline T remain unchanged when a node is removed from the graph. Consequently, the inclusion of time-delays on the sampled live-edges and T does not affect the probability of generating a particular temporal live-edge graph X_t. (2) We demonstrate that within the temporal settings, $|A_{X_t}(S_n, G[V \setminus B], T)| - |A_{X_t}(S_n, G[V \setminus (B \cup \{b\})], T)| \geq 0$ holds. However, the time-delays and deadline affect the node set in a certain X_t that can be reached from S_n, i.e., $A_{X_t}(S_n, G[V \setminus B], T)$. We consider two cases as follows. (i) Selecting the blocking node out of the deadline T does not cause any decrease or increase in the number of nodes influenced by S_n in X_t. (ii) If a blocking node is selected within T, it is equivalent to the scenario described in Lemma 1 with a special case $S_n = \{s\}$, where monotonicity is satisfied. Combining (i) and (ii), we have $|A_{X_t}(S_n, G[V \setminus B], T)| - |A_{X_t}(S_n, G[V \setminus (B \cup \{b\})], T)| \geq 0$. Therefore, the function $\sigma(S_n, G[V \setminus B], T)$ is monotonically non-increasing, i.e., $\sigma(S_n, G[V \setminus B], T) - \sigma(S_n, G[V \setminus (B \cup b), T]) \geq 0$. □

Theorem 3. *The expected influence spread $\sigma(S_n, G[V \setminus B], T)$ is supermodular under the* TLT *model.*

Proof. As illustrated in the proof of Theorem 2, the inclusion of the time-delays and deadline does not impact the space of temporal live-edge graphs $\mathcal{X}^t_{G \setminus B}$ or the corresponding probabilities $P(X_t | G \setminus B)$. Based on Lemma 2, we have to demonstrate that within the temporal settings, $|A_{X_t}(S_n, G[V \setminus B], T)| - |A_{X_t}(S_n, G[V \setminus (B \cup \{b\})], T)| \geq |A_{X_t}(S_n, G[V \setminus B'], T)| - |A_{X_t}(S_n, G[V \setminus (B' \cup \{b\})], T)|$ holds, where $B \subseteq B'$. Here, we denote $X_t = (V \setminus B, E_{X_t})$, $\hat{X}_t = (V \setminus B \cup \{b\}, E_{\hat{X}_t})$, $X'_t = (V \setminus B', E_{X'_t})$ and $\hat{X}'_t = (V \setminus B' \cup \{b\}, E_{\hat{X}'_t})$. All graphs are sampled under the same deadline, with $B' = B \cup \{a\}$, as depicted in Figure 2. Since temporal live-edge graphs are structured in such a way that each node has at most one incoming edge, resulting in each reachable node having a unique path from the seed node. Moreover, we have (1) a reachable live-edge path in X'_t is clearly also present in X_t, suggesting that if removing node b from X'_t leads to unreachability of some nodes in $\hat{X}'_t$, then those same nodes become unreachable when removing b from X_t; (2) if node a is selected within T, removing node b from X_t may disconnect some additional nodes whose paths from the seed v involve node a; and (3) if node a is selected without T, removing node b from X_t does not affect any nodes whose paths from the seed v involve node a. Therefore, the reduction in influenced nodes when removing node b from X_t is the same or larger than the reduction when removing b from X'_t. In conclusion, the function $\sigma(S_n, G[V \setminus B], T)$ is supermodular, i.e., $R(b|B, T) - R(b|B', T) \geq 0$, where $R(b|B, T) = \sigma(S_n, G[V \setminus B], T) - \sigma(S_n, G[V \setminus (B \cup \{b\})], T)$ (resp. $R(b|B', T) = \sigma(S_n, G[V \setminus B'], T) - \sigma(S_n, G[V \setminus (B' \cup \{b\})], T)$), $B \subseteq B'$. □

Algorithm 1 Timin-Greedy

Input: $G(V, E)$, S_n, k, T
Output: B

1: $B \leftarrow \emptyset$;
2: $\forall v \in V$, $\Delta(v|B, T) \leftarrow 0$;
3: **while** $|B| < k$ **do**
4: **for** $\forall v \in V \setminus (S_n \cup B)$ **do**
5: $\Delta(v|B, T) \leftarrow \sigma(S_n, G[V \setminus B], T) - \sigma(S_n, G[V \setminus (B \cup \{v\}), T]);$
6: $v^* \leftarrow \arg\max_{v \in V \setminus (S_n \cup B)} \Delta(v|B, T)$;
7: $B \leftarrow B \cup \{v^*\}$;
8: **Return** B;

III. TIMIN ALGORITHMS

In this section, we introduce the Timin-Greedy algorithm for the TIMIN problem, which achieves a $(1 - 1/e)$-approximation. Due to the complexity of efficiently implementing Timin-Greedy, we present its scalable version, called TESTIM algorithm, which achieves a $(1 - 1/e - \epsilon)$-approximation. Finally, we present a heuristic algorithm NReplacer to further improve the efficiency.

A. *Timin-Greedy Algorithm*

Since the TIMIN problem is *monotone* and *supermodular* under the TLT model, we introduce Timin-Greedy as our first proposal. The basic idea of Timin-Greedy is to iteratively select a blocking node that maximizes the influence reduction. Algorithm 1 outlines the pseudo-code of Timin-Greedy. Initially, Timin-Greedy initializes an empty blocking set B, and for each node $v \in V$, it sets the temporal marginal loss $\Delta(v|B, T) = 0$ (Lines 1–2). In each iteration (Lines 3–7), Timin-Greedy computes the temporal marginal loss $\Delta(v|B, T)$ for all nodes in $V \setminus (S_n \cup B)$ (Lines 4–5), selects the node with the maximum reduction in expected spread (Line 6), and adds it to B (Line 7). The process continues until k blocking nodes are selected. Finally, the algorithm returns the blocking node set B (Line 8). It is important to note that, given a negative seed set S_n, the computation of the temporal marginal loss for a node v must satisfy the following conditions: **(1)** there must exist an activation path from a node $s \in S_n$ to any node $u \in V$ through edges in E; **(2)** the cumulative time-delays should not exceed the deadline, i.e., $\varphi_{s,u} \leq T$; and **(3)** v must lie on the path from s to u.

The approximation of Timin-Greedy is guaranteed by Lemma 3.

Lemma 3. *For the* TIMIN *problem, let $R(B, T) = \sigma(S_n, G[V], T) - \sigma(S_n, G[V \setminus B], T)$ denote the reduction in influence spread by the deadline T under a blocking B in G, and B^* denote the optimal solution of the* TIMIN *problem. The solution B returned by Timin-Greedy satisfies*[1] *:*

$$R(B, T) \geq (1 - 1/e) \cdot R(B^*, T). \tag{5}$$

[1]Due to the space limitation, all the omitted proofs are moved to [31].

B. TESTIM Algorithm

Timin-Greedy requires numerous influence spread computations to identify the blocking set. However, computing the exact negative influence spread $\sigma(S_n, G[V], T)$ for any set S_n and deadline T under the TLT model is $\#\mathbf{P}$-hard, as shown in Theorem 4.

Theorem 4. *Given a graph $G(V, E)$, a negative seed set S_n, and a deadline T, it is $\#\mathbf{P}$-hard to compute the exact value of $\sigma(S_n, G[V], T)$ under the TLT model.*

Recent studies resort to sampling-based influence spread estimation, such as naive *Monte-Carlo (MC)* Simulations [22] and advanced *Reverse Influence Sampling (RIS)* [24]. Since RIS significantly improves the efficiency of estimating influence spread compared to MC simulations, it is widely used. To this end, in this paper, we propose a novel temporal reverse influence sampling method to estimate the influence spread.

Temporal Reverse Influence Sampling.

To efficiently estimate the expected influence spread $\sigma(S_n, G[V \setminus B], T)$ for any blocking set B given a negative seed set S_n and a deadline T, rather than computing it exactly, we propose the *Temporal Reverse Influence Sampling (TRIS)* method. First, we introduce two key concepts, i.e., the *Temporal-Reverse Reachable* (T-RR) set and the *Reverse Reachable-Blocking* (RR-B) set, applicable to any possible temporal live-edge graph under the TLT setting. The T-RR set aims to estimate the influence spread of a negative seed set without any nodes being blocked, while the RR-B set is used for optimal greedy selection of blocking nodes. Based on T-RR and RR-B sets, *TRIS* mainly involves generating these two sets. Unlike traditional RR sets, which only consider node inclusion in an RR set, the T-RR and RR-B sets account for factors such as *influence-decay*, *cumulative time-delay* and the *deadline*. Their generation proceeds as follows.

Under the TLT model, a random T-RR set R_t on G can be generated in three steps: (1) select a node v randomly from the graph G, (2) generate a live-edge path from v, and (3) insert all nodes on the live-edge path (including v) into R_t. To generate a live-edge path, at each step, we have two options: to stop or to continue. Let u be the node selected at the previous step and I_u denote its in-neighbors. The probabilities of stopping and continuing live-edge path generation in this step are $1 - \sum_{w \in I_u} w_{w,u}(\varphi_{w,u})$ and $\sum_{w \in I_u} w_{w,u}(\varphi_{w,u})$, respectively. In addition, if u is a negative seed node, live-edge path generation is terminated. If not stopped, a node $w \in I_u$ is sampled with probability $w_{w,u}(\varphi_{w,u})$ and added to the live-edge path for the next step. At each step, we also record the cumulative time-delay between the current sampling node u and v, denoted by $\varphi_{u,v}$.

For a T-RR set, if it contains a negative seed node and its cumulative time-delay is no less than the deadline T, the T-RR set is considered an RR-B set. Correspondingly, an RR-B set implies that selecting a node from it as a blocking node can prevent node v from receiving negative information within the deadline, thereby preventing v from being influenced. Clearly, the number of RR-B sets does not exceed the number of T-RR sets. Fig. 3 illustrates an example of T-RR and RR-B sets. Assuming v_5 is initially selected and $T = 5$. Figs. 3(b), (c), and (d) depict three different T-RR sets. Only T-RR set R_{t3} qualifies as an RR-B set R_b.

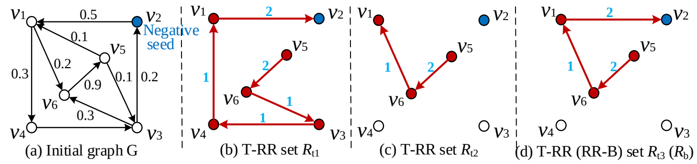


Fig. 3. An example of T-RR set and RR-B set

Based on the generated T-RR and RR-B sets, we need to design unbiased estimation methods to achieve two objectives: (1) estimate the influence spread of a negative seed set without considering blocking nodes using T-RR sets, and (2) greedily select optimal blocking nodes using RR-B sets. Traditional unbiased estimation methods typically do not consider temporal factors and only estimate the influence spread of a given node set based on RR sets. Thus, we develop a novel approach to estimate both the influence spread of a negative seed set and the influence reduction due to a blocking node set.

Given a negative seed set S_n, a random T-RR set R_t, and a deadline T, we define a random variable $\Lambda_{R_t}(S_n, G[V], T)$ such that $\Lambda_{R_t}(S_n, G[V], T) = 1$ iff (1) S_n intersects R_t, i.e., $S_n \cap R_t \neq \emptyset$, and (2) the cumulative time-delay from any negative seed $s \in S_n$ to the sampled source v in R_t does not exceed the deadline, i.e., $\varphi_{s,v} \leq T$. Otherwise, $\Lambda_{R_t}(S_n, G[V], T) = 0$. Condition (2) ensures that s has a possibility to activate v within the deadline. Thus, given a set $\mathcal{R}_t$ of random T-RR sets, we define $f^{\mathcal{R}_t}(S_n, G[V], T)$ as an unbiased estimation of $\sigma(S_n, G[V], T)$, where $\Lambda_{\mathcal{R}_t}(S_n, G[V], T) = \sum_{R_t \in \mathcal{R}_t} \Lambda_{R_t}(S_n, G[V], T)$. Formally,

$$f^{\mathcal{R}_t}(S_n, G[V], T) = \frac{\Lambda_{\mathcal{R}_t}(S_n, G[V], T)}{|\mathcal{R}_t|} \cdot n \tag{6}$$

Moreover, given a blocking set B and a random RR-B set R_b, we define a random variable $\Lambda_{R_b}(S_n, G[V \setminus B], T)$ such that $\Lambda_{R_b}(S_n, G[V \setminus B], T) = 1$ if there exists B intersecting R_b, i.e., $B \cap R_b \neq \emptyset$. Otherwise, $\Lambda_{R_b}(S_n, G[V \setminus B], T) = 0$. Given a set $\mathcal{R}_b$ of random RR-B sets, we use $g^{\mathcal{R}_b}(B, T)$ to denote an unbiased estimation of $R(B, T)$, which refers to the influence reduction of B, i.e., $R(B, T) = \sigma(S_n, G[V], T) - \sigma(S_n, G[V \setminus B], T)$. Formally,

$$g^{\mathcal{R}_b}(B, T) = \frac{\Lambda_{\mathcal{R}_b}(S_n, G[V \setminus B], T)}{|\mathcal{R}_b|} \cdot f^{\mathcal{R}_t}(S_n, G[V], T) \tag{7}$$

where $\Lambda_{\mathcal{R}_b}(S_n, G[V \backslash B], T) = \sum_{R_b \in \mathcal{R}_b} \Lambda_{R_b}(S_n, G[V \backslash B], T)$.

TESTIM. Based on the discussions above, we propose the *Trial-and-Error-based Scalable Time-aware Influence Minimization (TESTIM)* algorithm. TESTIM starts with an empty blocking node set B, utilizes the *TRIS* method to estimate influence spread of nodes, and iteratively selects a node with maximal marginal loss. Note that the returned B has a theoretical guarantee with sufficient RR-B sets. A key issue is *determining the sample size $|\mathcal{R}_b|$ required to achieve the approximation guarantee without excessive*

Algorithm 2 TESTIM

Input: $G(V,E)$, S_n, k, T, ϵ, δ
Output: B

1: $B \leftarrow \emptyset$; $\theta_1 \leftarrow n$; $i \leftarrow 1$;
2: **while true do**
3: Generate $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ with $|\mathcal{R}_{t1}| = |\mathcal{R}_{t2}| = \theta_i$;
4: Generate $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$ based on $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$;
5: Compute $f^{\mathcal{R}_{t1}}(S_n, G[V], T)$ and $f^{\mathcal{R}_{t2}}(S_n, G[V], T)$;
6: $B \leftarrow$ TIMIN-Oracle($\mathcal{R}_{b1}$);
7: Compute reduction influences $g^{\mathcal{R}_{b1}}(B,T)$ and $g^{\mathcal{R}_{b2}}(B,T)$;
8: $\lambda \leftarrow g^{\mathcal{R}_{b1}}(B,T)/g^{\mathcal{R}_{b2}}(B,T)$;
9: $\epsilon_1 \leftarrow (\epsilon_1+1)(\epsilon_1+2)/\epsilon_1^2 = g^{\mathcal{R}_{b2}}(B,T)/\ln(5\cdot i^2/\delta)\cdot|\mathcal{R}_{b2}|/n$;
10: $\epsilon_2 \leftarrow (2\epsilon_1+2)/\epsilon_2^2 = g^{\mathcal{R}_{b2}}(B,T)/\ln(5\cdot i^2/\delta)\cdot|\mathcal{R}_{b1}|/n$;
11: **if** $0 < \lambda \leq (\frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1})$, $\epsilon_1,\ \epsilon_2 \in (0,1)$ **then break**;
12: **if** $|\mathcal{R}_{b1}| \geq (8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2\cdot g^{\mathcal{R}_{b2}}(B,T)}$ **then break**;
13: $i \leftarrow i+1$; $\theta_i \leftarrow 2\theta_i$;
14: **Return** B;

computational overhead. To address this, TESTIM adopts a trial-and-error approach. During the generation of T-RR sets, we incrementally double the number of T-RR sets and monitor the approximation. TESTIM terminates under two conditions: when the influence reduction of B satisfies $0 < \frac{g^{\mathcal{R}_{b1}}(B,T)}{g^{\mathcal{R}_{b2}}(B,T)} \leq (\frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1})$, where $g^{\mathcal{R}_{b1}}(B,T)$ and $g^{\mathcal{R}_{b2}}(B,T)$ are the influenced reductions estimated on $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$, respectively, or when the number of RR-B sets is sufficient, i.e., $|\mathcal{R}_{b1}|$ reaches $(8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2\cdot g^{\mathcal{R}_{b2}}(B,T)}$. Detailed derivations of these formulas are provided below, with further details available in [31].

Algorithm 2 outlines the pseudo-code of TESTIM. Initially, TESTIM generates two sets $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ of T-RR sets with $|\mathcal{R}_{t1}| = |\mathcal{R}_{t2}| = n$, and two corresponding sets $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$ of RR-B sets based on $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ (Lines 3–4), respectively. It them uses $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ to unbiasedly estimate the influence spread of S_n without any nodes blocked (Line 5). Subsequently, TESTIM greedily selects the optimal blocking node set B via the function TIMIN-Oracle (Line 6). Following this, TESTIM verifies the quality of B based on $\mathcal{R}_{t2}$ and $\mathcal{R}_{b2}$ (Lines 7–12) which are independent of $\mathcal{R}_{t1}$ and $\mathcal{R}_{b1}$. If the reduction estimated from $\mathcal{R}_{b2}$ is much smaller than that from $\mathcal{R}_{b1}$, $\mathcal{R}_{b1}$ over-estimates B's reduction. In this case, TESTIM discards B, doubles the size of $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ (Line 13), and re-generates a new blocking node set until one of the following constraints is satisfied: **(1)** the parameter $\lambda = g^{\mathcal{R}_{b1}}(B,T)/g^{\mathcal{R}_{b2}}(B,T)$ satisfies the condition $0 < \lambda \leq (\frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1})$, where $\epsilon_1, \epsilon_2 \in (0,1)$ (Line 11). and **(2)** the number of generated RR-B sets $\mathcal{R}_{b1}$, i.e., $|\mathcal{R}_{b1}|$, reaches $(8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2\cdot g^{\mathcal{R}_{b2}}(B,T)}$ (Line 12). Finally, TESTIM returns the blocking node set B (Line 14).

Algorithm 3 describes the pesudo-code of TIMIN-Oracle, which is to greedily select the optimal blocking set B. It begins by initializing an empty set B (Line 1). In each iteration, TIMIN-Oracle calculates the number of RR-B sets covered by v (Lines 3-4), which denotes the temporal marginal loss $\Delta_{\mathcal{R}_b}(v)$ for v. chooses the node v^* that covers the most RR-B sets (Line 5), adds v^* to B (Line 6), and then removes all

Algorithm 3 TIMIN-Oracle

Input: $\mathcal{R}_b$
Output: B

1: $B \leftarrow \emptyset$;
2: **while** $|B| < k$ **do**
3: **for** each $v \in V \setminus (S_n \cup B)$ **do**
4: $\Delta_{\mathcal{R}_b}(v) \leftarrow$ the number of RR-B sets covered by v in $\mathcal{R}_b$;
5: $v^* \leftarrow \arg\max_{v \in V\setminus(S_n\cup B)} \Delta_{\mathcal{R}_b}(v)$;
6: $B \leftarrow B \cup \{v^*\}$;
7: Remove from $\mathcal{R}_b$ all RR-B sets that are covered by v^*;
8: **Return** B;

RR-B sets covered by v^* from $\mathcal{R}_b$ (Line 7). Finally, TIMIN-Oracle returns B (Line 8).

Theorem 5 shows the approximation guarantee provided by Algorithm 2, where B^* denotes the optimal solution and $\delta, \epsilon \in (0,1)$ are user-specified error parameters.

Theorem 5. (Theoretical guarantee of TESTIM). *For* TIMIN *problem, let $R(B,T)$ denote the reduction in influence spread before deadline T when B is blocked, and ϵ be the approximation factor for influence estimation by TRIS method. With a probability of at least $(1-\delta)$ for $\forall\ \delta \in (0,1)$, the solution B returned by TESTIM satisfies:*

$$R(B,T) \geq (1 - 1/e - \epsilon)\cdot R(B^*,T) \tag{8}$$

In what follows, *we address two key challenges in TESTIM while satisfying Theorem 5*: **(1)** determining the maximum number of RR-B sets $\mathcal{R}_{b1}$ (Line 12, Algorithm 2) and **(2)** establishing conditions to evaluate whether the current B satisfies the performance guarantee (Lines 11, Algorithm 2). We first propose the following concentration bounds based on the *Chernoff Inequalities* [32].

$$\begin{aligned} &\Pr[g^{\mathcal{R}_{b2}}(B,T) - R(B,T) \geq \epsilon_1 \cdot R(B,T)] \\ &\leq \exp\left(-\frac{\epsilon_1^2}{2+\epsilon_1}\frac{|\mathcal{R}_{b2}|}{f^{\mathcal{R}_{t2}}(S_n,G[V],T)}R(B,T)\right) \\ &\Pr[g^{\mathcal{R}_{b1}}(B^*,T) - R(B^*,T) \leq -\epsilon_2 \cdot R(B^*,T)] \\ &\geq \exp\left(-\frac{\epsilon_2^2}{2}\frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n,G[V],T)}R(B^*,T)\right) \end{aligned}$$

Here, $|\mathcal{R}_{b1}|$ and $|\mathcal{R}_{b2}|$ are the sizes of $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$, respectively. Then, in each round of TESTIM, the estimations $g^{\mathcal{R}_{b1}}(B^*,T)$ and $g^{\mathcal{R}_{b2}}(B,T)$ are concentration bounds with a high probability as shown in Lemma 4.

Lemma 4. *With probability at least 1-$\frac{2\delta}{3}$, for each iteration of Algorithm 2, where $\epsilon_1, \epsilon_2, \lambda > 0$, we have*

$$g^{\mathcal{R}_{b2}}(B,T) \leq (1+\epsilon_1)\, R(B,T) \tag{9}$$

$$g^{\mathcal{R}_{b1}}(B^*,T) \geq (1-\epsilon_2)\, R(B^*,T) \tag{10}$$

Referring to Lemma 4, we consider two cases based on whether Line 11 in TESTIM is satisfied.

- **Case (1):** Line 11 is satisfied. In the final iteration, we have $0 < \lambda \leq (\frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1})$. By combining Eq. (9), $g^{\mathcal{R}_{b1}}(B,T) \geq (1-1/e)g^{\mathcal{R}_{b1}}(B^*,T)$, and $\lambda = g^{\mathcal{R}_{b1}}(B,T)/g^{\mathcal{R}_{b2}}(B,T)$, we prove that Eq. (8) holds with a probability of at least $(1-\frac{2\delta}{3})$.
- **Case (2):** Line 11 is not satisfied. When TESTIM terminates, according to Eq. (9), we have $g^{\mathcal{R}_{b2}}(B,T) \leq$

TABLE I

MOTIVATING EXAMPLES OF HEURISTIC ALGORITHM (WE SET $|S_n| = 5$ AND $|B| = 50$. N_i DENOTES THE NUMBER OF i-HOP OUT-NEIGHBORS OF S_n; B_i DENOTES THE NUMBER OF i-HOP OUT-NEIGHBORS OF S_n IN B)

Dataset	B_1/N_1	B_2/N_2	B_3/N_3	B_4/N_4	B_5/N_5	B_6/N_6
EmailCore	40/310	8/603	2/49	0/0	0/0	0/0
Epinions	34/100	10/3,505	5/19,352	1/19,446	0/4,474	0/667
Amazon	21/34	14/187	10/399	2/1,361	2/3,193	1/9,574

$(1+\epsilon_1)R(B^*,T)$. Let $x = \epsilon R(B^*,T)/2R(B,T)$ for any $B \subseteq (V \setminus S_n)$.

$$\begin{aligned}
&\Pr[g^{\mathcal{R}_{b1}}(B,T) - R(B,T) \geq \frac{\epsilon}{2} \cdot R(B^*,T)] \\
&\leq \exp\left(-\frac{x^2}{2+x}\frac{|\mathcal{R}_{b1}|}{n}R(B,T)\right) \\
&\leq \exp\left(-\frac{\epsilon^2}{8+2\epsilon}\frac{|\mathcal{R}_{b1}|}{n}R(B^*,T)\right) \leq \frac{\delta}{6 \cdot 2^n}.
\end{aligned} \tag{11}$$

Then, $|\mathcal{R}_{b1}| = (8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2 \cdot g^{\mathcal{R}_{b2}}(B,T)}$. Therefore, when Line 11 is not satisfied, Eq. (8) holds.

By combining these two cases, the approximation guarantee of TESTIM is established.

Theorem 6. (Time complexity of TESTIM). *The expected time complexity of TESTIM is $O(\frac{\max\{\ln\frac{1}{\delta},n\}\cdot m \cdot R(\{v^*\},T)}{\epsilon^2})$, where v^* is the node in $G[V \setminus S_n]$ with the largest expected marginal loss spread.*

C. *NReplacer Algorithm*

First, we present an observation regarding the blocking node set selected by the TESTIM algorithm. Assuming $|S_n| = 5$ and $|B| = 10$, Table I illustrates the composition of the blocking node set B across three graphs. It shows that a significant portion of B consists of 1-hop and 2-hop out-neighbors of the seed nodes. Nodes farther away from the seed nodes are less likely to be selected as blocking nodes. Motivated by this observation, we propose a heuristic algorithm called NReplacer. The basic idea is to first identify the k nearest out-neighbors of the seed nodes to maximize the negative influence reduction, forming the initial blocking node set B. Subsequently, it iteratively updates B by examining remaining out-neighbors from closest to farthest. During each iteration of the update process, NReplacer greedily selects the node with the largest marginal loss from the next hop out-neighbors of the node being replaced. This updating process terminates when no node with a larger marginal loss can be found.

Algorithm 4 shows the pseudo-code of NReplacer. First, it initializes an empty blocking set B, a maximum heap H to store the unchecked nodes along with their marginal losses (Line 1), and a set N to store the out-neighbors of S_n of different hops, where $N[i][j]$ denotes the j-th node in the i-th hop out-neighbors of S_n (Line 2). The algorithm constructs N and t using a BFS traversal of the graph (Line 3), and iteratively computes the marginal losses of out-neighbors at each hop (Lines 4–7). Then, it constructs B while updating the marginal losses of nodes in H (Lines 8–11). Subsequently, it updates the blocking nodes in B in reverse order of their insertions (Lines 12–19). In each iteration, the vertex u most recently inserted into B is removed (Line 13), and Algorithm 4 finds vertex v^*, the node in the next hop out-neighbors of u with the largest marginal loss (Lines 14–15). If v^* outperforms u, u is replaced by v^* (Lines 16–17). Otherwise, it continues to search for vertex v^* in subsequent next hop out-neighbors (Line 18). The update process terminates when no nodes with higher influence spread reductions are available (Line 19). Finally, B is returned (Line 20).

Algorithm 4 NReplacer

Input: $G(V,E)$, S_n, k, T
Output: B

1: $B \leftarrow \emptyset$; a maximum heap $H \leftarrow \emptyset$; $t \leftarrow 0$;
2: $N \leftarrow \emptyset$; //N records out-neighbors of S_n at each hop
3: $(N,t) \leftarrow BFS(G,S_n)$; //$t$ is the max hop of S_n's out-neighbors
4: **for** $i \leftarrow 1$ to t **do**
5:     **for** $j \leftarrow 1$ to $N[i].size()$ **do**
6:         Compute $\Delta(N[i][j]|B,T)$; Insert $N[i][j]$ into H;
7:     **if** $|H| \geq k$ **then break**
8: **while** $|B| < k$ **do**
9:     $v \leftarrow H.top()$; $B \leftarrow B \cup \{v\}$;
10:     **for** $\forall x \in H \setminus B$ **do**
11:         Update $\Delta(x|B,T)$ and $H.pop()$;
12: **for** each $u \in B$ in reverse order of insertion **do**
13:     $B \leftarrow B \setminus \{u\}$;
14:     **for** $i \leftarrow \kappa + 1$ to t **do** // $\kappa \leftarrow$ the hop that u belongs to
15:         $v^* \leftarrow \arg\max_{w \in N[i] \setminus (S_n \cup B)} \Delta(w|B,T)$;
16:         **if** $\Delta(v^*|B,T) > \Delta(u|B,T)$ **then**
17:             $B \leftarrow B \cup \{v^*\}$; **break**;
18:         **else continue**;
19:     **if** $i = t + 1$ **then break**;
20: **Return** B;

In Algorithm 4, (1) the BFS traversal has a complexity of $O(|V|+|E|)$; (2) computing $\Delta(N[i][j]|B,T)$ and inserting $N[i][j]$ into H take $O(t \cdot |V| \cdot \log k)$ time; (3) the complexity of selecting initial B and updating $\Delta(x|B,T)$ is $O(k \cdot (|V| \cdot \log k))$; (4) B update requires $O(k \cdot t \cdot |V|)$ time. Therefore, the total time complexity of Algorithm 4 is $O(|V| \cdot t \cdot \log k)$.

IV. TIMIN PROBLEM VARIANTS

In this section, we introduce two variants of the TIMIN problem and present two effective $(1-1/e-\epsilon)$-approximation methods.

A. *Deadline-sensitive* TIMIN *Problem*

In real-world scenarios, understanding the duration for which the spread of negative influence can be effectively controlled is crucial. For example, in infectious disease control campaigns, achieving effective treatments or vaccines can be challenging [33]. Therefore, establishing a maximum deadline is vital. Temporarily isolating crucial patients before this deadline can help control the spread of infectious diseases below a predefined threshold. This deadline provides governments and healthcare systems with sufficient time to explore and implement effective treatments. Motivated by this, we define the Deadline-sensitive TIMIN (DSTIMIN) problem below.

Problem 2. (DSTIMIN). *Given a graph $G(V,E)$, a set of negative seed nodes $S_n \subseteq V$, a budget k, and a threshold α, the* DSTIMIN *problem is to find a node set $B^o \subseteq V - S_n$*

with $|B^o| \leq k$ *and a* maximum timestamp T^o*, such that if* B^o *is blocked, the proportion of the expected negative influence spread by* T^o *remains below* α*. Formally,*

$$(B^o, T^o) = \underset{All\ (B^o,T^o)\ pairs\ satisfy\ C_1 \wedge C_2 \wedge C_3}{\arg\max} T^o,$$

where C_1*:* $B^o \subseteq (V \setminus S_n)$*,* C_2*:* $|B^o| = k$*, and* C_3*:* $\sigma(S_n, G[V \setminus B^o], T^o)/n \leq \alpha$*.*

In a word, the maximum timestamp T^o signifies the tight deadline for people to make preparations against a larger proportion of the expected negative influence spread.

We can extend the TESTIM algorithm to handle the DsTimin problem by employing the binary search technique. Specifically, given a graph $G(V, E)$, let T_m be the maximum accumulative time delay between two vertices in G. Then, we can set the search space of T^o to $[1, T_m]$ and use the binary search to find the maximum timestamp T^o and the corresponding blocking node set B^o such that the proportion of the expected negative influence spread by T^o remains below α. Note that, in each iteration, for the examined timestamp $T \in [1, T_m]$, we employ the TESTIM algorithm to compute the corresponding expected negative influence spread $\sigma(S_n, G[V \setminus B], T)$.

The binary search totally requires $O(\log T_m)$ rounds. In each round, the TESTIM algorithm takes $O(\frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T^o)}{\epsilon^2})$ time. Hence, the DsTimin problem needs $O(\frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T^o)}{\epsilon^2} \cdot \log T_m)$ time to be addressed. Since the TESTIM algorithm offers a $(1 - 1/e - \epsilon)$-approximation guarantee, it can be infered that the binary search based method introduce above has the same approximation.

B. Budget-sensitive Timin Problem

In real applications, excessively suspending user accounts on social platforms can lead to user dissatisfaction [20], [34]. Therefore, when aiming to limit the expected negative influence spread, it is preferable to identify the fewest blocking nodes. For example, during new product launches, social platforms may suspend user accounts that propagate fake news across their networks. However, suspending too many accounts can adversely affect user experience. Thus, it's crucial to suspend the minimum number of accounts necessary to control the spread of fake news until the end of the launches. Motivated by these considerations, we introduce the Budget-sensitive Timin (BsTimin) problem as follows.

Problem 3. (BsTimin). *Given a graph* $G(V, E)$*, a set of negative seeds* S_n*, a deadline* T*, and a threshold* α*, the* BsTimin *problem is to find a* ***minimum*** *set* $\hat{B} \subseteq (V \setminus S_n)$*, such that, by blocking* $\hat{B}$*, the proportion of the expected negative influence spread by the timestamp* T *remains below* α*, i.e.,*

$$\hat{B} = \underset{\hat{B} \subseteq (V \setminus S_n) \wedge \sigma(S_n, G[V \setminus \hat{B}], T)/n \leq \alpha}{\arg\min} |\hat{B}|.$$

Similarly, we can set the search space of $|\hat{B}|$ to $[1, n]$ and employ the binary search to find the $\hat{B}$ with the minimum size. It takes $O(\frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T)}{\epsilon^2} \cdot \log n)$ time overall and has $(1 - 1/e - \epsilon)$ approximation as well.

TABLE II
Dataset Statistics

Dataset	n	m	d_{avg}	d_{max}	Type
EmailCore	1,005	25,571	49.6	544	Directed
Epinions	75,879	508,837	13.4	1,801	Directed
Amazon	334,863	925,872	5.5	549	Undirected
Youtube	1,134,890	2,987,624	5.3	28,754	Undirected
LiveJournal	4,847,571	68,993,773	28.5	20,293	Directed

V. Experiments

We empirically evaluate our proposed algorithms in this section. All methods are implemented in C++ and executed on a Linux server with 2.20 GHz CPU and 128GB of RAM.

A. Experimental Settings

Datasets. In experiments, we employ five real-world networks from SNAP [35]. Table II summarizes the statistics of these networks. Specifically, *EmailCore* is a network of email communication in an European research institution. *Epinions* is a who-trust-whom online social network of a general consumer review site. *Amazon* is a customer purchasing network, where each node corresponds to a product and edges signify co-purchasing relationships of products. *Youtube* is a video-sharing website. *LiveJournal* is an online community that allows users to formally declare their friendships. For each network, we use the *weighted-cascade* model [22] to determine the propagation weight $w_{u,v}$ of each edge, i.e., $w_{u,v} = 1/|N_{in}(v)|$, where $|N_{in}(v)|$ is the number of v's in-neighbors. Following [1], [36], we generate the influence time delay of each edge following the *Poisson* distribution or *Geometric* distribution [13], [27]. Due to space limitation and similar empirical results, we mainly report the results of Poisson distribution.

As with [1], we generate two kinds of negative seed nodes, i.e., *Skewed* and *Random*. The skewed negative seed nodes are sampled from the top-k most influential nodes representing popular/influential users in the network. The random negative seed nodes are randomly sampled from the network.

Algorithms. We test a set of algorithms in experiments.

- **TESTIM** and **NReplacer** (**NR**): our proposed approximate and heuristic algorithms.
- **IMM** [25]: a reverse influence sampling-based method with approximation guarantee, which is used in [3], [9] for influence minimization. We modify it to address the Timin problem by employing Algo. 3 to select blocking nodes.
- **Influential (INF)** [1]: a method that selects blocking nodes in descending order of the expected influence of nodes.
- **Proximity (PRO)** [37]: a method that selects blocking nodes from the out-neighbors of negative seeds, which are highly influenced by the negative seeds.
- **Random (RAN)**: a method that uniformly selects blocking nodes at random.
- **DSIM** and **BSIM**: methods that are proposed in this paper to address the DsTimin and BsTimin problems, respectively.

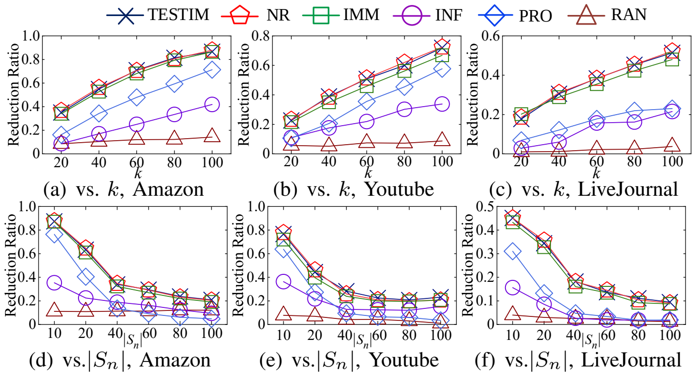


Fig. 4. Reduction Ratio vs. $k/|S_n|$

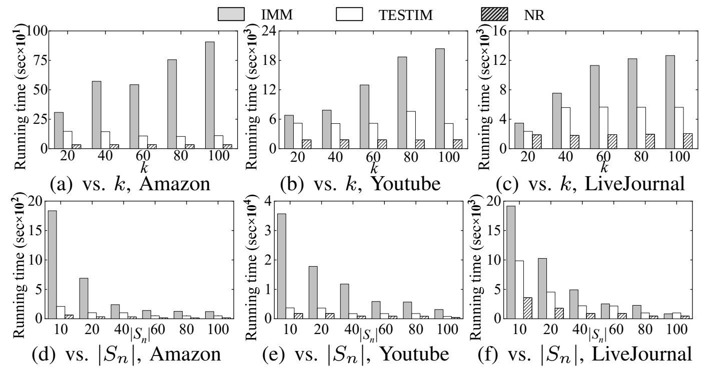


Fig. 5. Running time vs. $k/|S_n|$

Note that TESTIM, IMM, DSIM, and BSIM are approximate algorithms and the other algorithms are heuristic approaches.

Parameters. We evaluate the performance of algorithms on different parameters, including the budget k, the number of negative seeds $|S_n|$, the deadline T, the maximal survival time $\varphi_{\max}$, and sampling error factor ϵ. In each experiment, we vary only one parameter and set the other parameters to their default values, i.e., $k = 50$, $|S_n| = 20$, $T = 32$, and $\varphi_{\max} = 8$. Following [38], [39], we set $\epsilon = 0.2$ for the *EmailCore* and *Epinions*, $\epsilon = 0.3$ for *Amazon*, *Youtube* and *LiveJournal*, and $\delta = 1/n$ for all datasets as the default values. In all experiments, we estimate the reduction ratio of the algorithms by using $2^5 \times 10^5$ T-RR sets, generated independently of the evaluated algorithms.

Evaluation Metrics. We employ the *Reduction Ratio* (R^2) [2], [3] and running time as evaluation metrics in experiments. Specifically, the reduction ratio is the percentage of the saved active nodes, as shown in the following equation. The larger *Reduction Ratio*, the better. In each experiment, we run each method five times and report the average results.

$$R^2(B,T) = \frac{\sigma(S_n, G[V], T) - \sigma(S_n, G[V \setminus B], T)}{\sigma(S_n, G[V], T)}$$

B. *Evaluation of* TIMIN *Algorithms*

In this set of experiments, we evaluate the performance of TIMIN algorithms. Note that, due to space limitation and similar experimental results, we employ the skewed negative seed nodes and report the results of *Amazon*, *Youtube* and *LiveJournal*. Full experimental results can be found in [31].

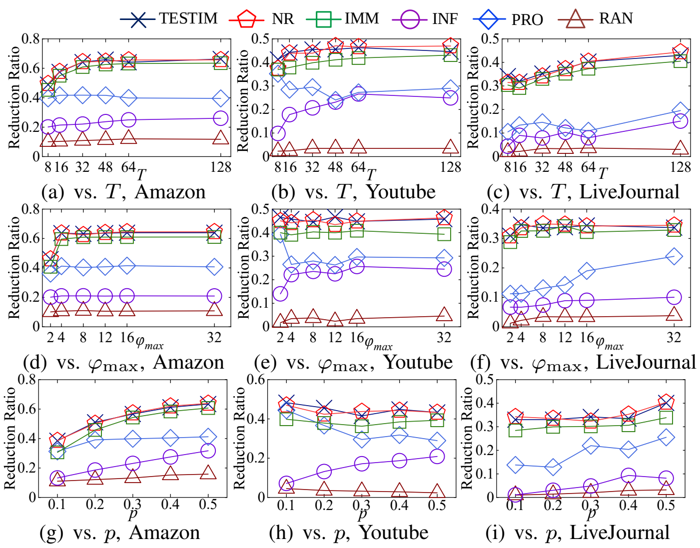
Varying k. We investigate the impact of the number of blocking nodes k by varying k from 20 to 100. The experimental results of the reduction ratio are presented in Figures 4(a)-4(c). It is evident that TESTIM and NR consistently yield the highest reduction ratios across all networks and settings, surpassing IMM, which achieves a slightly lower reduction ratio than TESTIM and NR but significantly outperforms the other heuristics methods. This is because both TESTIM and IMM employ our greedy approach (Algorithm 3) to select the blocking set B, which provides theoretical guarantees. The experimental results also demonstrate the superiority of NR over other heuristic solutions on reduction ratio. Moreover, we evaluate the running time of IMM, TESTIM and NR, as shown in Figures 5(a)-5(c). Notably, TESTIM is significantly faster than IMM across all settings. This is primarily attributed to the early termination of RR-B set generation in TESTIM compared to IMM, resulting in fewer RR-B sets. As discussed in Theorem 6, the time complexity of TESTIM is determined by the generation cost of RR-B sets. Additionally, we can observe that NR significantly outperforms TESTIM and IMM.

Varying $|S_n|$. We study the impact of the number of negative seeds $|S_n|$ and report the reduction ratios in Figures 4(d)-4(f). Notably, TESTIM and NR yield the highest reduction ratios across all settings, highlighting the scalability of our proposed methods. An important observation is that the reduction ratio decreases as $|S_n|$ increases. This is because the number of nodes influenced by negative seeds without considering blocking nodes increases with more negative seeds, while the number of blocking nodes keeps stable. Furthermore, Figures 5(d)-5(f) plot the running time of IMM, TESTIM and NR w.r.t. different $|S_n|$. The results show that NR has the best performance, followed by TESTIM and IMM.

Varying T. We explore the impact of the deadline T on the reduction ratio. As illustrated in Figures 6(a)-6(c), it is observed that as T becomes longer, the reduction ratio for all algorithms initially increases and then stabilizes. It is due to the diminishing marginal influence gain of nodes affected by the negative seeds over time in the diffusion process. Hence, if the deadline is sufficiently long, the number of saved nodes will be stable. Furthermore, since the reduction ratios increase slowly when $T > 32$ in all networks, we set the default deadline T to 32 for all experiments.

Varying φ_{max}. We demonstrate the effect of the influence weight maximal survival time φ_{max} on the reduction ratio, as shown in Figures 6(d)-6(f). It can be observed that as φ_{max} ascends, the reduction ratio initially increases rapidly. When φ_{max} becomes large (from 8 to 32), the reduction ratio keeps stable. This is because, as φ_{max} grows, the influence weights can survive longer, making it easier to successfully activate nodes. In addition, when $\varphi_t > 8$, the probability that the time-delay exceeds φ_{max} is very low under the default *Poisson* distribution, resulting in a stable reduction ratio. Hence, we set φ_{max} to 8 by default in all experiments.

Varying p. We evaluate the effect of the time-delay distribution when varying p. By default, we use *Poisson* distribution



(a) vs. T, Amazon (b) vs. T, Youtube (c) vs. T, LiveJournal

(d) vs. $\varphi_{\max}$, Amazon (e) vs. $\varphi_{\max}$, Youtube (f) vs. $\varphi_{\max}$, LiveJournal

(g) vs. p, Amazon (h) vs. p, Youtube (i) vs. p, LiveJournal

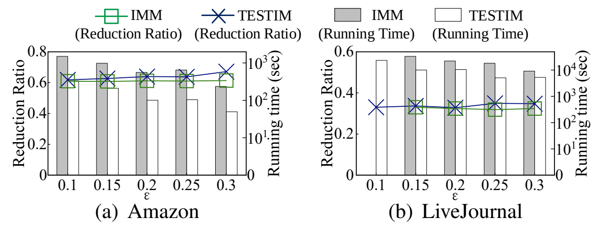
Fig. 6. Reduction Ratio vs. $T/\varphi_{\max}/p$

with a parameter of 1 as the time-delay distribution. In this set of experiments, we utilize the *Geometric* distribution, where $\varphi_t = t$ has a probability of $(1-p)^{t-1} \cdot p$. The reduction ratios of all algorithms are presented in Figures 6(g)-6(i). We observe that the reduction ratios of most algorithms increase as p rises. The reason behind is that the larger p, the faster the negative information propagation, leading to more nodes being activated within T. In addition, under the *Geometric* distribution, TESTIM and NR also have the highest reduction ratio.

Varying ϵ. We explore the effect of ϵ, the sampling error factor in the approximation guarantees achieved by *TRIS* technique. As both TESTIM and IMM utilize *TRIS* technique to select blocking sets and offer theoretical guarantees, we evaluate their reduction ratio (i.e., effectiveness) and running time (i.e., efficiency) by varying ϵ. For each network, we set the number of nodes in the network as the initial number of T-RR sets. Figure 7 reveals that the reduction ratio remains relatively stable when ϵ is varied. This is because the approximation guarantees of TESTIM and IMM indicate the worst-case performance, while their actual performance in real-world scenarios may be empirically robust. Hence, the experimental results highlight the efficiency of *TRIS* technique w.r.t. the variations of ϵ. Moreover, we observe that TESTIM consistently achieves a slightly higher reduction ratio than IMM, indicating that TESTIM's performance remains robust despite changes in ϵ. In addition, the running time decreases as ϵ grows. It is due to early termination in TESTIM (Lines 11) and IMM, resulting in generating fewer RR-B sets. As per Theorem 6, the primary computational cost of TESTIM lies in the generation of RR-B sets, leading to an overall reduction in running time. The reason for IMM is similar to that of TESTIM. Notably, on *LiveJournal* dataset, for $\epsilon = 0.1$, the results of IMM are absent since it does not complete within 24 hours.

C. *Evaluation of* DsTimin *and* BsTimin *Algorithms*

In this section, we investigate the performance of DSIM and BSIM algorithms for the DsTimin and BsTimin problems, respectively. In experiments, we vary the parameter α, i.e., the percentage of negative influence spread. Note that the value of α should be no larger than $\alpha_{\max}$, i.e., the percentage of negative influence spread with no nodes being blocked, as shown by the red numbers in Table III.



(a) Amazon (b) LiveJournal

Fig. 7. Reduction Ratio and Running time vs. ϵ

Table III reports the maximum deadline T^o and the minimum blocking node set $|\hat{B}|$ returned by DSIM and BSIM algorithms, respectively. Obviously, both T^o and $|\hat{B}|$ increase when α becomes larger. This suggests that with larger negative influence spread, the platforms and policymakers take more time and spend higher cost (i.e., blocking the nodes) to control the negative information.

Table IV shows the running time of DSIM and BSIM algorithms. Specifically, the running time of DSIM increases as α grows. It is because a greater α leads to a larger T^o, requiring more time to generate RR-B sets. On the contrary, the running time of BSIM decreases as α grows. The reason behind is that the larger α is, the smaller $|\hat{B}|$ is. Hence, BSIM requires fewer iterations to select the blocking nodes, leading to less running time.

D. *Case Study*

In the case study, we consider the following scenario. During a new product launch campaign, 20 users (highlighted in blue in Figure 8) disseminate fake news about the new products on *Facebook* ($n = 4,039$, $m = 88,234$). We visualize four cases to show how many users are influenced under the different actions taken by the company.

1) Figure 8(a) shows the spread of fake news under the classical LT model without considering time delay, influence decay, and deadline. In addition, no users are blocked. In Figure 8(a), nearly **60%** users are influenced (colored in red), which could significantly affect product sales. Moreover, we can observe that the influenced users are mainly clustered around the sources of fake news.
2) Figure 8(b) considers the same setting of Figure 8(a), with the exception that Figure 8(b) employs TESTIM to designate 100 users as blocking nodes. In this case, the ratio of users influenced by fake news decreases to **32%**. This highlights the effectiveness of our node-blocking strategy in mitigating the negative impact of fake news propagation.
3) Figure 8(c) shows the fake news spread under the TLT model[2], where (i) propagation of fake news terminates

[2]Given the fact that Facebook users open the app an average of 8 times a day [40], we set the average time delay in transmitting fake news from one user to another to 3 hours, and schedule the end date of the product launch 4 days later based on the default parameter settings.

TABLE III

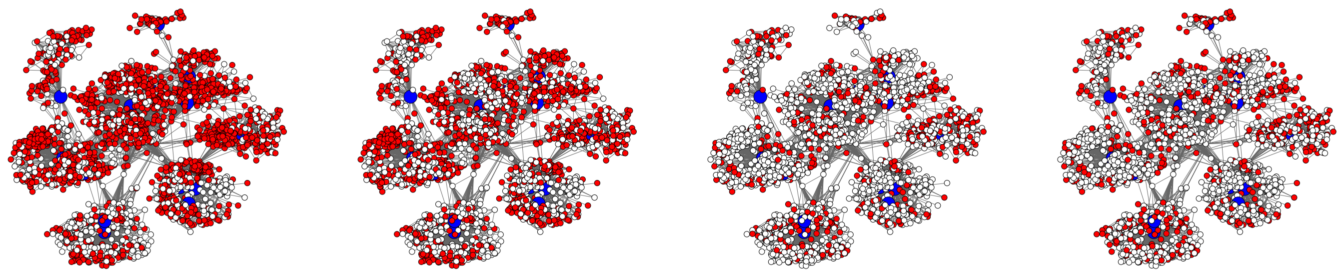
THE T^o AND $|\hat{B}|$ RETURNED BY DSIM AND BSIM, RESPECTIVELY. (THE RED NUMBERS ALONGSIDE SKEWED AND RANDOM ARE $\alpha_{\max}$.)

Dataset	Amazon								Youtube								LiveJournal							
S_n	Skewed (0.41)				Random (0.53)				Skewed (0.69)				Random (0.45)				Skewed (0.50)				Random (0.57)			
α	0.25	0.30	0.35	0.40	0.35	0.40	0.45	0.50	0.50	0.55	0.60	0.65	0.30	0.35	0.40	0.45	0.40	0.44	0.46	0.48	0.50	0.52	0.54	0.56
T^o	6	9	13	27	4	6	13	27	3	4	6	7	3	4	6	13	11	14	26	27	12	13	26	27
$\lvert\hat{B}\rvert$	17,588	12,291	8,333	5,191	17,745	12,713	8,826	5,667	74,329	55,227	36,476	21,666	75,510	46,948	28,072	12,649	124,337	93,591	80,404	68,335	99,882	84,744	71,168	59,139

TABLE IV

THE RUNNING TIME (SEC) OF DSIM AND BSIM VS. α

Dataset	Amazon								Youtube								LiveJournal							
S_n	Skewed				Random				Skewed				Random				Skewed				Random			
α	0.25	0.30	0.35	0.40	0.35	0.40	0.45	0.50	0.50	0.55	0.60	0.65	0.30	0.35	0.40	0.45	0.40	0.44	0.46	0.48	0.50	0.52	0.54	0.56
DSIM	10.75	13.48	14.41	16.11	10.68	13.37	11.87	15.45	20.52	24.33	21.55	27.07	304.77	340.95	363.99	572.76	376.3	441.1	412.5	415.4	369.8	428.3	471.9	499.5
BSIM	7.79	7.75	7.39	7.31	7.48	7.33	7.07	6.96	13.28	12.67	12.35	11.87	195	190.22	189.97	189.48	197.1	195.1	190.5	189.8	217.1	216.4	190.1	202.1



(a) LT model, no blocking node: nearly 60% users are influenced

(b) LT model, TESTIM algorithm: 32% users are influenced

(c) TLT model, TESTIM algorithm: 22% users are influenced

(d) TLT model, INF algorithm: 27% users are influenced

Fig. 8. Visualization of the influenced users by fake news in Facebook. (Red nodes: the influenced users; white nodes: the uninfluenced users; blued nodes: seed nodes to disseminate fake news.)

on the end date of the product launch, (ii) fake news disseminates with time delays, and (iii) the influence power of fake news decays over time. We utilize TESTIM to select 100 users to block. Finally, **22%** of users are influenced.

4) Figure 8(d) shows the spread of fake news under the TLT model when using INF to block 100 users. Finally, **27%** of users are influenced by the fake news. This significantly illustrates the superior performance of TESTIM. Due to space limitations, we report the results of INF since other methods yield similar experimental results.

In summary, the TLT model simulates a more realistic propagation process by considering temporal factors such as time delay, influence decay, and deadline, compared to the classical LT model. Furthermore, TESTIM exhibits the highest effectiveness in mitigating the spread of fake news compared to other methods.

VI. RELATED WORK

Influence Maximization. The problem of Influence Maximization (IM) was initially formulated as a discrete optimization problem by Kempe et al. [22]. Subsequently, substantial follow-up research has focused on developing efficient and scalable solutions for the IM problem [25], [26], [41]–[43]. Many of these approaches rely on reverse influence sampling methods [24]. Meanwhile, researchers have explored various variants of IM [39], [44]–[46]. In this paper, we focus on the influence minimization problem.

Influence Minimization. The Influence Minimization (IMIN) problem aims to minimize the spread of negative information/disease in social networks, as reviewed in [47], [48]. Existing IMIN solutions fall into two categories [49]. **(1)** Clarification-based methods [1], [2], [6], [7], [12], [50] aim to minimize the spread of negative information by strategically disseminating positive information. [2] and [7] used login events to simulate information diffusion delays, thus minimizing rumor influence before a specific deadline. [1] considered differential propagation rates between truth and misinformation and user reaction times when mitigating misinformation spread. **(2)** Blocking-based methods [3], [8]–[10], [14], [20], [51], [52] include blocking nodes [3], [10], [20], [51], [52] or edges [14], [21], [53]–[55]. For node blocking, [10] proposed a novel graph sampling technique that incorporates the dominator tree structure to select blocking nodes, while [3] adaptively selected blocking nodes based on real-time observations of negative influence spread. For edge blocking, [54] developed heuristic methods for edge blocking under a simpler variant of the LT model. [21] provided theoretical guarantee algorithms to add/delete a small set of edges using live-edge graphs. *However, existing blocking-based influence minimization studies often overlook temporal aspects, which distinguishes our work from the prior research.*

VII. CONCLUSION

In this paper, we study the TIMIN problem where we aim to minimize the negative influence in social networks by temporarily blocking key nodes w.r.t. a deadline. We introduce a novel TLT model that simulates the diffusion of negative information, considering temporal factors. Given the NP-hardness of the TIMIN problem, we develop **(1)** Timin-Greedy, a greedy algorithm with a $(1-1/e)$ approximation, **(2)** TESTIM, a scalable algorithm with a $(1-1/e-\epsilon)$ approximation, and **(3)** NReplacer, an efficient heuristic algorithm. Moreover, we introduce two variants of the TIMIN problem that incorporate time and budget constraints, respectively. Through extensive experiments and a case study, we demonstrate the effectiveness, efficiency, and scalability of our algorithms.

References

[1] M. Simpson, F. Hashemi, and L. V. Lakshmanan, "Misinformation mitigation under differential propagation rates and temporal penalties," *Proceedings of the VLDB Endowment*, vol. 15, no. 10, pp. 2216–2229, 2022.

[2] C. Song, W. Hsu, and M. L. Lee, "Temporal influence blocking: Minimizing the effect of misinformation in social networks," in *2017 IEEE 33rd international conference on data engineering (ICDE)*, 2017, pp. 847–858.

[3] Q. Shi, C. Wang, D. Ye, J. Chen, Y. Feng, and C. Chen, "Adaptive influence blocking: Minimizing the negative spread by observation-based policies," in *2019 IEEE 35th international conference on data engineering (ICDE)*, 2019, pp. 1502–1513.

[4] Y. Gao, X. Wang, X. He, H. Feng, and Y. Zhang, "Rumor detection with self-supervised learning on texts and social graph," *Frontiers of Computer Science*, vol. 17, no. 4, p. 174611, 2023.

[5] B. Guo, Y. Ding, Y. Sun, S. Ma, K. Li, and Z. Yu, "The mass, fake news, and cognition security," *Frontiers of Computer Science*, vol. 15, pp. 1–13, 2021.

[6] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 665–674.

[7] M. A. Manouchehri, M. S. Helfroush, and H. Danyali, "Temporal rumor blocking in online social networks: A sampling-based approach," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 7, pp. 4578–4588, 2021.

[8] H. T. Nguyen, A. Cano, T. Vu, and T. N. Dinh, "Blocking self-avoiding walks stops cyber-epidemics: a scalable gpu-based approach," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 32, no. 7, pp. 1263–1275, 2019.

[9] L. Sun, X. Rui, and W. Chen, "Scalable adversarial attack algorithms on influence maximization," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 760–768.

[10] J. Xie, F. Zhang, K. Wang, X. Lin, and W. Zhang, "Minimizing the influence of misinformation via vertex blocking," in *2023 IEEE 39th international conference on data engineering (ICDE)*, 2023, pp. 789–801.

[11] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[12] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model," in *Proceedings of the 2012 siam international conference on data mining*. SIAM, 2012, pp. 463–474.

[13] W. Chen, W. Lu, and N. Zhang, "Time-critical influence maximization in social networks with time-delayed diffusion process," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012, pp. 591–598.

[14] C. V. Pham, H. M. Dinh, H. D. Nguyen, H. T. Dang, and H. X. Hoang, "Limiting the spread of epidemics within time constraint on online social networks," in *Proceedings of the 8th International Symposium on Information and Communication Technology*, 2017, pp. 262–269.

[15] B. Chandramouli, J. Goldstein, and S. Duan, "Temporal analytics on big data for web advertising," in *2012 IEEE 28th international conference on data engineering (ICDE)*, 2012, pp. 90–101.

[16] customerAI, "How often active users access each social media platform." 2018. [Online]. Available: https://customers.ai/articles/number-of-times-users-access-social-media-platform

[17] D. Lewis, "Why the who took two years to say covid is airborne," *Nature*, vol. 604, no. 7904, pp. 26–31, 2022.

[18] N. Ohsaka, Y. Yamaguchi, N. Kakimura, and K.-i. Kawarabayashi, "Maximizing time-decaying influence in social networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 132–147.

[19] O. Puhach, B. Meyer, and I. Eckerle, "Sars-cov-2 viral load and shedding kinetics," *Nature Reviews Microbiology*, vol. 21, no. 3, pp. 147–161, 2023.

[20] B. Wang, G. Chen, L. Fu, L. Song, and X. Wang, "Drimux: Dynamic rumor influence minimization with user experience in social networks," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 29, no. 10, pp. 2168–2181, 2017.

[21] E. B. Khalil, B. Dilkina, and L. Song, "Scalable diffusion-aware optimization of network topology," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1226–1235.

[22] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 137–146.

[23] D. Chelkak and S. Smirnov, "Universality in the 2d ising model and conformal invariance of fermionic observables," *Inventiones mathematicae*, vol. 189, pp. 515–580, 2012.

[24] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 946–957.

[25] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1539–1554.

[26] J. Tang, X. Tang, X. Xiao, and J. Yuan, "Online processing algorithms for influence maximization," in *Proceedings of the 2018 ACM SIGMOD International Conference on Management of Data*, 2018, pp. 991–1005.

[27] B. Liu, G. Cong, Y. Zeng, D. Xu, and Y. M. Chee, "Influence spreading path and its application to the time constrained social influence maximization problem and beyond," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 26, no. 8, pp. 1904–1917, 2013.

[28] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the third ACM international conference on Web search and data mining*, 2010, pp. 241–250.

[29] M. Gomez Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," in *28th International Conference on Machine Learning (ICML 2011)*. International Machine Learning Society, 2011, pp. 561–568.

[30] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 4, pp. 1–37, 2012.

[31] X. Chang, J. Fu, Q. Liu, Y. Gao, and B. Zheng, "Time-aware influence minimization via blocking social networks," 2024. [Online]. Available: https://github.com/ZJU-DAILY/TIMIN

[32] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[33] C. News, "China approves its first mrna covid-19 vaccine." 2023. [Online]. Available: https://edition.cnn.com/2023/03/23/tech/china-covid-mrna-vaccine-hnk-intl/index.html

[34] A. C. Miller, "# dictatorerdogan: How social media bans trigger backlash," *Political Communication*, vol. 39, no. 6, pp. 801–825, 2022.

[35] J. Leskovec and A. Krevl, "Snap datasets: Stanford large network dataset collection," 2014, http://snap.stanford.edu/data.

[36] X. Miao, H. Peng, K. Chen, Y. Peng, Y. Gao, and J. Yin, "Maximizing time-aware welfare for mixed items," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 2022, pp. 1044–1057.

[37] A. Tong, D.-Z. Du, and W. Wu, "On misinformation containment in online social networks," in *Advances in neural information processing systems*, 2018, pp. 339–349.

[38] C. Aslay, F. B. L. V. Lakshmanan, and W. Lu, "Revenue maximization in incentivized social advertising," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, 2017.

[39] K. Han, B. Wu, J. Tang, S. Cui, C. Aslay, and L. V. Lakshmanan, "Efficient and effective algorithms for revenue maximization in social advertising," in *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*, 2021, pp. 671–684.

[40] Omniocre, "80+ facebook statistics you need to know in 2024." 2023. [Online]. Available: https://www.omnicoreagency.com/facebook-statistics/

[41] K. Huang, S. Wang, G. Bevilacqua, X. Xiao, and L. V. Lakshmanan, "Revisiting the stop-and-stare algorithms for influence maximization," *Proceedings of the VLDB Endowment*, vol. 10, no. 9, pp. 913–924, 2017.

[42] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 2014, pp. 75–86.

[43] Q. Guo, S. Wang, Z. Wei, W. Lin, and J. Tang, "Influence maximization revisited: efficient sampling with bound tightened," *ACM Transactions on Database Systems (TODS)*, vol. 47, no. 3, pp. 1–45, 2022.

[44] P. Banerjee, W. Chen, and L. V. Lakshmanan, "Maximizing welfare in social networks under a utility driven influence diffusion model," in *Proceedings of the 2019 ACM SIGMOD International Conference on Management of Data*, 2019, pp. 1078–1095.
[45] J. Tang, X. Tang, and J. Yuan, "Profit maximization for viral marketing in online social networks: Algorithms and analysis," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 30, no. 6, pp. 1095–1108, 2017.
[46] Y. Zhang, Y. Li, Z. Bao, B. Zheng, and H. Jagadish, "Minimizing the regret of an influence provider," in *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*, 2021, pp. 2115–2127.
[47] A. Zareie and R. Sakellariou, "Minimizing the spread of misinformation in online social networks: A survey," *Journal of Network and Computer Applications*, vol. 186, p. 103094, 2021.
[48] L. Sun, Y. Rao, L. Wu, X. Zhang, Y. Lan, and A. Nazir, "Fighting false information from propagation process: A survey," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–38, 2023.
[49] S. Thirumuruganathan, M. Simpson, and L. V. Lakshmanan, "To intervene or not to intervene: Cost based intervention for combating fake news," in *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*, 2021, pp. 2300–2309.
[50] Q. Fang, X. Chen, Q. Nong, Z. Zhang, Y. Cao, Y. Feng, T. Sun, S. Gong, and D. Du, "General rumor blocking: An efficient random algorithm with martingale approach," *Theoretical Computer Science*, vol. 803, pp. 82–93, 2020.
[51] J. Xie, "Hindering influence diffusion of community," in *Proceedings of the 2022 ACM SIGMOD International Conference on Management of Data*, 2022, pp. 2518–2520.
[52] C. Chen, H. Tong, B. A. Prakash, C. E. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau, "Node immunization on large graphs: Theory and algorithms," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 28, no. 1, pp. 113–126, 2015.
[53] M. Kimura, K. Saito, and H. Motoda, "Minimizing the spread of contamination by blocking links in a network." in *AAAI*, vol. 8, 2008, pp. 1175–1180.
[54] C. J. Kuhlman, G. Tuli, S. Swarup, M. V. Marathe, and S. Ravi, "Blocking simple and complex contagion by edge removal," in *2013 IEEE International Conference on Data Mining (ICDM)*, 2013, pp. 399–408.
[55] R. Yan, Y. Li, W. Wu, D. Li, and Y. Wang, "Rumor blocking through online link deletion on social networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 2, pp. 1–26, 2019.
[56] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—i," *Mathematical programming*, vol. 14, pp. 265–294, 1978.
[57] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *2010 IEEE International Conference on Data Mining (ICDM)*, 2010, pp. 88–97.
[58] T. Jin, Y. Yang, R. Yang, J. Shi, K. Huang, and X. Xiao, "Unconstrained submodular maximization with modular costs: Tight approximation and application to profit maximization," *Proceedings of the VLDB Endowment*, vol. 14, no. 10, pp. 1756–1768, 2021.

Appendix

A. Proof of Lemma 3

Proof. Given the monotonicity and supermodularity of the TIMIN problem established in §II-C, the assurance of approximation follows directly from [56]. □

B. Proof of Proposition 4

Proof. Computing the exact influence spread $\sigma(S)$ for any set S under the classical LT model is proven to be $\#\mathbf{P}$-hard [57]. The TLT model is a special case of the LT model, thus, the exact calculation of $\sigma(S_n, G[V], T)$ under the TLT model is confirmed to be $\#\mathbf{P}$-hard. □

C. Proof of Lemma 4

Proof. Before proving Lemma 4, we first introduce *Chernoff Inequalities* [32] in Lemma 5 as follows.

Lemma 5. (Chernoff Inequalities [32]). *Let X be the sum of k i.i.d. random variables sampled from a distribution on $[0,1]$ and μ be the mean. Then, for any $\lambda > 0$,*

$$\Pr[X - k\mu \geq \lambda \cdot k\mu] \leq \exp\left(-\frac{\lambda^2}{2+\lambda} k\mu\right) \tag{12}$$

$$\Pr[X - k\mu \leq -\lambda \cdot k\mu] \leq \exp\left(-\frac{\lambda^2}{2} k\mu\right) \tag{13}$$

Given any solution B to the TIM problem and the optimal solution B^*, and any set $\mathcal{R}_b$ of RR-B sets, we extend Lemma 5 and introduce the following concentration bounds:

$$\begin{aligned}&\Pr[g^{\mathcal{R}_{b2}}(B,T) - R(B,T) \geq \epsilon_1 \cdot R(B,T)] \\ &\leq \exp\left(-\frac{\epsilon_1^2}{2+\epsilon_1} \frac{|\mathcal{R}_{b2}|}{f^{\mathcal{R}_{t2}}(S_n, G[V], T)} R(B,T)\right)\end{aligned} \tag{14}$$

$$\begin{aligned}&\Pr[g^{\mathcal{R}_{b1}}(B^*,T) - R(B^*,T) \leq -\epsilon_2 \cdot R(B^*,T)] \\ &\geq \exp\left(-\frac{\epsilon_2^2}{2} \frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n, G[V], T)} R(B^*,T)\right)\end{aligned} \tag{15}$$

where $|\mathcal{R}_b|$ represents the number of RR-B sets. Based on this, in the i-th iteration, let Ω_{1i} denote the event that Eq. (9) holds, and Ω_{2i} denote the event that Eq. (10) holds. We set ϵ' and $\bar{\epsilon}$ as the solutions to Eq. (14) and Eq. (15) respectively, thus we have following equations:

$$\exp\left(-\frac{(\epsilon')^2}{2+\epsilon'} \frac{|\mathcal{R}_{b2}|}{f^{\mathcal{R}_{t2}}(S_n, G[V], T)} R(B,T)\right) = \frac{\delta}{5i^2}. \tag{16}$$

$$\exp\left(-\frac{(\bar{\epsilon})^2}{2} \frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n, G[V], T)} R(B^*,T)\right) = \frac{\delta}{5i^2}. \tag{17}$$

Then, we have $\Pr[\Omega_{i1}] \geq 1 - \delta/(5i^2)$ and $\Pr[\Omega_{i2}] \geq 1 - \delta/(5i^2)$.

$$\begin{aligned}\frac{(\epsilon')^2}{(2+\epsilon')(1+\epsilon')} &= \frac{f^{\mathcal{R}_{t2}}(S_n, G[V], T) \ln\left(5i^2/\delta\right)}{|\mathcal{R}_{b2}| \cdot (1+\epsilon') \cdot R(B,T)} \\ &\leq \frac{n \ln\left(5i^2/\delta\right)}{|\mathcal{R}_{b2}| \cdot g^{\mathcal{R}_{b2}}(B,T)}\end{aligned} \tag{18}$$

$$\begin{aligned}\frac{\bar{\epsilon}^2}{2(1+\epsilon_1)} &= \frac{f^{\mathcal{R}_{t1}}(S_n, G[V], T)}{|\mathcal{R}_{b1}| \cdot (1+\epsilon_1) \cdot R(B^*,T)} \ln\left(\frac{5i^2}{\delta}\right) \\ &\leq \frac{n}{|\mathcal{R}_{b1}| \cdot (1+\epsilon_1) \cdot R(B,T)} \ln\left(\frac{5i^2}{\delta}\right) \\ &\leq \frac{n}{|\mathcal{R}_{b1}| \cdot g^{\mathcal{R}_{b2}}(B,T)} \ln\left(\frac{5i^2}{\delta}\right)\end{aligned} \tag{19}$$

We have $\Pr[\Omega_{i1}] \geq 1 - \delta/(5i^2)$, $\Pr[\Omega_{i2}|\Omega_{i1}] \geq 1 - \delta/(5i^2)$. Thus, $\Pr[\Omega_{i2} \cap \Omega_{i1}] = \Pr[\Omega_{i2}|\Omega_{i1}] \cdot \Pr[\Omega_{i1}] \geq 1 - 2\delta/(5i^2)$. For all iterations, we have

$$\begin{aligned}\Pr\left[\bigcap_{i=1}^{\infty} \Omega_{1i} \bigcap_{i=1}^{\infty} \Omega_{2i}\right] &\geq \prod_{i=1}^{\infty} \Pr\left[\Omega_{1i} \cap \Omega_{2i}\right] \geq \prod_{i=1}^{\infty} \left(1 - \frac{2\delta}{5i^2}\right) \\ &\geq 1 - \sum_{i=1}^{\infty} \frac{2\delta}{5i^2} \geq 1 - \frac{\pi^2 \delta}{15} \geq 1 - \frac{2\delta}{3}.\end{aligned} \tag{20}$$

The details of proof are similar in spirit to those in [58]. □

With the above conclusions, we further prove Theorem 5.

D. Proof of Theorem 5

Proof. We consider two cases that depend on whether Line 11 in Algorithm 2 is satisfied.

Case 1: Line 11 is satisfied. Then in the last iteration, we have

$$\lambda \leq \frac{1-1/e}{1-1/e-\epsilon} \cdot \frac{1-\epsilon_2}{1+\epsilon_1} \tag{21}$$

where $\epsilon_1, \epsilon_2 \in (0,1)$ and $\lambda > 0$. Suppose that both Eq. (9) and Eq. (10) hold. Then,

$$\begin{aligned}g^{\mathcal{R}_{b1}}(B,T) &\geq (1-1/e) g^{\mathcal{R}_{b1}}(B^*,T) \\ &\geq (1-1/e)(1-\epsilon_2) R(B^*,T)\end{aligned} \tag{22}$$

where the first inequality is due to Lemma 3, and the second inequality is due to Eq. (10). Afterwards, via Eq. (9) we have:

$$g^{\mathcal{R}_{b1}}(B,T) = \lambda g^{\mathcal{R}_{b2}}(B,T) \leq \lambda(1+\epsilon_1) R(B,T) \tag{23}$$

Finally, we have

$$\begin{aligned}R(B,T) &\geq \frac{1}{\lambda(1+\epsilon_1)} g^{\mathcal{R}_{b1}}(B,T) \\ &\geq \frac{(1-1/e)(1-\epsilon_2)}{\lambda(1+\epsilon_1)} R(B^*,T) \\ &\geq (1-1/e-\epsilon) R(B^*,T)\end{aligned} \tag{24}$$

where the first inequality is from Eq. (23), the second inequality is from Eq. (22), and the third inequality is from Eq. (21). By Lemma 4, when Line 11 in Algorithm 2 is satisfied, with probability at least $1 - \frac{2\delta}{3}$, Theorem 5 holds.

Case 2: Line 11 is not satisfied. Then we have

$$|\mathcal{R}_{b1}| = (8+2\epsilon)(1+\epsilon_1) n \frac{\ln\frac{6}{\delta} + n\ln 2}{\epsilon^2 \cdot g^{\mathcal{R}_{b2}}(B,T)}$$

when Algorithm 2 terminates. Note that when $\bigcap_i \Omega_{1i}$ occurs, it implies that $g^{\mathcal{R}_{b2}}(B,T) \leq (1+\epsilon_1) R(B,T) \leq (1+\epsilon_1) R(B^*,T)$. Then

$$|\mathcal{R}_{b1}| \geq (8+2\epsilon) n \frac{\ln\frac{6}{\delta} + n\ln 2}{\epsilon^2 \cdot R(B^*,T)}$$

when Algorithm 2 terminates. Then by Lemma 5, let $x = \frac{\epsilon R(B^*,T)}{2R(B,T)}$ for any $B \subseteq (V \setminus S_n)$,

$$
\begin{aligned}
&\Pr[g^{\mathcal{R}_{b1}}R(B,T) - R(B,T) \geq \frac{\epsilon}{2} \cdot R(B^*,T)] \\
&\leq \exp\left(-\frac{x^2}{2+x}\frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n, G[V], T)}R(B,T)\right) \\
&\leq \exp\left(-\frac{x^2}{2+x}\frac{|\mathcal{R}_{b1}|}{n}R(B,T)\right) \\
&\leq \exp\left(-\frac{\epsilon^2}{8+2\epsilon}\frac{|\mathcal{R}_{b1}|}{n}R(B^*,T)\right) \\
&\leq \frac{\delta}{6 \cdot 2^n},
\end{aligned}
$$

where the second inequality is due to the fact that if $R(B,T) = R(B^*,T)$, the right side of the first inequality achieves its maximum. Similarly, we also have $\Pr[g^{\mathcal{R}_{b1}}R(B,T) - R(B,T) \leq (-\frac{\epsilon}{2}) \cdot R(B^*,T)] \leq \frac{\delta}{6 \cdot 2^n}$. Thus, we have $\Pr[|g^{\mathcal{R}_{b1}}R(B,T) - R(B,T)| \leq \frac{\epsilon}{2} \cdot R(B^*,T), \forall B \subseteq (V \setminus S_n)] \geq 1 - \frac{\delta}{3}$. When $|g^{\mathcal{R}_{b1}}R(B,T) - R(B,T)| \leq \frac{\epsilon}{2}R(B^*,T)$ for all $B \subset V$, we have

$$
g^{\mathcal{R}_{b1}}R(B^*,T) \geq (1 - \frac{\epsilon}{2})R(B^*,T) \ , \tag{25}
$$

$$
g^{\mathcal{R}_{b1}}R(B,T) \leq R(B,T) + \frac{\epsilon}{2}R(B^*,T). \tag{26}
$$

Based on the above results, when the event $\bigcap_i \Omega_{1i}$ occurs, we have

$$
\begin{aligned}
R(B,T) &\geq g^{\mathcal{R}_{b1}}R(B,T) - \frac{\epsilon}{2}R(B^*,T) \\
&\geq (1-1/e)g^{\mathcal{R}_{b1}}R(B^*,T) - \frac{\epsilon}{2}R(B^*,T) \\
&\geq (1-1/e)(1-\frac{\epsilon}{2})R(B^*,T) - \frac{\epsilon}{2}R(B^*,T) \\
&= (1-1/e-\frac{\epsilon}{2}+\frac{\epsilon}{2e})R(B^*,T) - \frac{\epsilon}{2}R(B^*,T) \\
&= (1-1/e-\epsilon+\frac{\epsilon}{2e})R(B^*,T) \\
&\geq (1-1/e-\epsilon) \cdot R(B^*,T)
\end{aligned}
$$

According to Eq. (20), the event $\bigcap_i \Omega_{1i}$ happens with probability at least $1 - \frac{2\delta}{3}$. Hence, when Line 11 is not satisfied, with probability at least $1 - \frac{2\delta}{3} - \frac{\delta}{3} \geq 1 - \delta$, we have

$$
R(B,T) \geq (1-1/e-\epsilon) \cdot R(B^*,T)
$$

Finally, we combine **Case 1** and **Case 2**, the Theorem 5 is demonstrated. □

E. PROOF OF THEOREM 6

Proof. The time complexity of Algorithm 2 is dominated by the cost of T-RR set generation. In Algorithm 2, the total number of T-RR set generated is at most $O(\frac{n \cdot \max\{\ln\frac{1}{\delta}, n\}}{\epsilon^2})$ [58]. The expected time of generating a random T-RR set is bounded by $\frac{m}{n} \cdot R(\{v^*\}, T)$ [42]. Hence, the expected time complexity of Algorithm 2 is $O(\frac{\max\{\ln\frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T)}{\epsilon^2})$. □

F. Full experiment results

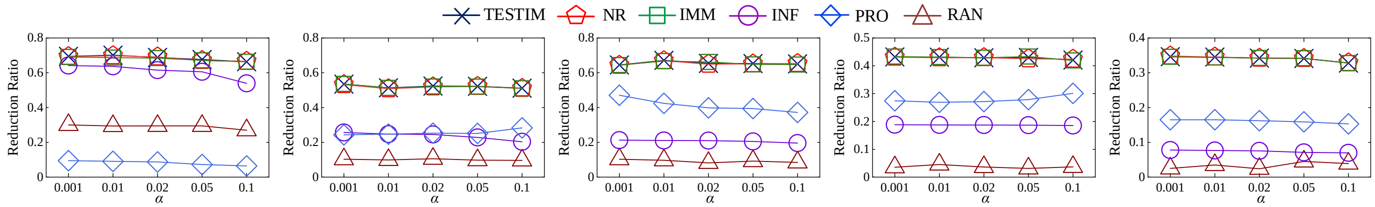
This section shows the complete experiment results that are omitted in Section V due to space constraints.

Varying k. We investigate the impact of the number of blocking nodes k on six algorithms by varying k from 20 to 100. The experimental results on the reduction ratio are presented in Figure 11 for skewed negative seeds, and Figure 12 for random negative seeds. It is evident that TESTIM and NR consistently yield the highest reduction ratios across all networks and settings, surpassing IMM, which achieves a slightly lower reduction ratio than TESTIM and NR but significantly outperforms the other heuristics methods. This is because both TESTIM and IMM employ our greedy approach (Algorithm 3) to select the blocking set B, which provides theoretical guarantees. The experimental results also demonstrate the superiority of NR over other heuristic solutions on reduction ratio. Moreover, we evaluate the running time of IMM, TESTIM and NR, as shown in Figures 15 and 16. Notably, TESTIM is significantly faster than IMM across all settings. This is primarily attributed to the early termination of RR-B set generation in TESTIM compared to IMM, resulting in fewer RR-B sets. As discussed in Theorem 6, the time complexity of TESTIM is determined by the generation cost of RR-B sets. Additionally, we can observe that NR significantly outperforms TESTIM and IMM.

Varying $|S_n|$. We study the impact of the number of negative seeds $|S_n|$ and report the reduction ratios in Figures 13 and 14. Notably, TESTIM and NR yield the highest reduction ratios across all settings, highlighting the scalability of our proposed methods. An important observation is that the reduction ratio decreases as $|S_n|$ increases. This is because the number of nodes influenced by negative seeds without considering blocking nodes increases with more negative seeds, while the number of blocking nodes keeps stable. Furthermore, Figures 17 and 18 plot the running time of IMM, TESTIM and NR under various number of negative seeds $|S_n|$. The results show that NR has the best performance, followed by TESTIM and IMM.

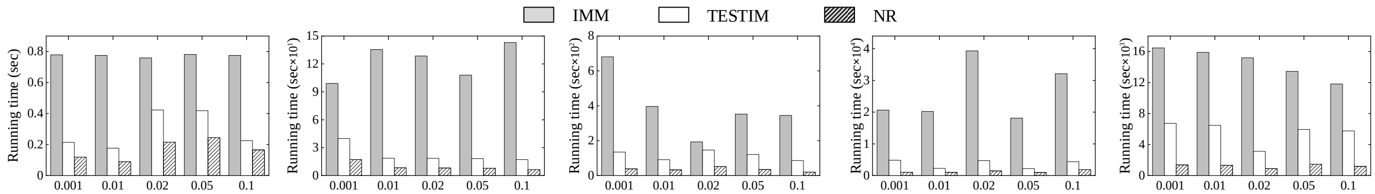
Varying T. We explore the impact of the deadline parameter T on the reduction ratio. As illustrated in Figure 19, it is observed that as T becomes longer, the reduction ratio for all algorithms initially increases and then stabilizes. It is due to the diminishing marginal influence gain of nodes affected by the negative seeds over time in the diffusion process. Hence, if the deadline is sufficiently long, the number of saved nodes will be stable. Furthermore, since the reduction ratios increase slowly when $T > 32$ in all networks, we set the default deadline T to 32 for all experiments.

Varying φ_{max}. We demonstrate the influence of the maximal survival time of influence weight on each edge, denoted as φ_{max}, on the reduction ratio, as shown in Figure 20. It can be observed that as φ_{max} ascends, the reduction ratio initially increases rapidly. When φ_{max} becomes large (from 8 to 32), the reduction ratio keeps stable. This is because, as φ_{max} grows, the influence weights can survive longer, making it easier to successfully activate nodes. In addition, when $\varphi_t > 8$, the probability that the time-delay exceeds φ_{max} is very low under the default *Poisson* distribution, resulting in a stable reduction ratio. Hence, we set φ_{max} to 8 by default in all experiments.



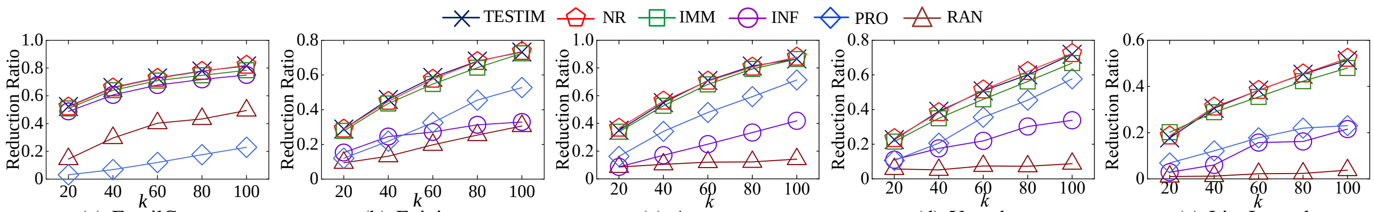
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 9. Reduction Ratio vs. α (skewed negative seeds)



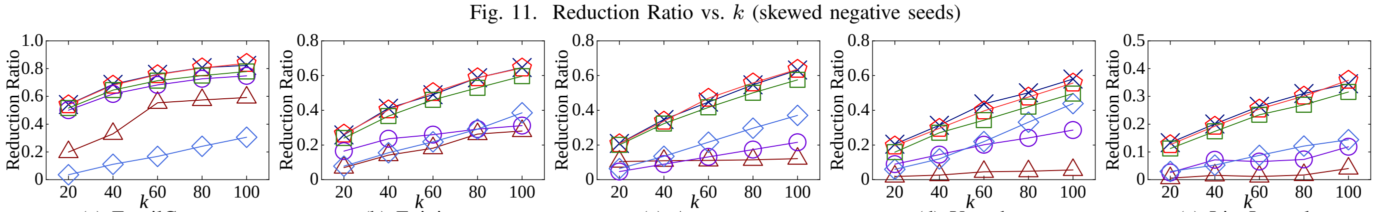
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 10. Running time vs. α (skewed negative seeds)



(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 11. Reduction Ratio vs. k (skewed negative seeds)



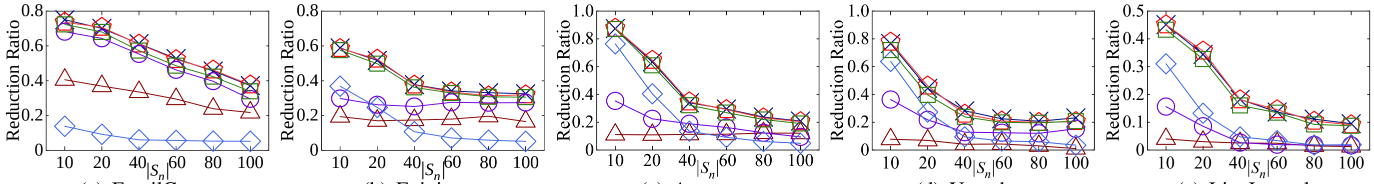
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 12. Reduction Ratio vs. k (random negative seeds)

Varying p. We evaluate the effect of the time-delay distribution when varying p. By default, we use *Poisson* distribution with a parameter of 1 as the time-delay distribution. In this set of experiments, we utilize the *Geometric* distribution, where $\varphi_t = t$ has a probability of $(1-p)^{t-1} \cdot p$. The reduction ratios of all algorithms are presented in Figure 21. The reduction ratios of most algorithms increase as p rises. The reason behind is that the larger p, the faster the negative information propagation, leading to more nodes being activated within T. In addition, under the *Geometric* distribution, TESTIM and NR also have the highest reduction ratio.

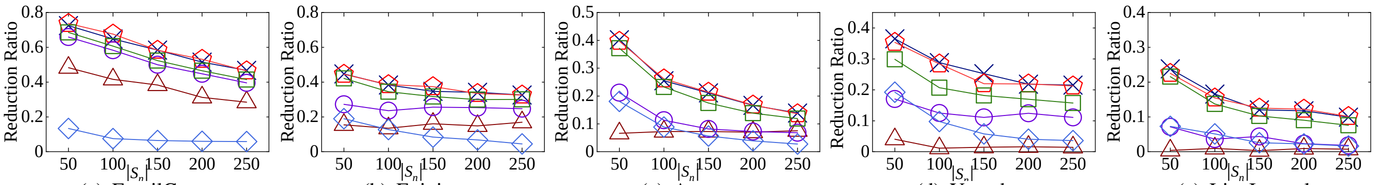
Varying ϵ. We explore the effect of ϵ, the sampling error factor in the approximation guarantees achieved by *TRIS* technique. As both TESTIM and IMM utilize *TRIS* technique to select blocking sets and offer theoretical guarantees, we compare their reduction ratio (i.e., effectiveness) and running time (i.e., efficiency) by varying ϵ. For each network, we set the number of nodes in the network as the initial number of T-RR sets. Figures 22 and 23 reveal that the reduction ratio remains relatively consistent when ϵ is varied. This is because the approximation guarantees of TESTIM and IMM indicate the worst-case performance, while their actual performance in real-world scenarios may be empirically robust. Hence, the experimental results highlight the efficiency of *TRIS* technique w.r.t. the variations of ϵ. Moreover, we observe that TESTIM consistently achieves a slightly higher reduction ratio than IMM, indicating that TESTIM's performance remains robust despite changes in ϵ. In addition, the running time decreases as ϵ grows. It is due to early termination in TESTIM (Lines 11) and IMM, resulting in generating fewer RR-B sets. As per Theorem 6, the primary computational cost of TESTIM lies in the generation of RR-B sets, leading to an overall reduction in running time. The reason for IMM is similar to that of TESTIM. Notably, on *LiveJournal*, for $\epsilon = 0.1$, the results of IMM are absent since it does not complete within 24 hours.

Varying α. We evaluate the impact of decay parameter α on reduction ratio and running time, specifically, the decay function we use is $f(\varphi_{u,v}) = \varphi_{u,v}^{-\alpha}$. As shown in Figure 9, as α increases, the reduction ratio slightly decreases. This is because as α increases, the marginal loss brought by block nodes is more obvious than the reduction in negative influence. In all datasets, TESTIM and NR are superior to other heuristic methods on the reduction ratio, IMM has similar reduction ratio to TESTIM and NR since it uses the same sampling method to get the final result. Furthermore, as demonstrated in Figure 10, NR is significantly faster than TESTIM and IMM among all datasets, on average, it exhibits $8\times$ faster than IMM.



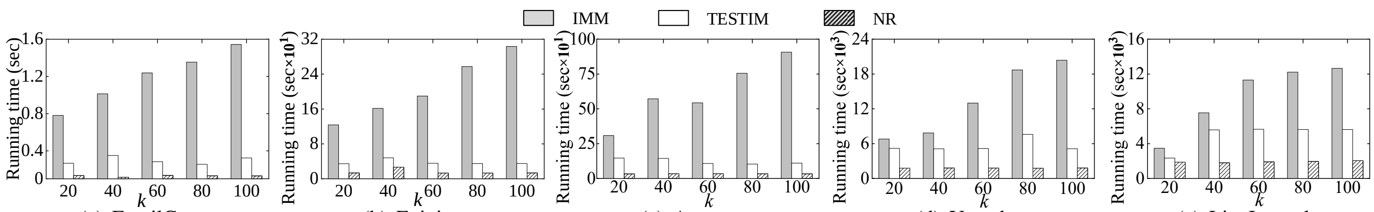
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 13. Reduction Ratio vs. $|S_n|$ (skewed negative seeds)



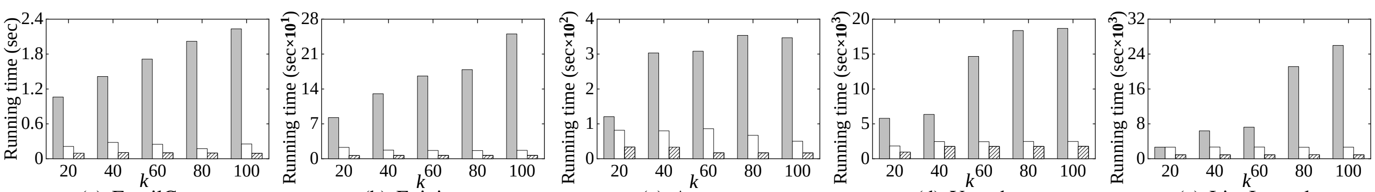
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 14. Reduction Ratio vs. $|S_n|$ (random negative seeds)



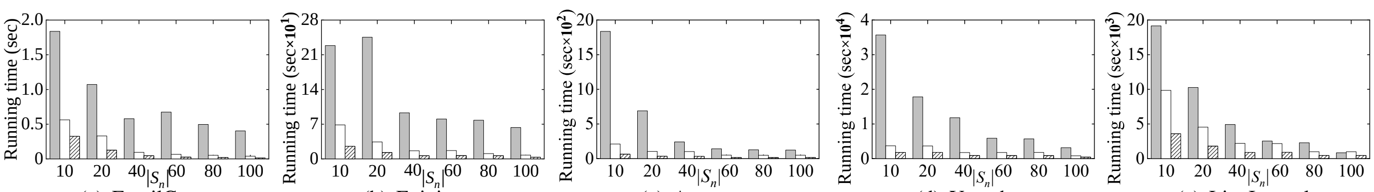
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 15. Running time vs. k (skewed negative seeds)



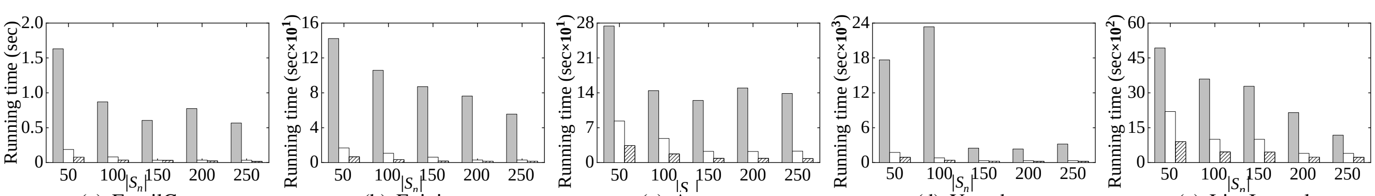
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 16. Running time vs. k (random negative seeds)



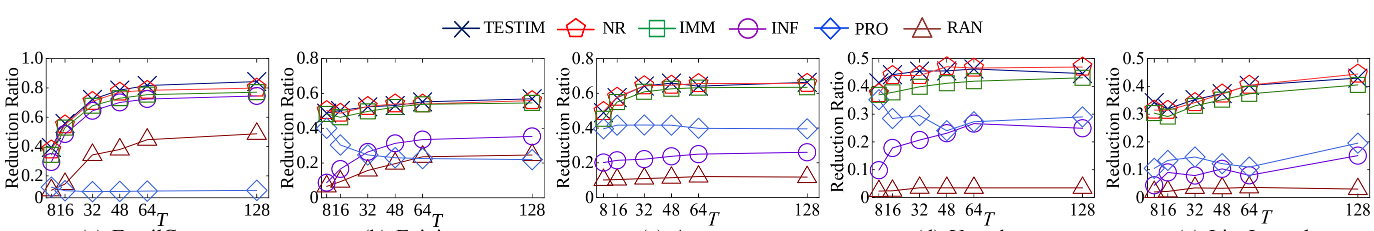
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 17. Running time vs. $|S_n|$ (skewed negative seeds)



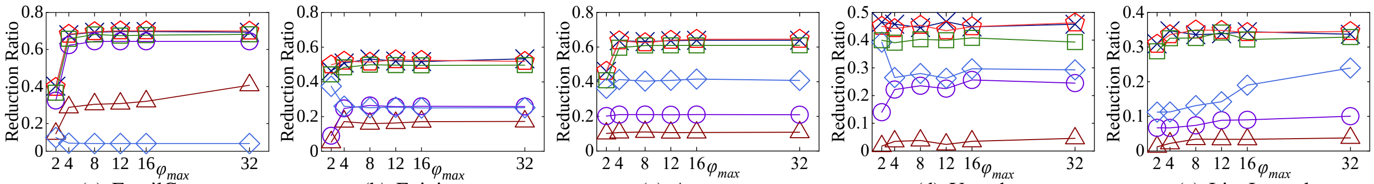
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 18. Running time vs. $|S_n|$ (random negative seeds)



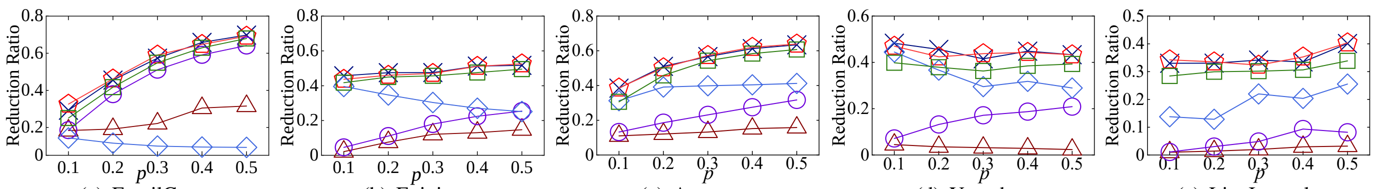
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 19. Reduction Ratio vs. T (skewed negative seeds)



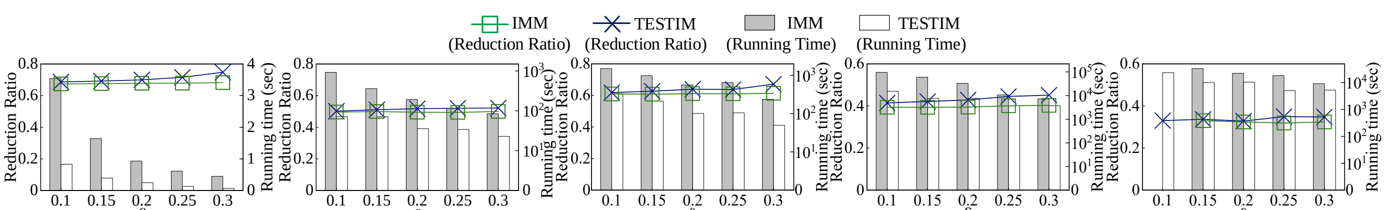
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 20. Reduction Ratio vs. $\varphi_{\max}$ (skewed negative seeds)



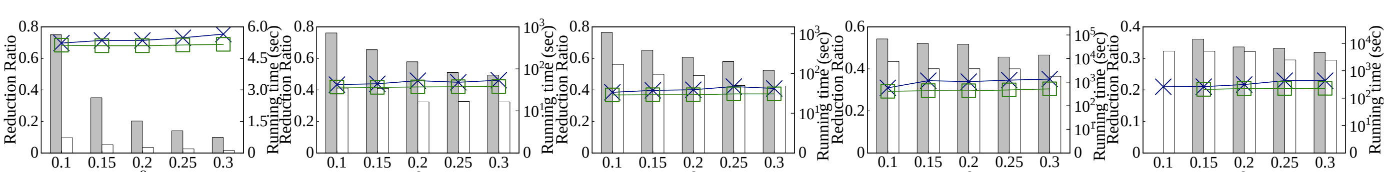
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 21. Reduction Ratio vs. p (skewed negative seeds)



(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 22. Reduction Ratio and Running time vs. ϵ (skewed negative seeds)



(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Fig. 23. The impact of ϵ on Reduction ratio and Running time (random negative seeds)