Time-aware Influence Minimization via Blocking Social Networks

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

ABSTRACT

We study the problem of Time-aware Influence Minimization (Timin) in social networks, aiming to minimize the negative influence concerning a critical deadline by temporarily blocking some nodes of the given social network. To this end, first, we introduce a novel Time-delayed Linear Threshold (TLT) model by considering the time delay of influence, i.e., when a node is active, its out-neighbors receive the influence weight after a certain time delay. Building on the TLT model, we formally define the Timin problem, and prove that it is **NP**-hard, *monotone*, and *supermodular*. To tackle the Timin problem, we initially devise a basic greedy algorithm, Timin-Greedy, achieving $(1-1/e)$ approximation. Since it is #**P**-hard to compute the exact negative influence spread for any node set in Timin-Greedy, we devise a *Temporal Reverse Influence Sampling* technique to estimate the expected negative influence spread, and propose a more efficient algorithm TESTIM, maintaining $(1-1/e-\epsilon)$ approximation. To further improve the efficiency, we propose a heuristic algorithm NeighborReplace based on an important observation that potential blocking nodes are often located near the negative source. Furthermore, we investigate two variants of the Timin problem, which consider additional constraints. Finally, our extensive experiments demonstrate that **(1)** TESTIM is up to 10× faster than the baselines, yielding 30%–50% more negative influence spread reductions, and **(2)** compared with TESTIM, NeighborReplace exhibits 5× speedup while having comparable negative influence spread reductions.

PVLDB Reference Format:
Xueqin Chang, Jiajie Fu, Qing Liu, Yunjun Gao, Baihua Zheng, Lu Chen. Time-aware Influence Minimization via Blocking Social Networks. PVLDB, 14(1): XXX-XXX, 2023.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at URL_TO_YOUR_ARTIFACTS.

1 INTRODUCTION

Influence minimization, with a wide range of applications, aims to minimize the expected influence of negative phenomena, be it fake news, rumors, or diseases [32–34]. For example, in political election campaigns, influence minimization can mitigate the adverse impact of rumors on candidates; in the realm of epidemiology control, it can curtail the spread of infectious diseases. Various approaches have been proposed for influence minimization, categorized as *clarification-based* [4, 14, 24, 33, 34] and *blocking-based* [29, 32, 37, 46, 47]. Clarification-based methods promote positive content for enhance user awareness and reduce the acceptance of misinformation. In contrast, blocking-based methods remove (or monitor) some critical users or connections from networks to minimize the spread of negative influences. In this paper, we employ the blocking techniques for influence minimization.

Many influence minimization studies assume that influence propagates immediately at successive timestamp [14, 29, 32, 37, 47]. However, in real-world applications, there is typically a delay in influencing others [5, 8, 25, 33, 34]. For example, in online social networks, users often do not immediately adopt or share newly received information from friends, for reasons like users being busy, having suspicions about the message, or waiting for confirmation from a sufficient number of friends [18]. Moreover, disease transmission varies among individuals due to different viral incubation periods [22], making influences among people time-sensitive and subject to varying time delays. Taking these factors into account, studies like [24, 33, 34] have examined the influence minimization problem with a focus on time delays, using clarification-based methods. Specifically, [34] and [24] assume that the negative and positive influences propagate at the same rate, while [33] considers different propagation rates for negative and positive influences. However, [24, 33, 34] may face limitations in effectiveness in real applications. As highlighted in [44], negative information propagates 6× faster than positive information. There may be scenarios where, at *a particular timestamp*, 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 a voting day in an election campaign, as mentioned in the previous paragraph.

Time-aware Influence Minimization Problem. Motivated by the above limitation, in this paper, we study a Time-aware Influence Minimization (Timin) problem via blocking social networks. To this end, first, we introduce a novel Time-delayed Linear Threshold (TLT) model to simulate the negative information/disease diffusion process considering time delay. To be specific, in the TLT model, a node receives the influence weights from its active in-neighbors after certain time delays, and continually accumulates the received influence weights within the survival time. When the accumulated influence *live-weights* are larger than the trust threshold, the node will be activated. 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 $|B| \leq k$, 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.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

Application 1. *During political elections, false rumors about candidates frequently circulate on social platforms [35]. However, information diffusion may experience delays due to user online behaviors and skepticism. To protect candidates from the impact of rumors, it is crucial to minimize rumor spread before the voting day. Thus, the campaign team can use* Timin *to identify key users in social networks and temporarily suspend their accounts until the election concludes, reducing the influence of rumors.*

Application 2. *Some infectious diseases, transmitted through the air and direct contact, exhibit varying transmission times due to viral incubation periods and individual immune responses [22]. To safeguard public health and relieve the strain on healthcare systems, it is essential to minimize the spread of infectious diseases before effective treatments or vaccines become available, and medical supplies are adequately prepared. Thus, in disease control campaign, the center for disease control and prevention can employ* Timin *to quarantine critical individuals and restrict the scope of disease transmission.*

Theoretical Analyses and Solutions. Existing works [18, 32, 37, 45, 47] have proposed blocking-based methods to address influence minimization problem without considering the influence time-delay. However, these methods are unsuitable for tackling the Timin problem for two key reasons. **(1)** The influence minimization problem under certain propagation models, such as the Independent Cascade model [47] and Susceptible-Infected-Recovered model [45], is non-supermodular. Thus, the corresponding algorithms [45, 47] are not theoretically guaranteed and cannot be applied to the Timin problem, which is supermodular. **(2)** The existing advanced influence estimation technique, i.e., reverse influence sampling technique utilized in [18, 32, 37], considers sampling the entire graph. However, when factoring in influence time-delay, nodes become activated sequentially. This necessitates the consideration of partial graphs, specifically the nodes activated by the deadline. Therefore, it is necessary to develop new 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. Given the inherent complexity of the Timin problem, we propose a greedy algorithm called Timin-Greedy, which is to select one node at a time until k nodes are found so that blocking the selected nodes has the minimum negative influence spread in each round. Timin-Greedy leverages the monotonicity and supermodularity properties of the Timin problem to provide a $(1-1/e)$-approximation. However, the computation of the exact negative influence spread for any node set in Timin-Greedy is proved to be #**P**-hard, which is impractical for large graphs. To address this challenge, we propose a *Temporal Reverse Influence Sampling (TRIS)* technique to compute the expected negative influence spread. Specifically, *TRIS* first generates two sets of random temporal-reverse reachable *(T-RR)* sets and reverse reachable-blocking *(RR-B)* sets. Then, along with a novel unbiased estimation method tailored specifically for TLT model, *TRIS* uses *T-RR* sets to estimate the expected influence spread of negative information with no blocking nodes, and utilizes *RR-B* sets to select the optimal blocking set. Based on *TRIS*, we propose a trial-and-error algorithm called TESTIM, which leverages the *TRIS* to greedily select k blocking nodes with a $(1-1/e-\epsilon)$-approximation guarantee. 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 NeighborReplace to further improve the efficiency. Specifically, NeighborReplace first finds k nodes from the out-neighbors of the negative nodes, and then updates the k nodes with negative nodes' h-hops neighbors by gradually increasing h ($h > 1$), which yields compected expected negative influence spread reductions.

Extensions. Moreover, we study two variations of the Timin problem, i.e., Deadline-sensitive Timin (DsTimin) and Budget-sensitive Timin (BsTimin), which are presented as follows.

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 $B \subseteq V - S_n$ with $|B| \leq k$ and a maximum timestamp T^*, such that if B is blocked, the proportion of the expected negative influence spread before T^* remains below α. This maximum timestamp T^* indicates the tight deadline for people to make preparations against the larger proportion of the expected negative influence spread, say surpass α. For example, in infectious disease control campaigns, the pursuit of effective treatments or vaccines often proves daunting. Therefore, it is vital to establish a maximum deadline, before which, temporarily isolating crucial patients can help keep the spread of infectious diseases below a predefined threshold. This deadline provides governments and healthcare systems with sufficient time to explore effective treatments.

BsTimin. Given a graph $G(V, E)$, a set of negative seed nodes $S_n \subseteq V$, a deadline T, and a threshold α, the BsTimin problem tries to return *a minimum blocking node set* $B^* \subseteq V - S_n$ such that there does not exit $B' \subseteq V - S_n$ satisfying **(1)** after blocking B^* or B', the proportion of the expected negative influence spread is less than α before the deadline T, and **(2)** $|B^*| < |B'|$. BsTimin helps users be aware of the minimum cost for blocking. For example, during political elections, social platforms may suspend user accounts responsible for spreading rumors on their networks. However, suspending too many accounts can detrimentally affect the user experience. Hence, it is critical to temporarily suspend the smallest set of user accounts that can effectively restrain the spread of rumors to an acceptable threshold until the end of elections.

Contributions. Our contributions are summarized as follows.

- We introduce a new TLT model by considering influence time-delays. Based on the TLT model, we formulate the Timin problem, which is proved to be **NP**-hard, *monotone*, and *supermodular*.
- We propose a greedy algorithm Timin-Greedy and an optimized algorithm TESTIM, with $(1-1/e)$ and $(1-1/e-\epsilon)$ approximations, respectively. We also devise a heuristic algorithm NeighborReplace 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 $(1-1/e-\epsilon)$ approximation.
- Extensive experiments on five real networks and a visualized case study on Facebook demonstrate the effectiveness, efficiency, and scalability of the proposed algorithms.

Roadmap. § 2 formally defines the TLT model and Timin problem. § 3 introduces the Timin problem algorithm. § 4 presents the variants of the Timin problem and the corresponding solutions. Experimental results are reported in § 5. § 6 reviews the related work, and § 7 concludes the paper.

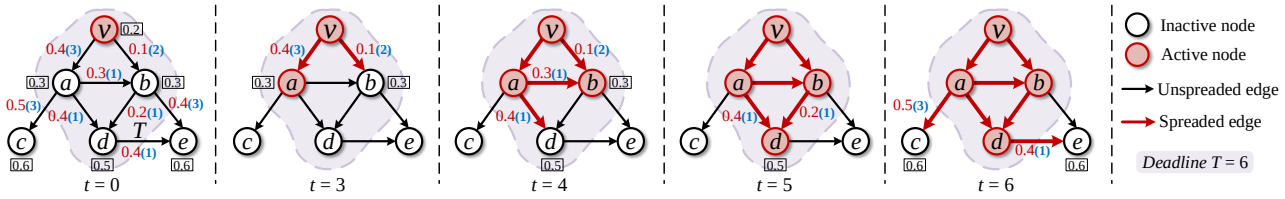


Figure 1: Illustrating information propagation under *Time-delayed Linear Threshold* (TLT) model; red numbers on the edges are influence weights, blue numbers inside the brackets are time delays, and numbers besides nodes are activation thresholds

2 PRELIMINARIES

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

2.1 The TLT Model

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, i.e., if a node u is active at timestamp t, then v will be influenced by u with influence weight $w_{u,v}$ at timestamp $t + \varphi_{u,v}$. Note that the delay times of all edges follow *Poisson* or *Geometric* distribution [8, 23]. In real applications, the influence weights are not everlasting. Thus, we introduce the maximum survival time φ_{max} for influence weight [11], and define the *live-weight* as follows.

Definition 1. **(Live-weight).** *Given a graph $G(V, E)$, two nodes $u, v \in V \wedge (u, v) \in E$, and a maximum survival time φ_{max}. Assume that at timestamp t, the node u becomes active and starts to influence the node v with the influence weight $w_{u,v}$, then $w_{u,v}$ is alive for v during the time $[t, t + \varphi_{max}]$. We employ $\widetilde{w}_{u,v}(t)$ to denote the live-weight at the timestamp t.*

Definition 1 specifies the lifecycle for an influence weight. Note that the influence of u on v will disappear after the timestamp $t + \varphi_{max}$. If the delay time is larger than φ_{max}, the influence weight drops to zero before being received by the influenced nodes. Based on the above settings, we present the TLT model. Given a graph $G(V, E)$, a negative seed nodes set $S_n \subseteq V$ 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. 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., $\sum_{u:(u,v)\in E} w_{u,v} \leq 1$). At timestamp 0, the 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 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., $\sum_{u:(u,v)\in E \wedge u \text{ is acticated}} \widetilde{w}_{u,v}(t') \geq \theta_v$. Otherwise, v remains inactive. Once the node v becomes active, it remains active until the end of the diffusion process. *The* TLT *propagation terminates until meeting the deadline T or no nodes are activated.*

Example 1. *Figure 1 illustrates the influence propagation process under the* TLT *model. Let v be the negative seed node, the deadline $T = 6$, and the maximum survival time $\varphi_{max} = 4$. The number in the rectangle next to each node represents the activation threshold. The red and blue numbers on the edges represent influence weight and time delay, respectively. 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 $\widetilde{w}_{v,a}(3) = 0.4 > \theta_a = 0.3$. Although node b receives the influence weight from v too at $t = 2$, b remains inactive since $\widetilde{w}_{v,b}(3) = 0.1 < \theta_b = 0.3$.*
- *$t = 4$, b receives additional influence weight from a and becomes active since $\widetilde{w}_{v,b}(4)+\widetilde{w}_{a,b}(4) = 0.1+0.3 = 0.4 > 0.3$. 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 the accumulated live-weights is $\widetilde{w}_{a,d}(5) + \widetilde{w}_{b,d}(5) = 0.4 + 0.2 = 0.6 > 0.5$. Thus, d is successfully activated by a and b.*
- *$t = 6$, nodes c and e remain inactive and the influence propagation process terminates. Finally, the active node set is $\{v, a, b, d\}$.*

Comparison with existing time-aware influence propagation models. Simpson et al. [33] proposed the TCIC model, which considers different propagation rates of truth and misinformation, and the user real-reaction times. Song et al. [34] employed IC-M model [8] to address the influence minimization problem, incorporating time-delay as a node-level online probability that models user's log-in and log-out behavior. However, both TCIC and IC-M models are extended from Independent Cascade (IC) model, which cannot effectively represent user skepticism as the activation in IC model is subject to random variables. Additionally, Pham et al. [30] developed the T-DLT model, which considers the maximum propagation hops among users for epidemic control, based on the Linear Threshold (LT) model. To our best knowledge, our proposed TLT model is the first to incorporate edge-level time delays into LT model to simulate the influence delays.

2.2 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 impact of a seed nodes set by the deadline T on the entire graph.

Definition 2. **(Expected influence spread).** *Given a graph $G(V, E)$, a negative seed nodes set S_n, 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 adopt a strategy that blocks some key nodes (excluding 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 vertex v in V, if v is a blocking node, the influence weight of all v's incoming edges is set to 0, i.e., $\forall\ (u, v) \in E,\ w_{u,v} = 0$.*

Since any incoming edges of blocking nodes have an influence weight of zero, the accumulated influence weights cannot exceed the activation thresholds of the blocking nodes, and thus these nodes will never be activated. Based on the above definitions, we formally define the Timin problem.

Problem 1. **(Timin).** *Given a graph $G(V, E)$, a negative seed nodes set $S_n \subseteq V$, 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:*

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

2.3 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. The Influence Minimization problem under the classical LT model has been proved to be **NP**-hard [18]. Timin, as a special case of Influence Minimization under the TLT model, specifically with a time distribution following the Geometric distribution with probability of 1, along with an infinite maximum survival time φ_{max} and *deadline* T, is also **NP**-hard.

Proposition 1. *The Timin problem is* **NP***-hard under the* TLT *model.*

Temporal live-edge graph. Kempe et al. [17] 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 $e = (u, v) \in E$ from its incoming edges with a probability $w_{u,v}$. No live-edge is selected with probability $1 - \sum_{u:(u,v)\in E} w_{u,v}$. Consequently, each X_t exists with a probability $P(X_t|G) = \prod_{(u,v)\in E_{X_t}} w_{u,v} \cdot \prod_{(u,v)\in E\setminus E_{X_t}} (1 - w_{u,v})$. 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 that used for live-edge graphs, with the key distinction being the inclusion of time-delay on each edge.

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 any 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 most importantly, **(3)** there must be no nodes in B lying 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. The expected influence spread can be computed as follows, as demonstrated in [17, 18]:

$$\begin{aligned}\sigma(S_n, G[V \setminus B], T) &= \mathbb{E}[|A_{X_t}(S_n, G[V \setminus B], T)|] \\ &= \sum\nolimits_{X_t \in \mathcal{X}^t_{G\setminus B}} P(X_t|G \setminus B) \cdot |A_{X_t}(S_n, G[V \setminus B], T)|\end{aligned} \tag{1}$$

where $\mathcal{X}^t_{G\setminus B}$ is the space of all possible temporal live-edge graphs based on $G \setminus B := (V \setminus B, E)$. The expectation is taken across the distribution of these temporal live-edge graphs. Consider the function in Eq. (1), with respect to a blocking set B, by blocking any node $b \in V \setminus B$, we can calculate the influence reduction in the graph, i.e., $\sigma(S_n, G[V \setminus B], T) - \sigma(S_n, G[V \setminus (B \cup \{b\})], T)$. The former function is a sum over $\mathcal{X}^t_{G\setminus B}$, while the latter sums over $\mathcal{X}^t_{G\setminus (B\cup\{b\})}$. The challenge here is that as nodes are removed from the graph, *the set of temporal live-edge graphs and the associated probabilities involved in computing the influence function change.* It is not immediately clear that this function remains monotone, and a similar challenge arises in proving supermodularity. The objective influence function can be viewed as a set function over the blocking nodes B to be removed, i.e., $\sigma(S_n, G[V \setminus B], T) = \sum_{s\in S_n} \sigma(s, G[V \setminus B], T)$. Fortunately, it has been demonstrated in [18] that under the classical LT model without temporal constraints, each $\sigma(s, G[V \setminus B])$ forms a monotonically decreasing and supermodular function of B. Therefore, the same holds true for their cumulative sum $\sum_{s\in S_n} \sigma(s, G[V \setminus B])$. With this foundation, we present the following Lemmas [18].

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 $b \in V \setminus (S_n \cup B)$,*

$$\sigma(s, G[V \setminus B]) - \sigma(s, G[V \setminus (B \cup \{b\})]) \geq 0 \tag{2}$$

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'),*

$$R(b|B) - R(b|B') \geq 0 \tag{3}$$

The following theorems demonstrate that *our* Timin *problem retains the properties of being monotonically non-increasing and supermodular under* TLT *model*, even with the inclusion of time delays φ_t on the edges and propagation deadline T.

Theorem 1. *The expected influence spread function $\sigma(S_n, G[V \setminus B], T)$ is monotonically non-increasing under the* TLT *model.*

Proof. After generating a new graph $G\setminus B$, the inclusion of time-delays on the sampled live-edges does not affect the probability of generating a particular temporal live-edge graph X_t. Therefore, the space of temporal live-edge graphs $\mathcal{X}^t_{G\setminus B}$ and their associated probabilities $P(X_t|G \setminus B)$ remain the same as there are no time-delays on the edges. *Deadline* T also does not impact the probability of sampling a particular temporal live-edge graph. Instead, it only affects the set nodes in a certain X_t that can be reached from a seed $s \in S_n$, i.e., $A_{X_t}(s, G[V \setminus B], T)$. Two cases exist: **(1)** selecting the blocking node without considering the deadline T doesn't cause any decrease or increase in the number of nodes influenced by s in X_t; **(2)** if a blocking node is selected within T, it reduces to the scenario described in Lemma 1, where monotonicity is satisfied. Hence, regardless of whether the blocking node is

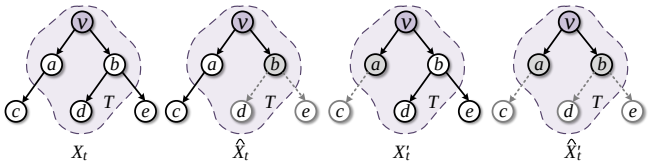


Figure 2: Example of a temporal live-edge graph; node v is the negative seed node; grey nodes are blocked nodes

selected within T, adding any blocking node to any set B cannot increase the expected influence spread of negative information under the TLT model. Building on Lemma 1, we conclude that the function $\sigma(s, G[V \setminus B], T)$ is monotonically non-increasing, i.e., $\sigma(s, G[V \setminus B], T) - \sigma(s, G[V \setminus (B \cup b), T]) \geq 0$. The influence spread of S_n is the cardinality of the union $\cup_{s \in S_n} \sigma(s, G \setminus B, T)$, hence constituting a monotonic function of B. □

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

Proof. As illustrated in the proof of Theorem 1, the inclusion of time-delays does not impact the space of temporal live-edge graphs $\mathcal{X}^t_{G \setminus B}$ or the corresponding probabilities $P(X_t | G \setminus B)$. We simply need to demonstrate that within the *deadline* setting, we hold $|A_{X_t}(s, G[V \setminus B], T)| - |A_{X_t}(s, G[V \setminus (B \cup \{b\})], T)| \geq |A_{X_t}(s, G[V \setminus B'], T)| - |A_{X_t}(s, G[V \setminus (B' \cup \{b\})], T)|$, where $B' = B \cup \{a\}$. 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, 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 $\sigma(s, G[V \setminus B], T)$ is supermodular, i.e., $R(b|B, T) - R(b|B', T) \geq 0$, where $R(b|B, T) = \sigma(s, G[V \setminus B], T) - \sigma(s, G[V \setminus (B \cup \{b\})], T)$ (resp. $R(b|B', T) = \sigma(s, G[V \setminus B'], T) - \sigma(s, G[V \setminus (B' \cup \{b\})], T)$). The influence spread of S_n is the cardinality of the union $\cup_{s \in S_n} \sigma(s, G \setminus B, T)$, thus constituting a supermodular function of B. □

3 TIMIN ALGORITHMS

In this section, we propose Timin-Greedy, designed for time-aware scenarios with a $(1 - 1/e)$-approximation guarantee. Since the efficient implementation of Timin-Greedy is challenging, we present its scalable version, TESTIM, which leverages the novel temporal reverse influence sampling technique. Finally, we design an efficiency-improving heuristic NeighborReplace.

Algorithm 1 Timin-Greedy

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

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

3.1 TIMIN-Greedy Algorithm

Timin-Greedy. Algorithm 1 outlines the pseudo-code of Timin-Greedy. We begin by initializing an empty blocking set B and set the temporal marginal loss $\Delta(v|B, T) = 0$ for each node $v \in V$, when removing v while nodes in B are blocked (Lines 1–2). In each iteration (Lines 3–5), we compute the temporal marginal loss $\Delta(v|B, T)$ for all nodes in $V \setminus (S_n \cup B)$, select the node with the maximum reduction of expected spread (Line 4), and insert it in B (Line 5). The process terminates when $|B| = k$ (Line 3), and the resulting solution B is returned (Line 6). It's important to note that the primary distinction between the classical greedy and Timin-Greedy lies in how they calculate the marginal loss of a node, considering temporal factors. Specifically, when computing the temporal marginal loss of a node v given a negative seed set S_n, the following conditions must be satisfied: **(1)** an activation path from any node $s \in S_n$ to any node $u \in V$ must exist through edges in E; **(2)** the cumulative time-delays should be within the deadline, i.e., $\varphi_{s,u} \leq T$; and **(3)** v must lie on the path from s to u. The performance 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 before deadline T when blocking B in G, and B^* denotes the optimal solution to our problem. The solution B returned by* Timin-Greedy *satisfies:*

$$R(B, T) \geq (1 - 1/e) \cdot R(B^*, T) \tag{4}$$

Proof. Given the monotonicity and supermodularity of the Timin problem established in §2.3, the assurance of approximation follows directly from [28]. □

Proposition 2. *Given a graph $G(V, E)$, a negative seed set S_n, and a deadline T, it is #**P**-hard to exactly compute $\sigma(S_n, G[V], T)$ under the* TLT *model.*

Proof. Computing the exact influence spread $\sigma(S)$ for any set S under the classical LT model is proven to be #**P**-hard [9]. 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 #**P**-hard. □

3.2 TESTIM Algorithm

Algorithm 1 (Timin-Greedy) requires numerous influence spread computations to identify the blocking nodes that maximally reduce influence spread. 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 #**P**-hard as proven in Proposition 2. Recent studies have

shifted towards sampling-based influence spread estimation, ranging from naive *Monte-Carlo (MC)* Simulations [17] to advanced *Reverse Influence Sampling (RIS)* [3]. In *RIS*, each sampled *Reverse Reachable (RR)* set R is generated as follows:

- Select a node v randomly from V in G.
- Generate a live-edge path from v by following incoming edges.
- R includes nodes on the live-edge path, including v.

Given a node set S and a random RR set R, we define a random variable $Y(S,R)$ such that $Y(S,R) = 1$ if $S \cap R \neq \emptyset$ and $Y(S,R) = 0$ otherwise. Tang et al. [41] showed that $\sigma(S) = n \cdot \mathbb{E}[Y(S,R)]$. When generating a sufficient number of RR sets $\mathcal{R} = \{R_1, R_2, ...\}$, $n \cdot \mathbb{E}[Y(S,R)]$ could be estimated unbiasedly using the empirical mean $\sum_{R\in\mathcal{R}} Y(S,R)/|\mathcal{R}|$, based on concentration bounds.

For our objective function, given a negative seed set S_n and a deadline T, we need to develop a method to estimate the expected influence spread $\sigma(S_n, G[V \setminus B], T)$ for any blocking set B. To address this, we extend upon classical *RIS* and propose the *Temporal Reverse Influence Sampling (TRIS)* method, which accounts for temporal factors. We introduce two key concepts: the *Temporal-Reverse Reachable (T-RR)* set and the *Reverse Reachable-Blocking (RR-B)* set for any possible temporal live-edge graph under the TLT setting. The T-RR set is designed for estimating the influence spread of a negative seed set without any nodes being blocked, while the RR-B set is used for greedy selection of optimal blocking nodes. Unlike classical RR sets, which only focus on a node's inclusion in an RR set, our T-RR set and RR-B set additionally consider *cumulative time-delay* and the *deadline*. These sets can be viewed as weighted variants of classical RR set and are generated as follows:

Under the TLT model, during the generation of T-RR set for a node v, we add any newly visited node along the live-edge path to the set. T-RR set generation terminates when we either reach a negative seed $s \in S_n$ or can no longer explore new node. When we encounter a negative seed s, we record the cumulative time-delay $\varphi_{s,v}$ from seed s to the sampled node v. If this cumulative time-delay is within the deadline, i.e., $\varphi_{s,v} \leq T$, the T-RR set is duplicated to form the RR-B set. This duplication signifies that a blocking node on the live-edge path has a potential to prevent node v from receiving negative information within the deadline. Notably, in our problem, the influence of a negative seed on a node is considered valid only if it occurs within the deadline, and similarly, *the action of a blocking node saving that particular node is deemed valid if accomplished within the same deadline.* Given the time-aware constraints in our model and problem, traditional unbiased estimation methods fall short. As a result, we design a novel approach to bridge this gap.

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 a negative seed $s \in S_n$ to the sampled source v in R_t does not exceed the deadline, i.e., $\varphi_{s,v} \leq T$. Condition **(2)** ensures that the activation from s to v can't occur after the deadline. Otherwise, $\Lambda_{R_t}(S_n, G[V], T) = 0$. Given a set $\mathcal{R}_t$ of random T-RR set, we denote $f^{\mathcal{R}_t}(S_n, G[V], T)$ 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,

$$f^{\mathcal{R}_t}(S_n, G[V], T) = \frac{\Lambda_{\mathcal{R}_t}(S_n, G[V], T)}{|\mathcal{R}_t|} \cdot n \tag{5}$$

Algorithm 2 TESTIM

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

1: Initialize $B \leftarrow \emptyset$, $\theta_1 \leftarrow n$, $i \leftarrow 1$
2: **while true do**
3: Generate two sets of random T-RR sets, $|\mathcal{R}_{t1}| = |\mathcal{R}_{t2}| = \theta_i$
4: Generate two sets of random RR-B sets, $\mathcal{R}_{b1} \subseteq \mathcal{R}_{t1}$, $\mathcal{R}_{b2} \subseteq \mathcal{R}_{t2}$
5: **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**
6: **break**
7: Compute $f^{\mathcal{R}_{t1}}(S_n, G[V], T)$, $f^{\mathcal{R}_{t2}}(S_n, G[V], T)$
8: $B \leftarrow$ TIMIN-Oracle$(\mathcal{R}_{b1}, f^{\mathcal{R}_{t1}}(S_n, G[V], T))$
9: $\lambda \leftarrow g^{\mathcal{R}_{b1}}(B,T)/g^{\mathcal{R}_{b2}}(B,T)$
10: $\epsilon_1 := (\epsilon_1+1)(\epsilon_1+2)/\epsilon_1^2 = g^{\mathcal{R}_{b2}}(B,T)/\ln(5\cdot i^2/\delta)\cdot\theta_i/n$
11: $\epsilon_2 := (2\epsilon_1+2)/\epsilon_2^2 = g^{\mathcal{R}_{b2}}(B,T)/\ln(5\cdot i^2/\delta)\cdot\theta_i/n$
12: **if** $0 < \lambda \leq (\frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1})$, ϵ_1, $\epsilon_2 \in (0,1)$ **then**
13: **break**
14: $i \leftarrow i+1$, $\theta_i \leftarrow 2\theta_i$
15: **Return** B

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 set, we have $g^{\mathcal{R}_b}(B,T)$ as an unbiased estimation of $R(B,T)$, which refers to the influence reduction of B: $R(B,T) = \sigma(S_n, G[V], T) - \sigma(S_n, G[V\setminus B], T)$, formally,

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

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

Based on TIMIN-Greedy and *TRIS* technique, we propose the *Trial-and-Error for Scalable Time-aware Influence Minimization (TESTIM)* algorithm. TESTIM starts with an empty solution set B and iteratively adds node with the maximal marginal loss. It utilizes the *TRIS* method to estimate influence spread of nodes and returns a solution set B with a theoretical guarantee when a sufficient number of RR-B sets are generated. A key challenge arises: *how large a sample set $\mathcal{R}_b$ is needed to achieve the approximation guarantee without excessive computational overhead?* Inspired from [16], we adopt a trial-and-error approach in TESTIM to overcome this hurdle. During the generation of T-RR sets, we incrementally double the number of T-RR sets and monitor approximation guarantee. TESTIM terminates when the approximation reaches the desired value or when the number of RR-B sets is sufficient. Algorithm 2 lists the pseudo-code of TESTIM, while Algorithm 3 (TIMIN-Oracle) presents a sub-routine invoked to greedily select the optimal blocking set B and estimate its expected influence reduction.

Algorithm 2 first generates two collections of T-RR sets with $|\mathcal{R}_{t1}| = |\mathcal{R}_{t2}| = n$ and two sets of RR-B sets $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$ based on $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ (Lines 1–4). TESTIM uses $\mathcal{R}_{t1}$ ($\mathcal{R}_{t2}$) to unbiasedly estimate the influence spread of negative seed set S_n when no node is blocked (Line 7). Afterwards, it uses $\mathcal{R}_{b1}$ as the input to the TIMIN-Oracle, which generates a solution B by employing the *TRIS* technique in TIMIN-Greedy (Line 8). Subsequently, it uses $\mathcal{R}_{t2}$ and $\mathcal{R}_{b2}$ to verify the quality of solution B (Lines 9–13) since they are independent of $\mathcal{R}_{t1}$ and $\mathcal{R}_{b1}$. We suppose that if the estimation

Algorithm 3 TIMIN-Oracle

Input: RR-B sets $\mathcal{R}_b$, expected influence spread $f^{\mathcal{R}_t}(S_n, G[V], T)$
Output: B

1: Initialize $B \leftarrow \emptyset$
2: Let $\Delta_{\mathcal{R}_b}(v)$ be the number of RR-B sets covered by v in $\mathcal{R}_b$
3: **while** $|B| < k$ **do**
4: $\quad v^* \leftarrow \arg\max_{v \in V \setminus (S_n \cup B)} \Delta_{\mathcal{R}_b}(v)$
5: $\quad B \leftarrow B \cup \{v^*\}$
6: $\quad$ Remove from $\mathcal{R}_b$ all RR-B sets that are covered by v^*
7: Compute the reduction influence $g^{\mathcal{R}_b}(B, T)$
8: **Return** B

reduction derived from $\mathcal{R}_{b2}$ is much smaller than the estimation derived from $\mathcal{R}_{b1}$, it means that $\mathcal{R}_{b1}$ over-estimates B's reduction. In this case, TESTIM discards solution B, doubles the size of $\mathcal{R}_{t1}$ and $\mathcal{R}_{t2}$ (Line 14) and repeats the above process until a satisfying solution is returned, i.e., **(1)** $\mathcal{R}_{b2}$ agrees the quality of B generated by $\mathcal{R}_{b1}$ (Lines 12–13), or **(2)** the number of generated RR-B sets $\mathcal{R}_{b1}$, i.e., $|\mathcal{R}_{b1}|$, reaches $(8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2 \cdot g^{\mathcal{R}_{b2}}(B,T)}$ (Lines 5–6). Finally, Algorithm 2 terminates with blocking set B (Line 15). Theorem 3 shows the approximation guarantee provided by TESTIM, where B^* is the optimal solution and $\delta, \epsilon \in (0,1)$ are input parameters.

THEOREM 3. **(Approximation Guarantee of TESTIM).** *For TIMIN problem, let $R(B,T)$ denote the reduction in influence spread before deadline T when B is blocked in V, 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:*

$$R(B,T) \geq (1 - 1/e - \epsilon) \cdot R(B^*, T) \tag{7}$$

In what follows, we tackle two key challenges in TESTIM while satisfying Theorem 3, that is, **(1)** how to set the maximum number of RR-B sets $\mathcal{R}_{b1}$ (Lines 5–6) and **(2)** how to set conditions to evaluate whether the current solution satisfies the performance guarantee (Lines 9–12). First, we extend the *Chernoff Inequalities* [27] and propose the following concentration bounds:

$$\Pr[g^{\mathcal{R}_{b2}}(B,T) - R(B,T) \geq \epsilon_1 \cdot R(B,T)] \leq \exp\left(-\frac{\epsilon_1^2}{2+\epsilon_1}\frac{|\mathcal{R}_{b2}|}{\Gamma_2}R(B,T)\right)$$

$$\Pr[g^{\mathcal{R}_{b1}}(B^*,T) - R(B^*,T) \leq -\epsilon_2 \cdot R(B^*,T)] \geq \exp\left(-\frac{\epsilon_2^2}{2}\frac{|\mathcal{R}_{b1}|}{\Gamma_1}R(B^*,T)\right)$$

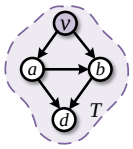
where $|\mathcal{R}_{b1}|$ and $|\mathcal{R}_{b2}|$ is the size of $\mathcal{R}_{b1}$ and $\mathcal{R}_{b2}$, $\Gamma_1 = f^{\mathcal{R}_{t1}}(S_n, G[V], T)$, $\Gamma_2 = f^{\mathcal{R}_{t2}}(S_n, G[V], T)$. Then, in each round of TESTIM (Lines 3–15), the estimations $g^{\mathcal{R}_{b1}}(B^*,T)$ and $g^{\mathcal{R}_{b2}}(B,T)$ 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*

$$g^{\mathcal{R}_{b2}}(B,T) \leq (1+\epsilon_1)\, R(B,T) \tag{8}$$

$$g^{\mathcal{R}_{b1}}(B^*,T) \geq (1-\epsilon_2)\, R(B^*,T) \tag{9}$$

Based on Lemma 4, consider two cases that depend on whether Line 12 in TESTIM is satisfied. **Case (1):** Line 12 is satisfied, then in the last iteration, we have $0 < \lambda \leq (\frac{1-1/e}{1-1/e-\epsilon} \cdot \frac{1-\epsilon_2}{1+\epsilon_1})$. By Eq (8) and $g^{\mathcal{R}_{b1}}(B,T) \geq (1-1/e) g^{\mathcal{R}_{b1}}(B^*,T)$ and $\lambda = g^{\mathcal{R}_{b1}}(B,T)/g^{\mathcal{R}_{b2}}(B,T)$,



RR-B sets
{d, a, v} {b, a, v}
{d, b, v} {b, v}
{d, b, a, v} {a, v}
$\Delta(a)$:4 $\Delta(b)$:4 $\Delta(d)$:3

Graphs	|B|	|Neighbors|
EmailCore (n=1k, m=26k)	50	49 (98%)
Epinions (n=76k, m=509k)	50	47 (94%)
Amazon (n=335k, m=926k)	50	49 (98%)

Figure 3: Motivating examples of heuristic algorithm

we prove that Eq. (7) holds with probability at least $1 - \frac{2\delta}{3}$. **Case (2):** Line 12 is not satisfied, when TESTIM terminates, by Eq. (8), we have $g^{\mathcal{R}_{b2}}(B,T) \leq (1+\epsilon_1)R(B^*,T)$, let $x = \epsilon R(B^*,T)/2R(B,T)$ for any $B \subseteq (V \setminus S_n)$, we have $\Pr[g^{\mathcal{R}_{b1}}(B,T) - R(B,T) \geq \frac{\epsilon}{2} \cdot R(B^*,T)] \leq \exp\left(-\frac{x^2}{2+x}\frac{|\mathcal{R}_{b1}|}{n}R(B,T)\right) \leq \exp\left(-\frac{\epsilon^2}{8+2\epsilon}\frac{|\mathcal{R}_{b1}|}{n}R(B^*,T)\right) \leq \frac{\delta}{6\cdot 2^n}$, then $|\mathcal{R}_{b1}| = (8+2\epsilon)(1+\epsilon_1)n\frac{\ln\frac{6}{\delta}+n\ln 2}{\epsilon^2 \cdot g^{\mathcal{R}_{b2}}(B,T)}$. Thus, when Line 12 is not satisfied, Eq. (7) holds. Combining these two cases, the approximation guarantee of TESTIM is demonstrated (detailed proofs can be found in [6]). The theoretical time complexity of TESTIM is as below.

THEOREM 4. **(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.*

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 $O(\frac{n \cdot \max\{\ln\frac{1}{\delta}, n\}}{\epsilon^2})$ [16]. The expected time of generating a random T-RR set is bounded by $\frac{m}{n} \cdot R(\{v^*\}, T)$ [41]. Hence, the expected time complexity of Algorithm 2 is $O(\frac{\max\{\ln\frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\},T)}{\epsilon^2})$. □

3.3 NeighborReplace Algorithm

As discussed in § 3.2, the selection of blocking set B relies on the RR-B set, which includes a negative seed $s \in S_n$ and a cumulative time-delay within the specified deadline. Referring to the generation process of temporal live edge graphs in § 2.3, the out-neighbors of negative seeds play a significant role in RR-B sets, as negative seeds act as termination nodes during generation. We simulate the spread of negative information from the top-20 influential nodes, i.e., $|S_n| = 20$, by selecting a set of 50 blocking nodes, i.e., $|B| = 50$. Figure 3 plots that, across three real-world social networks, the out-neighbors of negative seeds are predominantly included in B, confirming their selection as blocking nodes. This makes it easy to sample the out-neighbors of S_n into RR-B sets, as depicted in Figure 3. Consequently, these nodes tend to appear in a large number of RR-B sets (i.e., a larger $\Delta_{\mathcal{R}_b}(\cdot)$) in Algorithm 3. In addition, we apply TESTIM to select blocking set B while theoretically minimizing the negative influence spread.

Building on these theoretical and empirical observations, inspired by [47], we propose a new heuristic NeighborReplace. It initially selects k out-neighbors of negative seeds as the initial blocking nodes. Subsequently, we replace these blocking nodes in reverse order of their insertion into B. During each replacement iteration, we greedily choose the node with the largest marginal loss from the next level neighbors of the node being replaced. The replacement process terminates when no node with larger marginal loss is available. Note that, we employ the *TRIS* technique to compute the reduction in expected influence spread.

Algorithm 4 NeighborReplace

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

1: Initialize $B \leftarrow \emptyset$, a maximum heap $H \leftarrow \emptyset$, $t \leftarrow 0$
2: Initialize $N \leftarrow \emptyset$ //N records out-neighbors of S_n at different levels
3: $(N, t) \leftarrow BFS(G, S_n)$ //t is the maximum level 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 $H \leftarrow N[i][j]$
7: **if** $|H| \geq k$ **then break**
8: **while** $|B| < k$ **do**
9: $v \leftarrow H.top()$, $B \leftarrow B \cup \{v\}$
10: For $\forall x \in H \setminus B$, update $\Delta(x|B, T)$, $H.pop()$
11: **for** each $u \in B$ in reverse order of insertion **do**
12: $B \leftarrow B \setminus \{u\}$
13: **for** $i \leftarrow \kappa + 1$ to t **do** // $\kappa \leftarrow$ the level that u belongs to
14: $v^* \leftarrow \arg\max_{w \in N[i] \setminus (S_n \cup B)} \Delta(w|B, T)$
15: **if** $\Delta(v^*|B, T) > \Delta(u|B, T)$ **then** $B \leftarrow B \cup \{v^*\}$, **break**
16: **else continue**
17: **if** $i = t + 1$ **then break**
18: **Return** B

Algorithm 4 shows the NeighborReplace approach. First, it initializes an empty blocking set B, an empty maximum heap H containing the out-neighbors of S_n with their marginal losses (Line 1) and an empty 2-dimensional vector N recording the out-neighbors of S_n at different levels, where $N[i][j]$ denotes the j-th node in the i-th level out-neighbors of S_n (Line 2). A Breadth-First Search is then applied to construct N and t (Line 3). The algorithm iteratively computes the marginal losses of out-neighbors at each level and inserts them into the heap, along with their marginal losses (Lines 4–7). It then constructs B and updates the marginal losses of nodes in the heap (Lines 8–10). Next, it replaces the blocking nodes in B in reverse order of their insertions (Lines 11–17). The replaced node is removed from B (Line 12), and v^* maintains the node with the largest marginal loss computed so far, which belongs to u's next level out-neighbors (Lines 13–14). If v^* outperforms u, a replacement is executed and the algorithm proceeds to check the next node in B (Line 15); otherwise, it continues searching for nodes in the next level out-neighbors (Line 16). If there are no nodes with higher influence spread reductions, the replacement terminates (Line 17), and the solution B is returned (Line 18). We observe that the replacement ends at the third level in our experiments.

4 EXTENSIONS

In this section, we explore the significance of time and budget constraints in influence minimization problem. We introduce two variant problems, DsTimin and BsTimin, and present two effective $(1 - 1/e - \epsilon)$-approximation methods for solving them.

4.1 DsTimin Problem

In real-world scenarios, it's vital to keep negative influence below a certain threshold to prevent severe consequences. As Claude Shannon, the mathematician, showed, as long as negative influences stay beneath a specific threshold, positive aspect can thrive [31]. Setting an acceptable threshold provides a practical goal for limiting the reach of negative influence, particularly in complex and ever-changing social environments. This approach enables platforms and policymakers to target their efforts on the most critical negative influence without overwhelming resources. To characterize this specific threshold, we introduce a key parameter α, which represents the proportion of negative influence within a social network.

Algorithm 5 DSIM

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

1: Initialize $B^o \leftarrow \emptyset$, $l \leftarrow 0$, $r \leftarrow T_m$
2: **while** $l \leq r$ **do**
3: $T^o \leftarrow (l + r)/2$
4: $B^o \leftarrow$ TESTIM(G, S_n, k, T^o, ϵ, δ)
5: **if** $\sigma(S_n, G[V \setminus B^o], T^o)/n = \alpha$ **then break**
6: **else if** $\sigma(S_n, G[V \setminus B^o], T^o)/n < \alpha$ **then** $l \leftarrow T^o + 1$
7: **else if** $\sigma(S_n, G[V \setminus B^o], T^o)/n > \alpha$ **then** $r \leftarrow T^o - 1$
8: **Return** (B^o, T^o)

Furthermore, we acknowledge a real-world observation: negative influence tends to spread much faster than positive influence [44]. This underscores the importance of keeping negative influence under control for as long as possible, providing sufficient time for positive aspects (i.e., truth, clarification and medical treatments) to propagate and mitigate the harmful effects of negative influence. Based on this, we extend our Timin problem and formally define the Deadline-sensitive Timin (DsTimin), as shown below.

Problem 2. **(DsTimin).** *Given a graph $G(V, E)$, a negative seed node set S_n, a budget k, and a threshold α, the DsTimin problem is to find a set $B^o \subseteq (V \setminus S_n)$ with $|B^o| = k$ and a **maximum** timestamp T^o, such that if B^o is blocked, the proportion of the expected active nodes by the timestamp T^o remains below α, i.e.,*

$$(B^o, T^o) := \arg\max_{B^o \subseteq (V \setminus S_n) \wedge |B^o| = k \wedge \sigma(S_n, G[V \setminus B^o], T^o)/n \leq \alpha} T^o$$

Based on Algorithm 2 and binary search technique, we propose the *Binary Search for Ddl-Sensitive Influence Minimization (DSIM)* algorithm to tackle the DsTimin problem. Specifically, in Algorithm 5 (DSIM), we pre-determine the input parameter maximum deadline T_m, at which negative influence (i.e., $\sigma(S_n, G[V \setminus B], T)$) first stabilizes, and initialize r, the left border of binary search to T_m (Line 1). In each iteration of the binary search (Lines 2–7), we employ Algorithm 2 to identify the optimal blocking set B^o with an approximation guarantee (Line 4). When the binary search concludes, we return the maximum deadline and corresponding blocking set (B^o, T^o) as the solution to DsTimin problem (Line 8).

Time Complexity and Theoretical Analysis. In DSIM algorithm, the binary search requires $O(\log T_m)$ rounds. As elaborated in § 3.2, the time complexity of TESTIM is $O(\frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T)}{\epsilon^2})$. Therefore, it demands $O(\frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T^o)}{\epsilon^2})$ in each round of the binary search, and in total $O(\log T_m \frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T^o)}{\epsilon^2})$ is required. Since the final solution of DsTimin is returned by TESTIM algorithm, as per Theorem 3, and TESTIM offers a $(1 - 1/e - \epsilon)$-approximation guarantee, it can be concluded that DSIM maintains the same theoretical guarantee.

Algorithm 6 BSIM

Input: $G(V, E)$, S_n, T, α, ϵ, δ
Output: $\hat{B}$

1: Initialize $\hat{B} \leftarrow \emptyset$, $l \leftarrow 0$, $r \leftarrow n$
2: **while** $l \leq r$ **do**
3: $\quad k' \leftarrow (l + r)/2$
4: $\quad \hat{B} \leftarrow$ TESTIM(G, S_n, k', T, ϵ, δ)
5: $\quad$ **if** $\sigma(S_n, G[V \setminus \hat{B}], T)/n = \alpha$ **then break**
6: $\quad$ **else if** $\sigma(S_n, G[V \setminus \hat{B}], T)/n < \alpha$ **then** $l \leftarrow k' + 1$
7: $\quad$ **else if** $\sigma(S_n, G[V \setminus \hat{B}], T)/n > \alpha$ **then** $r \leftarrow k' - 1$
8: **Return** $\hat{B}$

4.2 BsTimin Problem

Building upon the discussion of influence threshold (i.e., α) in social network in § 4.1, this subsection delves into another real-world observation: an excessive suspension of user accounts on social platforms may lead to user dissatisfaction [26, 45]. As a result, we explore the selection of a minimal blocking set to effectively control the spread of negative business below the threshold before a given deadline. Consequently, we extend the Timin problem and formally define the Budget-sensitive Timin (BsTimin) problem as follows.

Problem 3. **(BsTimin).** *Given a graph $G(V, E)$, a negative seed node set 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 only blocking $\hat{B}$, the proportion of the expected active nodes by the timestamp T remains below α, i.e.,*

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

Similarly, we propose *Binary Search for Budget-Sensitive Influence Minimization (BSIM)* algorithm to solve the BsTimin problem. Specifically, in Algorithm 6 (BSIM), we set the initial maximum size of blocking to n (Line 1). In each round of the binary search (Lines 2–7), Algorithm 2 is once again employed to find the optimal set $\hat{B}$, minimizing the spread of negative business (Line 4) while maintaining the approximation guarantee. Finally, we return the blocking set $\hat{B}$ as the solution for BsTimin problem (Line 8).

Time Complexity and Theoretical Analysis. In BSIM algorithm, it requires $O(\log n \frac{\max\{\ln \frac{1}{\delta}, n\} \cdot m \cdot R(\{v^*\}, T)}{\epsilon^2})$ time overall. Given that TESTIM offers a $(1 - 1/e - \epsilon)$-approximation guarantee, it follows that BSIM also maintains $(1 - 1/e - \epsilon)$ theoretical guarantee.

5 EXPERIMENTS

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

5.1 Experimental Settings

Datasets. In experiments, we employ five real-world networks from SNAP [21]. Table 1 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.

Table 1: Dataset Statistics

Dataset	$n = \|V\|$	$m = \|E\|$	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

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 [17] 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. As with [25, 33], we generate the influence time delay of each edge, following the *Poisson* distribution or *Geometric* distribution [8, 23]. Given space constraints, we default to the *Poisson* distribution and assess the effect of the *Geometric* distribution in a single experiment.

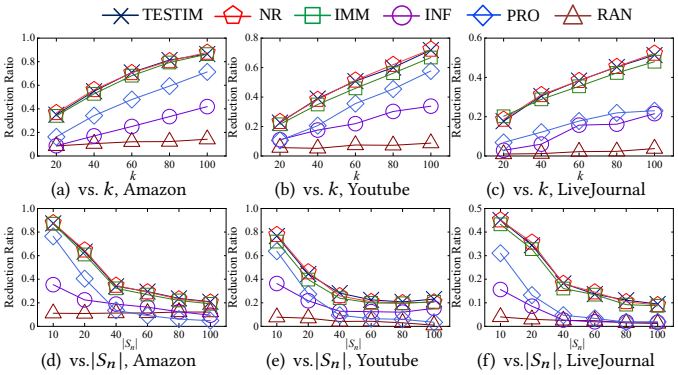
Following [33], we generate the negative seed nodes in two ways. The first way is to sample a small number of nodes from the top-k most influential nodes in the network to simulate the spread of negative influence by a few popular users on the network. The second way is to uniformly sample a large number of users at random from the network, simulating the coordinated propagation of negative influence orchestrated by numerous automated bots or newly established puppet accounts. Since these two types of negative seed nodes yield similar experimental results, we only report the results for the first type due to space constraints.

Algorithms. We test a set of algorithms in experiments.

- **TESTIM** and **NeighborReplace (NR)** are newly proposed approximate and heuristic algorithms respectively.
- **IMM** [40] is a state-of-the-art sampling-based method for influence maximization with approximation guarantee. We modify it to address the Timin problem by employing Algorithm 1 to choose blocking nodes.
- **Influential (INF)** [33] method selects blocking nodes in descending order of the expected influence of nodes.
- **Proximity (PRO)** [33, 43] method selects blocking nodes from the out-neighbors of negative seeds, with a preference for nodes connected by high-probability edges.
- **Random (RAN)** method uniformly selects blocking nodes at random.
- **DSIM** and **BSIM** are two methods proposed in this paper to address the DsTimin and BsTimin problems, respectively.

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

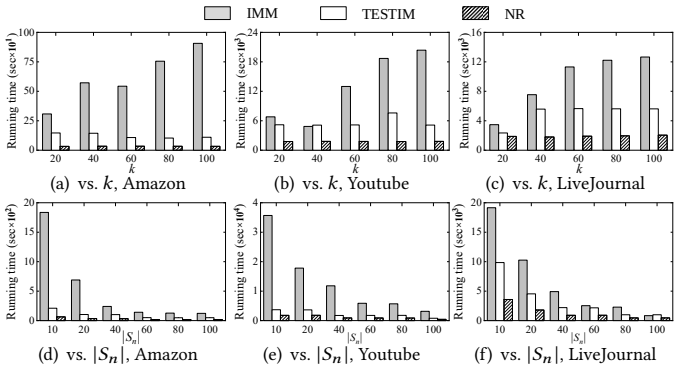
Parameters. We evaluate the performance of algorithms on different parameters, including the budget k, the number of negative seeds $|S_n|$, the deadline T, the maximal survival time $\varphi_{\max}$, and sampling error factor ϵ. In each experiment, we vary only one parameter and keep the others by their default values, i.e., $k = 50$, $|S_n| = 20$, $T = 32$, and $\varphi_{\max} = 8$. Following [1, 13], we set $\epsilon = 0.2$ for the *EmailCore* and *Epinions* datasets, and $\epsilon = 0.3$ for *Amazon*, *Youtube* and *LiveJournal* 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 considered algorithms.



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

(d) vs.$|S_n|$, Amazon (e) vs.$|S_n|$, Youtube (f) vs.$|S_n|$, LiveJournal

Figure 4: Reduction Ratio vs. $k/|S_n|$



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

(d) vs. $|S_n|$, Amazon (e) vs. $|S_n|$, Youtube (f) vs. $|S_n|$, LiveJournal

Figure 5: Running time vs. $k/|S_n|$

We use the number of nodes in the network as the initial number of T-RR sets in TESTIM and IMM when evaluating the running time.

Evaluation Metrics. We employ the *Reduction Ratio* (R^2) [32, 34] 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.

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

5.2 Evaluation of Timin Algorithms

In this set of experiments, we evaluate the performance of Timin algorithms. Note that, due to space limitation, we report the results of *Amazon*, *Youtube* and *LiveJournal*. Full experimental results can be found in [6].

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 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 1) to select the blocking set B, which provides theoretical guarantees. The experimental results also demonstrate the superiority of our 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 4, the time complexity of TESTIM is determined by the generation cost of RR-B sets. Additionally, we can observe that NR significantly improves the efficiency of 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 initial nodes influenced by negative information increases with the growth of 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 under various number of negative seeds $|S_n|$. The results show that TESTIM significantly outperforms IMM under these conditions, and NR achieves even faster performance than TESTIM.

Varying T. We explore the impact of the deadline parameter T on the reduction ratio. As illustrated in Figures 6(a)-6(c), it is observed that as T is extended, the reduction ratio for each algorithm initially increases and then stabilizes. This is because of the diminishing marginal influence gain of nodes affected by negative information over time in the diffusion process, resulting in a stable number of saved nodes when the deadline is sufficiently long. 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 Figures 6(d)-6(f). It can be observed that as φ_{max} ascends, the reduction ratio initially increases rapidly, and then stabilizes when φ_{max} becomes large (from 8 to 32). 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 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, as the activation process is executed faster, leading to more nodes being activated within T. In addition, under the *Geometric* distribution, our TESTIM and NR perform effectively, achieving 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. Figure 7 reveals that the reduction ratio

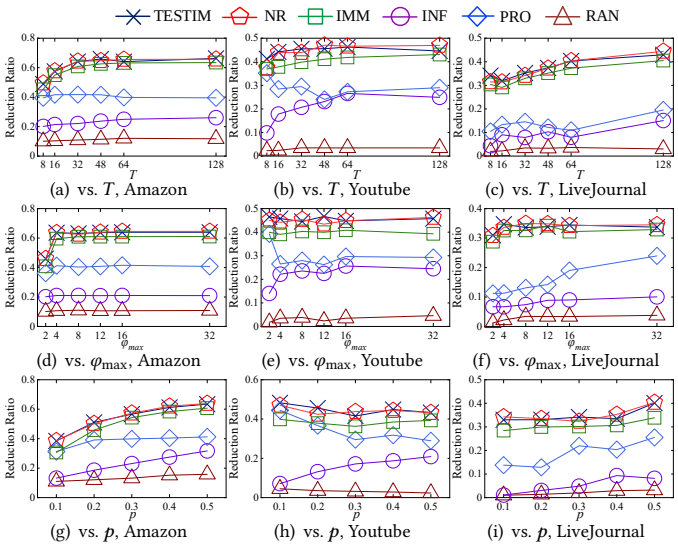


Figure 6: Reduction Ratio vs. $T/\varphi_{\max}/p$

remains relatively consistent when ϵ is varied. This is because the approximate guarantees of TESTIM and IMM represent worst-case performance, while their actual performance in real-world scenarios may be empirically robust. Hence, the experimental results highlight the resilience of *TRIS* technique to variations in ϵ. Moreover, we observe that TESTIM 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 due to early termination in TESTIM (Lines 12–13) and IMM, resulting in generating fewer RR-B sets. As per Theorem 4, 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, with $\epsilon = 0.1$, the execution time for IMM exceeds 24 hours, promoting us to terminate the process. This outcome further underscores the scalability of TESTIM on large datasets.

5.3 Evaluation of DsTimin and BsTimin Algorithms

In this set of experiments, we evaluate the impact of *threshold* α when addressing our DsTimin and BsTimin variant problems.

We conduct experiments to investigate the effect of threshold α, the percentage of negative influence spread in a network, on the solutions of the maximum deadline T and the minimum size of the blocking set $|B|$. Specifically, we set the number of negative seeds to be 5% of the total nodes for influential seeds and 10% for random seeds. When examining the maximum deadline T, we fix the number of blocking nodes at 1% of the total nodes in the networks. Note that the value of α must not exceed $\alpha_{\max}$, as detailed in Table 2. $\alpha_{\max}$ represents the percentage of negative influence spread in a network with no nodes blocked. This range varies among different networks due to their distinct structures. Table 2 provides the experimental results for the maximum T and the minimum $|B|$ when varying α. Due to space constraints, we only report the running time of DSIM and BSIM algorithms for the *LiveJournal*. The results indicate that the maximum deadline T increases as α increases. This suggests

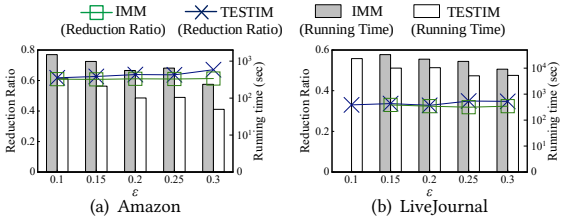


Figure 7: Reduction Ratio and Running time vs. ϵ

that with a more relaxed constraint on negative influence spread, platforms and policymakers have additional time to manage the adverse effects of negative information by spreading positive aspects, such as truth, clarification and medical treatments. Additionally, as α increases, the minimum number of required blocking nodes decreases. This indicates that under more lenient constraints, the spread of negative influence can be controlled with fewer blocking nodes, thereby alleviating user dissatisfaction. These experimental results demonstrate the effectiveness of our DSIM and BSIM algorithms in solving real-world variant problems.

5.4 Case Study

In last set of experiments, we visualize the propagation of negative influence within the *Facebook* social network ($n = 4,039$, $m = 88,234$), initiated by 20 influential negative seeds (highlighted in blue). Figure 8(a) illustrates the spread of negative influence under the classical LT model with zero nodes being blocked. It's evident that the influence propagation process concludes with nearly 60% of the total nodes being influenced (colored in red). This is because the classical LT model has no constraints, allowing negative influence to spread widely across the network. Furthermore, the influenced nodes primarily cluster around the negative seeds, forming a clustered distribution, as nodes near the negative seeds are more likely to be activated. Notably, nodes that remain inactive typically have sum of influence weights from their neighbors that do not surpass their activation thresholds. In our experiment, we find that these inactive nodes are mostly isolated nodes.

Figure 8(b) depicts the spread of negative influence under the classical LT model with 100 nodes blocked by our TESTIM algorithm. Notably, the inclusion of blocking nodes reduces the negative influence spread from 60% to 32%, clearly demonstrating the effectiveness of our blocking nodes selection strategy. In addition, most of the saved nodes (transitioning from red to white) are distributed around the negative seeds. This observation aligns with our discussion in § 3.3, suggesting that the out-neighbors of negative seeds are more likely to be chosen as blocking nodes, thereby further limiting the spread of negative influence to their neighbors.

Figure 8(c) shows the spread of negative influence under our TLT model, which incorporates temporal aspects into the classical LT model including deadline T, time-delay φ_t, and maximum survival time of influence weights $\varphi_{\max}$. We employ our TESTIM algorithm to block 100 nodes, using the default temporal parameter settings. Under these temporal constraints and with the blocking set in place, we observe a reduction in negative influence spread from 32% to 22%. Several factors contribute to this reduction: **(1)** nodes influenced by negative influence after the deadline are considered invalid in Timin problem, **(2)** the time-delay on each edge decelerates node activation, further slowing the spread rate of negative influence,

Table 2: The DSIM and BSIM performance vs. α

EmailCore								
S_n	Influential (0.72)				Random (0.60)			
α	0.40	0.50	0.60	0.70	0.40	0.45	0.50	0.55
T	6	9	13	27	7	9	13	16
$\|B\|$	85	52	32	17	46	35	26	18

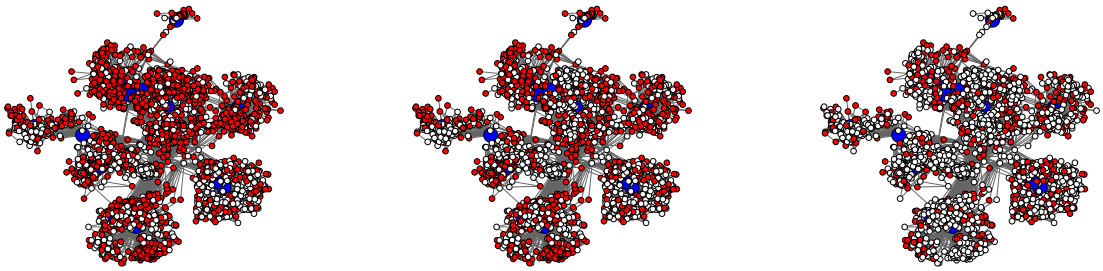
Epinions								
S_n	Influential (0.40)				Random (0.53)			
α	0.25	0.30	0.35	0.40	0.15	0.20	0.25	0.30
T	2	3	4	6	2	3	4	9
$\|B\|$	6,052	4,054	2,819	1,611	6,927	4,007	2,347	1,103

Amazon								
S_n	Influential (0.45)				Random (0.32)			
α	0.25	0.30	0.35	0.40	0.35	0.40	0.45	0.50
T	6	9	13	27	4	6	13	27
$\|B\|$	17,588	12,291	8,333	5,191	17,745	12,713	8,826	5,667

Youtube								
S_n	Influential (0.69)				Random (0.45)			
α	0.50	0.55	0.60	0.65	0.30	0.35	0.40	0.45
T	3	4	6	7	3	4	6	13
$\|B\|$	74,329	55,227	36,476	21,666	75,510	46,948	28,072	12,649

LiveJournal								
S_n	Influential (0.50)				Random (0.57)			
α	0.40	0.44	0.46	0.48	0.50	0.52	0.54	0.56
T	11	14	26	27	12	13	26	27
$\|B\|$	124,337	93,591	80,404	68,335	99,882	84,744	71,168	59,139

LiveJournal (running time (sec))								
S_n	Influential				Random			
α	0.40	0.44	0.46	0.48	0.50	0.52	0.54	0.56
T	376.331	441.173	412.526	415.476	369.814	428.335	471.903	299.531
$\|B\|$	197.155	195.095	190.556	189.811	217.161	216.414	190.077	202.091



(a) Case 1 (LT model, $B = \emptyset$) (b) Case 2 (LT model, $|B| = 100$) (c) Case 3 (TLT model, $|B| = 100$)

Figure 8: Visualization of the spread of the negative influence (red nodes) in the Facebook social network initiated by 20 influential negative seeds (blue nodes)

and **(3)** when the time-delay exceeds the maximum survival time, influence weights drop to zero, making it harder to activate nodes. Moreover, we find that 55% of the blocked nodes under the TLT model overlap with those blocked under the LT model, indicating differences between the two blocking sets in these two different information propagation models. Further experimental analysis revealed that over 90% of the blocking nodes are neighbors of the negative seeds, proving additional evidence in support of our discussion in § 3.3 under the TLT model.

6 RELATED WORK

Influence Maximization. The problem of Influence Maximization (IM) was initially formulated as a discrete optimization problem by Kempe et al. [17]. They introduced two fundamental propagation models, i.e., the IC and LT models, proved that the IM problem is **NP**-hard under both models, and proposed a greedy algorithm with $(1 - 1/e)$-approximation. Subsequently, substantial follow-up research has focused on developing efficient and scalable solutions for the IM problem [12, 15, 38, 40, 41]. Many of these approaches rely on reverse influence sampling methods [3]. Meanwhile, researchers have explored various variants of IM [2, 13, 39, 50]. In this paper, we focus on the influence minimization problem.

Influence Minimization. Extensive research has focused on the Influence Minimization (IMIN) problem, which aims to minimize the spread of negative information/disease in social networks, as surveyed in [36, 49]. Existing IMIN solutions fall into two categories [42]. **(1)** Clarification-based methods [4, 10, 14, 24, 33, 34] aim to minimize the spread of negative information by selecting nodes to disseminate positive information. [4] introduced centrality-based heuristics to identify effective positive spreaders. [34] and [24] used login events to simulate information diffusion delays, thus minimizing rumor influence before a specific deadline. [33] considered differential propagation rates between truth and misinformation and user reaction times when mitigating misinformation spread. **(2)** Blocking-based methods [7, 29, 30, 32, 37, 45–47] include node blocking [7, 32, 45–47] and edge blocking [18–20, 30, 48]. For node blocking, [47] proposed a novel graph sampling technique that incorporates the dominator tree structure to address IMIN problem, [46] studied hindering the influence spread of a community by removing nodes based on a novel interaction frequency metric, and [32] studied the adaptive IMIN problem by adaptively selecting blocking nodes based on real-time observations of negative influence spread. For edge blocking, [19] proposed an approximate algorithm to greedily block edges, while [20] developed heuristic algorithms for edge blocking under a simpler deterministic variant of the LT model. Khalil et al. [18] provided theoretical guarantee algorithms to add/delete a small set of edges using live-edge graphs when modifying network under the LT model. However, none of the existing studies that utilized blocking-based methods have taken into account the influence time-delay, where our work differs from existing work.

7 CONCLUSION

In this paper, we study the TIMIN problem via temporarily blocking key nodes in social networks. We first introduce a novel TLT model to simulate the negative influence diffusion process considering time delay. Then, we define the TIMIN problem, which aims to identify a set of blocking nodes to minimize the negative influence w.r.t. a deadline. We prove the TIMIN problem is **NP**-hard, monotone, and supermodular. We develop a TIMIN-Greedy algorithm with $(1 - 1/e)$ approximation, propose its scalable version TESTIM with $(1 - 1/e - \epsilon)$ approximation, which incorporates a novel temporal reverse influence sampling technique, and devise an efficiency-improving heuristic algorithm NeighborReplace. Moreover, considering time and budget constraints in influence minimization problem, we introduce two variants of the TIMIN problem and propose two methods with $(1-1/e-\epsilon)$ approximation. Extensive experiments on five real-world networks and a visualization case study demonstrate the superiority of our algorithms in effectiveness, efficiency, and scalability.

REFERENCES

[1] Cigdem Aslay, Francesco Bonchi Laks VS Lakshmanan, and Wei Lu. 2017. Revenue Maximization in Incentivized Social Advertising. *Proceedings of the VLDB Endowment* 10, 11 (2017).

[2] Prithu Banerjee, Wei Chen, and Laks VS Lakshmanan. 2019. 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*. 1078–1095.

[3] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing social influence in nearly optimal time. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 946–957.

[4] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. 2011. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*. 665–674.

[5] Badrish Chandramouli, Jonathan Goldstein, and Songyun Duan. 2012. Temporal analytics on big data for web advertising. In *2012 IEEE 28th international conference on data engineering (ICDE)*. 90–101.

[6] Xueqin Chang, Jiajie Fu, Qing Liu, Yunjun Gao, and Baihua Zheng. 2023. Time-aware Influence Minimization via Blocking Social Networks. https://github.com/ZJU-DAILY/TIMIN

[7] Chen Chen, Hanghang Tong, B Aditya Prakash, Charalampos E Tsourakakis, Tina Eliassi-Rad, Christos Faloutsos, and Duen Horng Chau. 2015. Node immunization on large graphs: Theory and algorithms. *IEEE Transactions on Knowledge and Data Engineering* 28, 1 (2015), 113–126.

[8] Wei Chen, Wei Lu, and Ning Zhang. 2012. Time-critical influence maximization in social networks with time-delayed diffusion process. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 591–598.

[9] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *2010 IEEE International Conference on Data Mining (ICDM)*. 88–97.

[10] Qizhi Fang, Xin Chen, Qingqin Nong, Zongchao Zhang, Yongchang Cao, Yan Feng, Tao Sun, Suning Gong, and Dingzhu Du. 2020. General rumor blocking: An efficient random algorithm with martingale approach. *Theoretical Computer Science* 803 (2020), 82–93.

[11] Amit Goyal, Francesco Bonchi, and Laks VS Lakshmanan. 2010. Learning influence probabilities in social networks. In *Proceedings of the third ACM international conference on Web search and data mining*. 241–250.

[12] Qintian Guo, Sibo Wang, Zhewei Wei, Wenqing Lin, and Jing Tang. 2022. Influence Maximization Revisited: Efficient Sampling with Bound Tightened. *ACM Transactions on Database Systems (TODS)* (2022).

[13] Kai Han, Benwei Wu, Jing Tang, Shuang Cui, Cigdem Aslay, and Laks VS Lakshmanan. 2021. Efficient and effective algorithms for revenue maximization in social advertising. In *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*. 671–684.

[14] Xinran He, Guojie Song, Wei Chen, and Qingye Jiang. 2012. Influence blocking maximization in social networks under the competitive linear threshold model. In *Proceedings of the 2012 siam international conference on data mining*. SIAM, 463–474.

[15] Keke Huang, Sibo Wang, Glenn Bevilacqua, Xiaokui Xiao, and Laks VS Lakshmanan. 2017. Revisiting the stop-and-stare algorithms for influence maximization. *Proceedings of the VLDB Endowment* 10, 9 (2017), 913–924.

[16] Tianyuan Jin, Yu Yang, Renchi Yang, Jieming Shi, Keke Huang, and Xiaokui Xiao. 2021. Unconstrained submodular maximization with modular costs: Tight approximation and application to profit maximization. *Proceedings of the VLDB Endowment* 14, 10 (2021), 1756–1768.

[17] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 137–146.

[18] Elias Boutros Khalil, Bistra Dilkina, and Le Song. 2014. Scalable diffusion-aware optimization of network topology. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1226–1235.

[19] Masahiro Kimura, Kazumi Saito, and Hiroshi Motoda. 2008. Minimizing the spread of contamination by blocking links in a network.. In *AAAI*, Vol. 8. 1175–1180.

[20] Chris J Kuhlman, Gaurav Tuli, Samarth Swarup, Madhav V Marathe, and SS Ravi. 2013. Blocking simple and complex contagion by edge removal. In *2013 IEEE 13th international conference on data mining*. IEEE, 399–408.

[21] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.

[22] Dyani Lewis. 2022. Why the WHO took two years to say COVID is airborne. *Nature* 604, 7904 (2022), 26–31.

[23] Bo Liu, Gao Cong, Yifeng Zeng, Dong Xu, and Yeow Meng Chee. 2013. Influence spreading path and its application to the time constrained social influence maximization problem and beyond. *IEEE Transactions on Knowledge and Data Engineering* 26, 8 (2013), 1904–1917.

[24] Mohammad Ali Manouchehri, Mohammad Sadegh Helfroush, and Habibollah Danyali. 2021. Temporal rumor blocking in online social networks: A sampling-based approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52, 7 (2021), 4578–4588.

[25] Xiaoye Miao, Huanhuan Peng, Kai Chen, Yuchen Peng, Yunjun Gao, and Jianwei Yin. 2022. Maximizing Time-aware Welfare for Mixed Items. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 1044–1057.

[26] Andrew Cesare Miller. 2022. # DictatorErdogan: How Social Media Bans Trigger Backlash. *Political Communication* 39, 6 (2022), 801–825.

[27] Michael Mitzenmacher and Eli Upfal. 2017. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press.

[28] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming* 14 (1978), 265–294.

[29] Hung T Nguyen, Alberto Cano, Tam Vu, and Thang N Dinh. 2019. Blocking self-avoiding walks stops cyber-epidemics: a scalable gpu-based approach. *IEEE Transactions on Knowledge and Data Engineering* 32, 7 (2019), 1263–1275.

[30] Canh V Pham, Hoang M Dinh, Hoa D Nguyen, Huyen T Dang, and Huan X Hoang. 2017. Limiting the spread of epidemics within time constraint on online social networks. In *Proceedings of the 8th International Symposium on Information and Communication Technology*. 262–269.

[31] MIT Technology Review. 2018. A mathematical model captures the political impact of fake news. Retrieved August 1, 2023 from https://www.technologyreview.com/2018/09/11/2098/a-mathematical-model-captures-the-political-impact-of-fake-news/

[32] Qihao Shi, Can Wang, Deshi Ye, Jiawei Chen, Yan Feng, and Chun Chen. 2019. Adaptive influence blocking: Minimizing the negative spread by observation-based policies. In *2019 IEEE 35th international conference on data engineering (ICDE)*. 1502–1513.

[33] Michael Simpson, Farnoosh Hashemi, and Laks VS Lakshmanan. 2022. Misinformation mitigation under differential propagation rates and temporal penalties. *Proceedings of the VLDB Endowment* 15, 10 (2022), 2216–2229.

[34] Chonggang Song, Wynne Hsu, and Mong Li Lee. 2017. Temporal influence blocking: Minimizing the effect of misinformation in social networks. In *2017 IEEE 33rd international conference on data engineering (ICDE)*. 847–858.

[35] Statista. 2020. 2020 Presidential Election and the Media - Statistics Facts. Retrieved March 27, 2023 from https://www.statista.com/topics/5934/2020-presidential-election-and-the-media/#topicOverview

[36] Ling Sun, Yuan Rao, Lianwei Wu, Xiangbo Zhang, Yuqian Lan, and Ambreen Nazir. 2023. Fighting False Information from Propagation Process: A Survey. *Comput. Surveys* 55, 10 (2023), 1–38.

[37] Lichao Sun, Xiaobin Rui, and Wei Chen. 2023. Scalable Adversarial Attack Algorithms on Influence Maximization. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 760–768.

[38] Jing Tang, Xueyan Tang, Xiaokui Xiao, and Junsong Yuan. 2018. Online processing algorithms for influence maximization. In *Proceedings of the 2018 International Conference on Management of Data*. 991–1005.

[39] Jing Tang, Xueyan Tang, and Junsong Yuan. 2017. Profit maximization for viral marketing in online social networks: Algorithms and analysis. *IEEE Transactions on Knowledge and Data Engineering* 30, 6 (2017), 1095–1108.

[40] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence maximization in near-linear time: A martingale approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 1539–1554.

[41] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. 75–86.

[42] Saravanan Thirumuruganathan, Michael Simpson, and Laks VS Lakshmanan. 2021. To intervene or not to intervene: Cost based intervention for combating fake news. In *Proceedings of the 2021 International Conference on Management of Data*. 2300–2309.

[43] Amo Tong, Ding-Zhu Du, and Weili Wu. 2018. On misinformation containment in online social networks. In *Advances in neural information processing systems*. 339–349.

[44] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *science* 359, 6380 (2018), 1146–1151.

[45] Biao Wang, Ge Chen, Luoyi Fu, Li Song, and Xinbing Wang. 2017. Drimux: Dynamic rumor influence minimization with user experience in social networks. *IEEE Transactions on Knowledge and Data Engineering* 29, 10 (2017), 2168–2181.

[46] Jiadong Xie. 2022. Hindering Influence Diffusion of Community. In *Proceedings of the 2022 International Conference on Management of Data*. 2518–2520.

[47] Jiadong Xie, Fan Zhang, Kai Wang, Xuemin Lin, and Wenjie Zhang. 2023. Minimizing the Influence of Misinformation via Vertex Blocking. In *2023 IEEE 39th international conference on data engineering (ICDE)*. 789–801.

[48] Ruidong Yan, Yi Li, Weili Wu, Deying Li, and Yongcai Wang. 2019. Rumor blocking through online link deletion on social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 13, 2 (2019), 1–26.

[49] Ahmad Zareie and Rizos Sakellariou. 2021. Minimizing the spread of misinformation in online social networks: A survey. *Journal of Network and Computer Applications* 186 (2021), 103094.
[50] Yipeng Zhang, Yuchen Li, Zhifeng Bao, Baihua Zheng, and HV Jagadish. 2021. Minimizing the regret of an influence provider. In *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*. 2115–2127.

APPENDIX

A PROOF OF LEMMA 4

Proof. Before proving Lemma 4, we first introduce *Chernoff Inequalities* [27] in Lemma 5 as follows.

Lemma 5. **(Chernoff Inequalities [27]).** *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 $\lambda > 0$,*

$$\Pr[X - k\mu \geq \lambda \cdot k\mu] \leq \exp\left(-\frac{\lambda^2}{2+\lambda}k\mu\right) \tag{10}$$

$$\Pr[X - k\mu \leq -\lambda \cdot k\mu] \leq \exp\left(-\frac{\lambda^2}{2}k\mu\right) \tag{11}$$

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:

$$\begin{aligned}&\Pr[g^{\mathcal{R}_{b2}}(B,T) - R(B,T) \geq \epsilon_1 \cdot R(B,T)]\\ &\leq \exp\left(-\frac{\epsilon_1^2}{2+\epsilon_1}\frac{|\mathcal{R}_{b2}|}{f^{\mathcal{R}_{t2}}(S_n, G[V],T)}R(B,T)\right)\end{aligned} \tag{12}$$

$$\begin{aligned}&\Pr[g^{\mathcal{R}_{b1}}(B^*,T) - R(B^*,T) \leq -\epsilon_2 \cdot R(B^*,T)]\\ &\geq \exp\left(-\frac{\epsilon_2^2}{2}\frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n, G[V],T)}R(B^*,T)\right)\end{aligned} \tag{13}$$

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. (8) holds, and Ω_{2i} denote the event that Eq. (9) holds. We set ϵ' and $\bar{\epsilon}$ as the solutions to Eq. (12) and Eq. (13) respectively, thus we have following equations:

$$\exp\left(-\frac{(\epsilon')^2}{2+\epsilon'}\frac{|\mathcal{R}_{b2}|}{f^{\mathcal{R}_{t2}}(S_n, G[V],T)}R(B,T)\right) = \frac{\delta}{5i^2}. \tag{14}$$

$$\exp\left(-\frac{(\bar{\epsilon})^2}{2}\frac{|\mathcal{R}_{b1}|}{f^{\mathcal{R}_{t1}}(S_n, G[V],T)}R(B^*,T)\right) = \frac{\delta}{5i^2}. \tag{15}$$

Then 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

$$\begin{aligned}\Pr\left[\bigcap_{i=1}^{\infty}\Omega_{1i}\bigcap_{i=1}^{\infty}\Omega_{2i}\right] &\geq \prod_{i=1}^{\infty}\Pr\left[\Omega_{1i}\cap\Omega_{2i}\right] \geq \prod_{i=1}^{\infty}\left(1-\frac{2\delta}{5i^2}\right)\\ &\geq 1 - \sum_{i=1}^{\infty}\frac{2\delta}{5i^2} \geq 1 - \frac{\pi^2\delta}{15} \geq 1 - \frac{2\delta}{3}.\end{aligned} \tag{16}$$

The details of proof are similar in spirit to those in [16]. □

With the above conclusions we further prove Theorem 3.

B PROOF OF THEOREM 3

Proof. We consider two cases that depend on whether Line 12 in Algorithm 2 is satisfied.

Case 1: Line 12 is satisfied. Then in the last iteration, we have

$$\lambda \leq \frac{1-1/e}{1-1/e-\epsilon}\cdot\frac{1-\epsilon_2}{1+\epsilon_1} \tag{17}$$

where $\epsilon_1, \epsilon_2 \in (0,1)$ and $\lambda > 0$. Suppose that both Eq. (8) and Eq. (9) hold. Then,

$$\begin{aligned}g^{\mathcal{R}_{b1}}(B,T) &\geq (1-1/e)g^{\mathcal{R}_{b1}}(B^*,T)\\ &\geq (1-1/e)(1-\epsilon_2)R(B^*,T)\end{aligned} \tag{18}$$

where the first inequality is due to Lemma 3, and the second inequality is due to Eq. (9). Afterwards, via Eq. (8) we have:

$$g^{\mathcal{R}_{b1}}(B,T) = \lambda g^{\mathcal{R}_{b2}}(B,T) \leq \lambda(1+\epsilon_1)R(B,T) \tag{19}$$

Finally, we have

$$\begin{aligned}R(B,T) &\geq \frac{1}{\lambda(1+\epsilon_1)}g^{\mathcal{R}_{b1}}(B,T)\\ &\geq \frac{(1-1/e)(1-\epsilon_2)}{\lambda(1+\epsilon_1)}R(B^*,T)\\ &\geq (1-1/e-\epsilon)R(B^*,T)\end{aligned} \tag{20}$$

where the first inequality is from Eq. (19), the second inequality is from Eq. (18), and the third inequality is from Eq. (17). By Lemma 4, when Line 12 in Algorithm 2 is satisfied, with probability at least $1 - \frac{2\delta}{3}$, Theorem 3 holds.

Case 2: Line 12 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

$$g^{\mathcal{R}_{b1}} R(B^*, T) \geq (1 - \frac{\epsilon}{2}) R(B^*, T) \,, \tag{21}$$

$$g^{\mathcal{R}_{b1}} R(B, T) \leq R(B, T) + \frac{\epsilon}{2} R(B^*, T). \tag{22}$$

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

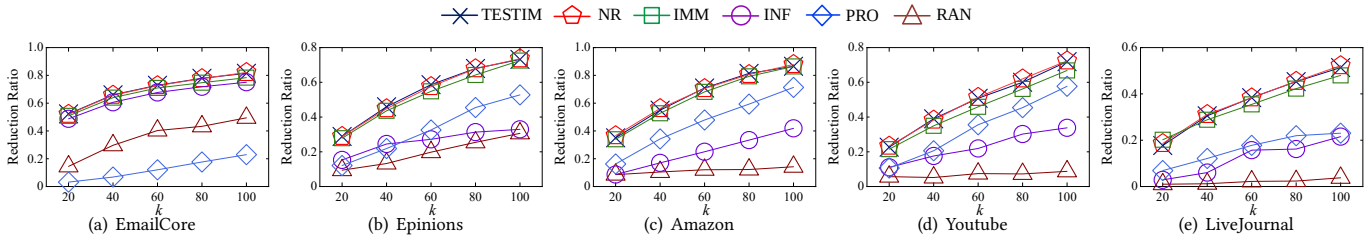
$$\begin{aligned}
R(B, T) &\geq g^{\mathcal{R}_{b1}} R(B, T) - \frac{\epsilon}{2} R(B^*, T) \\
&\geq (1 - 1/e) g^{\mathcal{R}_{b1}} R(B^*, T) - \frac{\epsilon}{2} R(B^*, T) \\
&\geq (1 - 1/e)(1 - \frac{\epsilon}{2}) R(B^*, T) - \frac{\epsilon}{2} R(B^*, T) \\
&= (1 - 1/e - \frac{\epsilon}{2} + \frac{\epsilon}{2e})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

According to Eq. (16), the event $\bigcap_i \Omega_{1i}$ happens with probability at least $1 - \frac{2\delta}{3}$. Hence, when Line 12 is not satisfied, with probability at least $1 - \frac{2\delta}{3} - \frac{\delta}{3} \geq 1 - \delta$, we have

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

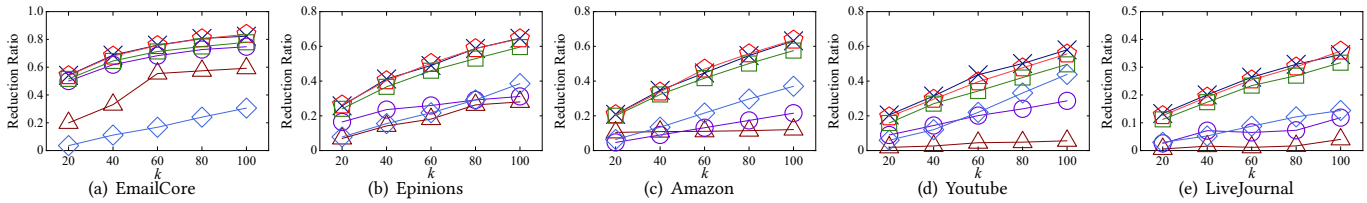
Finally, we combine **Case 1** and **Case 2**, the Theorem 3 is demonstrated. □

C FULL EXPERIMENT RESULTS



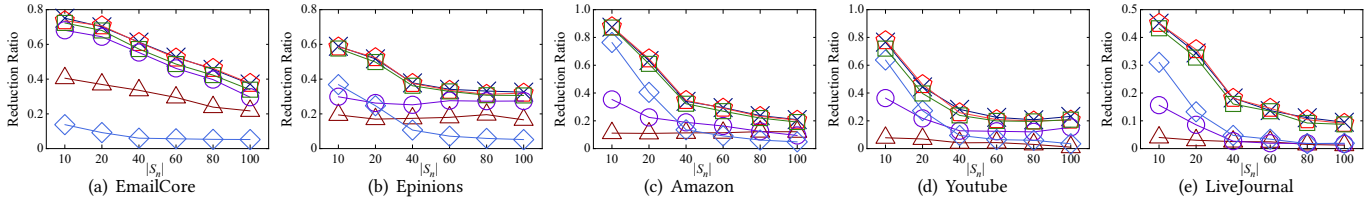
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 9: Reduction Ratio vs. k (influential negative seeds)



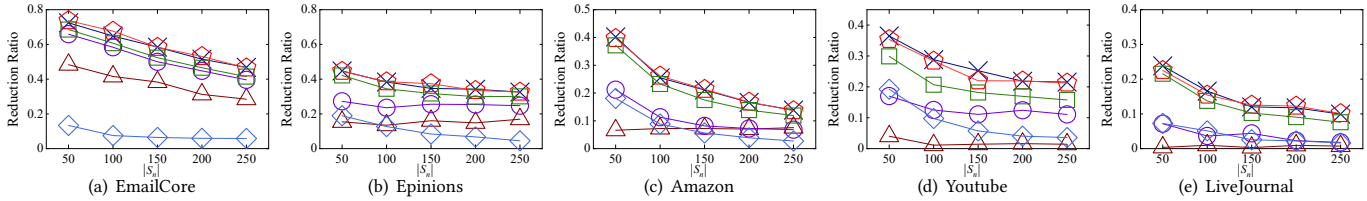
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 10: Reduction Ratio vs. k (random misinformation seeds)



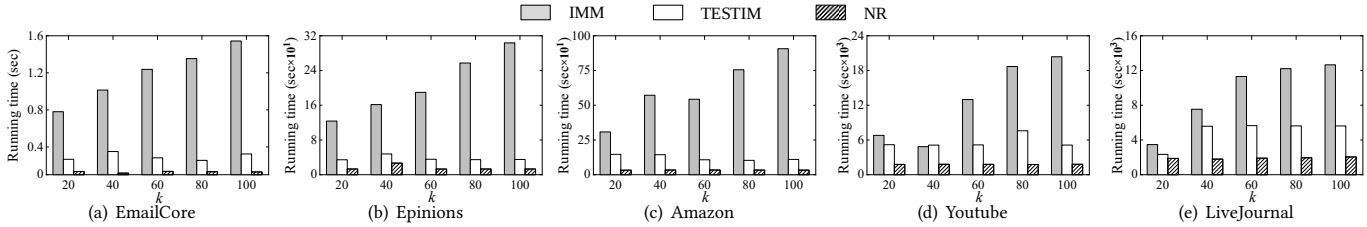
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 11: Reduction Ratio vs. $|S_n|$ (influential negative seeds)



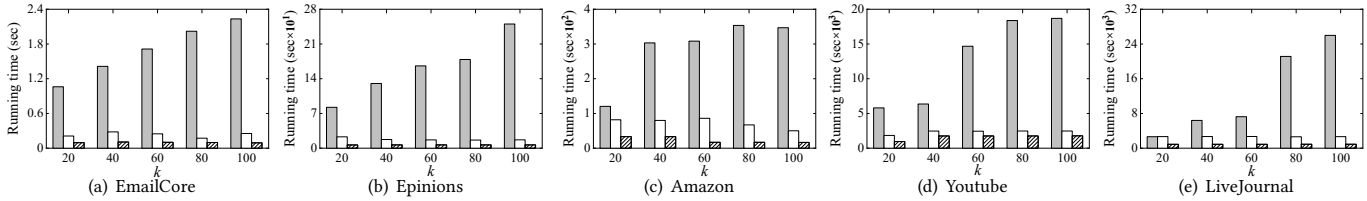
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 12: Reduction Ratio vs. $|S_n|$ (random misinformation seeds)



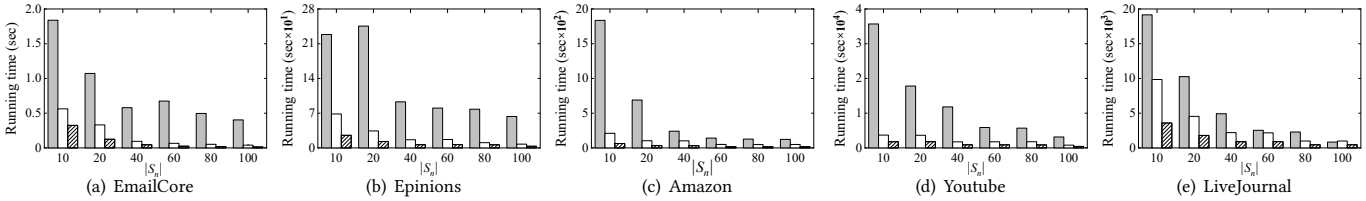
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 13: Running time vs. k (influential negative seeds)



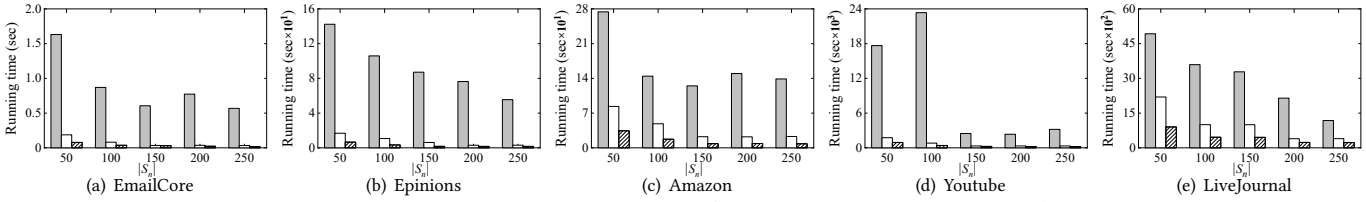
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 14: Running time vs. k (random misinformation seeds)



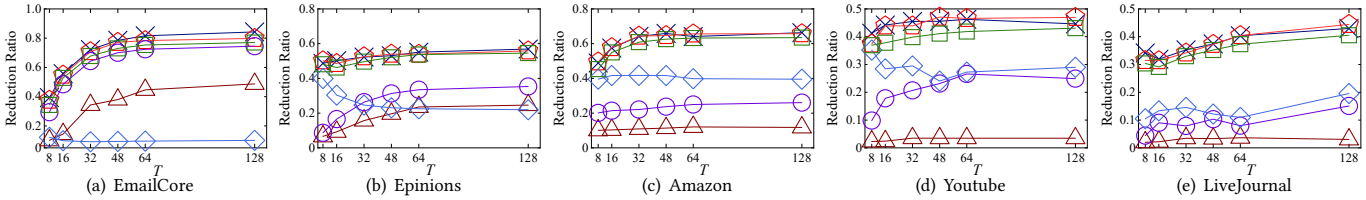
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 15: Running time vs. $|S_n|$ (influential negative seeds)



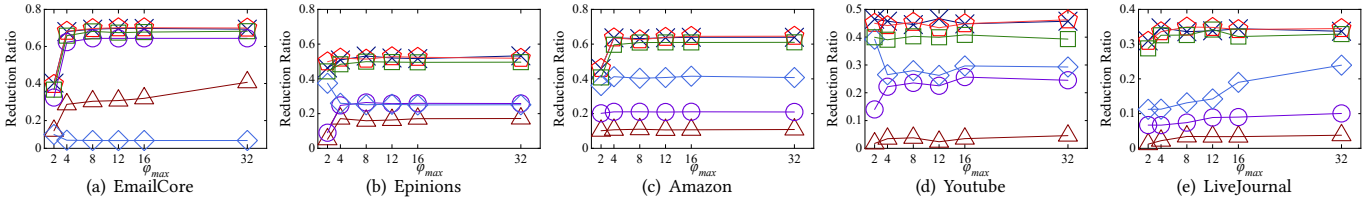
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 16: Running time vs. $|S_n|$ (random misinformation seeds)



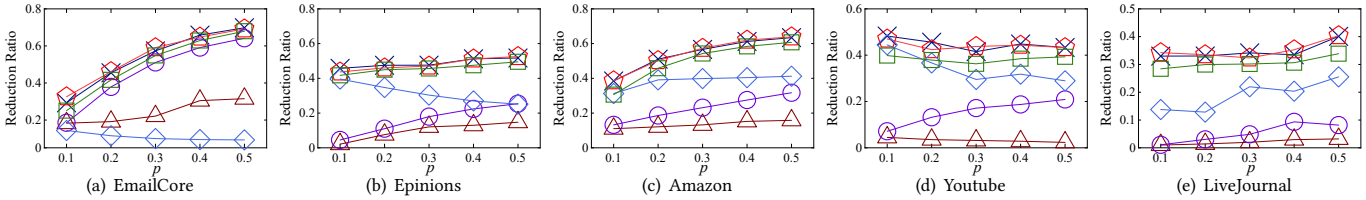
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 17: Reduction Ratio vs. T (influential negative seeds)



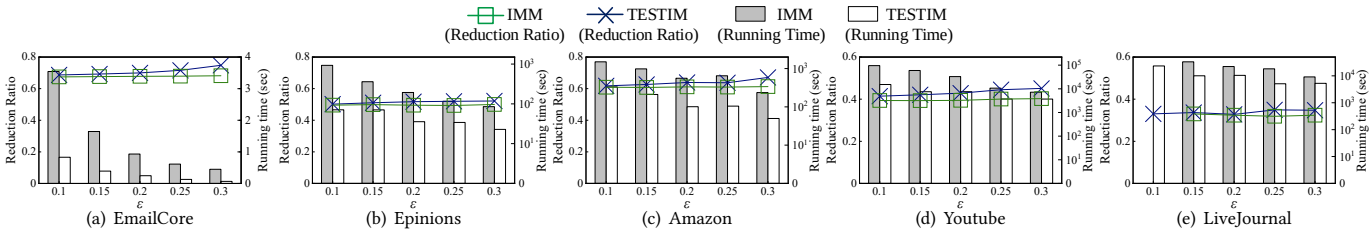
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 18: Reduction Ratio vs. $\varphi_{\max}$ (influential negative seeds)



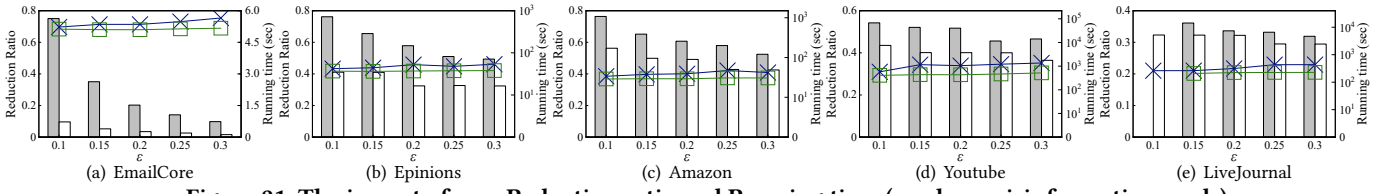
(a) EmailCore (b) Epinions (c) Amazon (d) Youtube (e) LiveJournal

Figure 19: Reduction Ratio vs. p (influential negative seeds)



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

Figure 20: Reduction Ratio and Running time vs. ϵ (influential negative seeds)



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

Figure 21: The impact of ϵ on Reduction ratio and Running time (random misinformation seeds)