Influence Maximization in Near-Linear Time: A Martingale Approach

Youze Tang Yanchen Shi Xiaokui Xiao
School of Computer Engineering
Nanyang Technological University
Singapore
tangyouze@gmail.com shiy0017@e.ntu.edu.sg xkxiao@ntu.edu.sg

ABSTRACT

Given a social network G and a positive integer k, the *influence maximization* problem asks for k nodes (in G) whose adoptions of a certain idea or product can trigger the largest expected number of follow-up adoptions by the remaining nodes. This problem has been extensively studied in the literature, and the state-of-the-art technique runs in $O((k+\ell)(n+m)\log n/\varepsilon^2)$ expected time and returns a $(1-1/e-\varepsilon)$-approximate solution with at least $1-1/n^\ell$ probability.

This paper presents an influence maximization algorithm that provides the same worst-case guarantees as the state of the art, but offers significantly improved empirical efficiency. The core of our algorithm is a set of estimation techniques based on *martingales*, a classic statistical tool. Those techniques not only provide accurate results with small computation overheads, but also enable our algorithm to support a larger class of information diffusion models than existing methods do. We experimentally evaluate our algorithm against the states of the art under several popular diffusion models, using real social networks with up to 1.4 billion edges. Our experimental results show that the proposed algorithm consistently outperforms the states of the art in terms of computation efficiency, and is often orders of magnitude faster.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining

Keywords

Influence maximization; sampling

1. INTRODUCTION

Given a social network G and a positive integer k, the *influence maximization* problem asks for k nodes (in G) whose adoptions of a certain idea or product can trigger the largest expected number of follow-up adoptions by the remaining nodes. This problem originates from *viral marketing* [14, 31], where a company provides free samples of a product to a set of *influential* individuals in a social network, aiming to create a cascade of product adoptions via the word-of-mouth effect. Kempe et al. [25] formulate influence maximization as a combinatorial optimization problem, and show that it is NP-hard; as a solution, they propose a greedy approach that yields $(1-1/e-\varepsilon)$-approximations under several formal models of information diffusion. Since then, a plethora of techniques [4–9,15,18,19,24–27,29,32,33,37] have been proposed for efficient influence maximization in large social networks.

Most existing techniques, however, either trade approximation guarantees for practical efficiency, or vice versa. In particular, methods [5,8,20,25,29] that provide $(1-1/e-\varepsilon)$-approximation solutions often require days to process small social networks with several thousands of nodes and edges; on the other hand, techniques that offer empirical efficiency rely on heuristics, due to which they are unable to offer any worst-case performance guarantee. The only exception is two recent techniques called *TIM* and *TIM*$^+$ [33]: both techniques run in $O((k+\ell)(n+m)\log n/\varepsilon^2)$ expected time and return $(1-1/e-\varepsilon)$-approximations with at least $1-1/n^\ell$ probability, where n and m are the numbers of nodes and edges, respectively, in the social network; furthermore, their empirical performance is shown to be comparable to the state-of-the-art heuristic solutions.

Although *TIM* and *TIM*$^+$ offer strong theoretical guarantees and good practical efficiency, they still leave much room for improvement in terms of their computational costs. Specifically, *TIM* consists of two phases, namely, *parameter estimation* and *node selection*. The node selection phase samples a number θ of node sets from the social network G, and then utilizes them to derive a size-k node set with a large expected influence. The value of θ is crucial: to ensure a $(1-1/e-\varepsilon)$-approximate solution, it is shown that θ should be at least λ/OPT [33], where OPT is the maximum expected influence of any size-k node set, and λ is a function of k, ℓ, n, and ε. To derive an appropriate θ, the parameter estimation phase of *TIM* first computes a lower bound of OPT, and then sets θ as λ over the lower bound. This approach, albeit intuitive, has two deficiencies. First, the lower bound derived by *TIM* is rather conservative and can be n/k times smaller than OPT in the worst case. As a consequence, θ could be excessively large, resulting in unnecessary overheads in the node selection phase. Second, the computation of the lower bound itself is costly – as we show in Section 6, the time required for deriving the lower bound can be more than 3 times longer than that required for node selection.

TIM$^+$ improves upon *TIM* by adding an intermediate step (between parameter estimation and node selection) that heuristically refines θ into a tighter lower bound of OPT, which leads to higher efficiency. However, in the worst case, even the improved lower bound may still be as small as $\frac{k}{n}OPT$. Furthermore, adding the in-

SIGMOD'15, May 31 - June 04, 2015, Melbourne, VIC, Australia.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2758-9/15/05 ...$15.00.
http://dx.doi.org/10.1145/2723372.2723734.

termediate step does not address the significant overheads incurred by the node selection phase.

Contributions. The paper presents *IMM*[1], an influence maximization algorithm that overcomes the deficiencies of *TIM* and *TIM*$^+$, while retaining their superior approximation guarantee and expected time complexity. The core of *IMM* is a set of estimation techniques based on *martingales* [38], a classic statistical tool. With those techniques, *IMM* adopts the two-phase paradigm of *TIM*, but incorporates a drastically different parameter estimation phase, which is able to derive a lower bound of OPT that is asymptotically tight. Specifically, the lower bound obtained by *IMM* is no less than $OPT \cdot (1-1/e)/(1+\varepsilon')^2$ with a high probability, where ε' is a tunable parameter. In addition, the computation of the lower bound is optimized, so that it incurs a considerably smaller overhead than in the case of *TIM* and *TIM*$^+$. Furthermore, the node selection phase of *IMM* is significantly improved with a martingale approach, which helps eliminate a large amount of unnecessary computation that is otherwise unavoidable. With the above optimizations, *IMM* significantly outperforms *TIM* and *TIM*$^+$ in terms of practical performance. In particular, we experimentally evaluate *IMM* against *TIM* and *TIM*$^+$ on a variety of social networks with up to 1.4 billion edges, and our results show that *IMM*'s running time is up to two orders of magnitude smaller than that of *TIM* and *TIM*$^+$. Moreover, when we set $\varepsilon = 0.5$ and $\ell = 1$, *IMM* even yields higher efficiency than the state-of-the-art heuristic solutions, while still offering empirically accurate results.

In addition, we show that *IMM* can be extended to support any diffusion model for which a certain sampling procedure is well-defined. As an application, we present a variant of *IMM* for the *continuous-time independent cascade (CTIC)* model [6, 16, 30], which is widely adopted in the machine learning literature [15, 17, 18] but not supported by *TIM*. We compare our approach with the most advanced method under the CTIC model, and demonstrate that *IMM* is vastly superior in both asymptotic and practical performance.

In summary, our contributions are as follows:

1. We propose *IMM*, an influence maximization algorithm runs in $O((k+\ell)(n+m)\log n/\varepsilon^2)$ expected time and returns a $(1-1/e-\varepsilon)$-approximation with at least $1-1/n^\ell$ probability under the triggering model [25], a general diffusion model adopted by most existing work. (Section 3)

2. We investigate the conditions under which *IMM* can be extended to a larger class of diffusion models, while still retaining the $(1-1/e-\varepsilon)$-approximation ratio. As an application, we present an extended version of *IMM* that supports the CTIC model [16]. (Section 4)

3. We experimentally evaluate *IMM* with the largest social networks ever tested in the literature, and show that it is orders of magnitude faster than the state-of-the-art methods for several popular diffusion models. (Section 6)

2. PRELIMINARIES

This section formally defines the influence maximization problem, and presents an overview of *TIM* [33] to facilitate the discussions of our solution in subsequent sections.

2.1 Problem Definition

Let G be a social network with a set V of n nodes and a set E of m directed edges. For any two nodes $u, v \in V$, if $\langle u, v\rangle$ is an edge in E, we say that v is an outgoing neighbor of u, and u is an incoming neighbor of v. We consider a time-stamped diffusion process on G as follows:

1. At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
2. When a node u is first activated at a certain timestamp, it samples a set of its outgoing neighbors according to a certain probability distribution, and activates them at the next timestamp. After that, u cannot activate any other nodes.
3. Once a node becomes activated, it remains activated in all subsequent timestamps.

Let $I(S)$ be the number of activated nodes in G when the above process converges, i.e., when no more nodes can be activated. We refer to S as the *seed set*, and $I(S)$ as the *influence* of S. Note that $I(S)$ is a random variable that depends on the probability distributions from which each node samples out-neighbors to activate. We refer to the collection of those probability distributions as the *diffusion model* that underlies the diffusion process.

Problem Statement. Given G, a diffusion model M, and a positive integer k, the influence maximization problem asks for a size-k node set S_k with the maximum expected influence $\mathbb{E}\left[I(S_k)\right]$.

2.2 Diffusion Models

Unless specified otherwise, we focus on the *triggering model* [25] , which is a classic and general diffusion model adopted by most existing work on influence maximization. To explain the triggering model, we first introduce one of its special cases referred to as the *independent cascade (IC)* [25] model, for ease of exposition. The IC model originates from the marketing literature [22,23], and it assumes that each edge $e \in E$ is associated with a probability $p(e) \in [0,1]$. For any node u and any of its outgoing neighbors v, if u is first activated at timestamp i, then it has $p(\langle u, v\rangle)$ probability to activate v at timestamp $i+1$. In other words, whether or not u can activate v is independent of the history of diffusion before u is activated, and hence, the order of node activations does not affect the diffusion results. For such a model, the diffusion process from a seed set S has an equivalent formulation as follows [25]:

1. We flip a coin for each edge e in G and remove it with $1 - p(e)$ probability. Let g be the resulting social network.
2. We activate the nodes in S, as well as any nodes in g that can be reached[2] from S.

Kempe et al. [25] show that the above formulation is essential in establishing the hardness and constant approximations for influence maximization under the IC model.

The triggering model is a generalization of the IC model, and it has a formulation similar to the one mentioned above. Specifically, it assumes that each node v in G is associated with a distribution $\mathcal{T}(v)$ over subsets of v's incoming neighbors. We refer to $\mathcal{T}(v)$ as the *triggering distribution* of v, and refer to a sample from $\mathcal{T}(v)$ as a *triggering set* of v. For any seed set S, a diffusion process from S under the triggering model is formulated as follows:

1. For each node v, we sample a triggering set of v from $\mathcal{T}(v)$, and remove any incoming edge of v that starts from a node not in the triggering set. Let g be the resulting social network.
2. We activate the nodes in S, as well as any nodes in g that can be reached from S.

[1] **I**nfluence **M**aximization via **M**artingales

[2] We say that a node v in g can be reached from S, if there exists a directed path in g that starts from a node in S and ends at v.

For convenience, we use $\mathcal{G}$ to denote the distribution of g induced by the randomness in sampling from each node's triggering distribution.

As with many existing solutions under the triggering model (e.g., [4–6, 8, 15, 18–20, 25, 26, 29, 32, 33]), our influence maximization algorithms require taking samples from various nodes' triggering distributions, as a means to evaluate the expected influence of various node sets. The cost of this sampling process varies with the specific type of triggering model considered. For example, under the IC model, sampling from the triggering distribution $\mathcal{T}(v)$ of a node v requires flipping a coin for each of v's incoming edges, which takes time linear to the number x of incoming edges that v has. As another example, under the *linear threshold (LT)* model [25] (which is another special case of the triggering model), a sample from $\mathcal{T}(v)$ has p_i ($i \in [1, x]$) probability to be a unit set containing the i-th incoming neighbor of v, and has $1 - \sum_{i=1}^{x} p_i$ probability to be an empty set. Such a sample can be derived in $O(x)$ time as follows. We first generate a random number μ from a uniform distribution on $[0, 1]$. If $\mu > \sum_{i=1}^{x} p_i$, we return $\emptyset$; otherwise, we identify the smallest j ($j \in [1, x]$) such that $\sum_{i=1}^{j} p_j > \mu$, and return the j-th incoming neighbor of v. Note that the complexity of this sampling process can even be reduced to $O(1)$, if we allow $O(x)$-time preprocessing on $\mathcal{T}(v)$ [12, 35].

In general, with some pre-computation, the sampling methods in [12, 35] allow us to sample a random number from an arbitrary discrete distribution in $O(1)$ time. Given such methods, we can sample from an arbitrary triggering distribution $\mathcal{T}(v)$ in $O(x)$ time, by first spending $O(1)$ time to choose a triggering set based on $\mathcal{T}(v)$, and then returning the chosen triggering set in $O(x)$ time. (Recall that each triggering set contains at most x elements.)

In the remainder of the paper, we assume that the input graph G is properly preprocessed so that sampling from $\mathcal{T}(v)$ requires $O(x)$ time for any $v \in G$. Note that this assumption does not unfairly favor our algorithms, since all influence maximization methods considered in this paper (e.g., *TIM* and *TIM*$^+$) rely on efficient methods to sample from triggering distributions. In addition, the same assumption is implicitly adopted in previous work [33]. Furthermore, for the IC and LT models (i.e., the two diffusion models considered in most existing work), the assumption holds even without preprocessing G, as we previously analyzed.

2.3 Revisiting *TIM* and *TIM*$^+$

As mentioned in Section 1, both *TIM* and *TIM*$^+$ [33] have a parameter estimation phase and a node selection phase, the latter of which samples numerous node sets from the social network G to derive a solution for influence maximization. The type of node sets sampled by *TIM* and *TIM*$^+$ is referred to as *random RR sets*, defined as follows:

DEFINITION 1 (REVERSE REACHABLE SET). *Let v be a node in V. A* reverse reachable (RR) set *for v is generated by first sampling a graph g from $\mathcal{G}$, and then taking the set of nodes in g that can reach v. A* random RR set *is an RR set for a node selected uniformly at random from V.*

Intuitively, if a node u appears in an RR set of another node v, then a diffusion process from a seed set containing u should have a certain probability to activate v. This connection between RR sets and node activations is formalized in the following lemma.

LEMMA 1 ([5]). *For any seed set S and any node v, the probability that a diffusion process from S can activate v equals the probability that S overlaps an RR set for v.* □

Based on Lemma 1, the node selection phase of *TIM* (resp. *TIM*$^+$) adopts the following approach for influence maximization:

1. Generate a sizable set $\mathcal{R}$ of *independent* random RR sets.
2. Consider the *maximum coverage* problem [34] of selecting k nodes to *cover*[3] the maximum number of RR sets in $\mathcal{R}$. Apply the standard greedy algorithm to obtain a $(1 - 1/e)$-approximate solution S_k^* to the problem.
3. Return S_k^* as the answer for influence maximization.

To understand why the above approach works, observe that for any seed set S, the fraction of RR sets in $\mathcal{R}$ covered by S is an unbiased estimator of $\mathbb{E}[I(S)]$ (this is due to Lemma 1). Therefore, a seed set S_k^* that covers a large number of RR sets in $\mathcal{R}$ is likely to have a large expected influence, which makes S_k^* a good solution to the influence maximization problem.

To ensure that S_k^* provides a $(1 - 1/e - \varepsilon)$-approximation with at least $1 - 1/n^\ell$ probability, it is shown [33] that the number of RR sets in $\mathcal{R}$ should be at least λ/OPT, where OPT is the maximum expected influence of any size-k node set in G, and λ is a function of k, ℓ, n, and ε. Since OPT is unknown in advance, *TIM* sets $|\mathcal{R}| = \lambda/KPT$, where KPT is a lower bound of OPT that is derived by the parameter estimation phase of *TIM*. As mentioned in Section 1, however, KPT can be considerably smaller than OPT (which leads to an excessive number of RR sets in $\mathcal{R}$), and its derivation often incurs significant computational costs. To alleviate this issue, *TIM*$^+$ incorporates a heuristic approach to refine KPT. In particular, it selects a certain size-k node set S_k, and then estimates S_k's expected influence $\mathbb{E}[I(S_k)]$. After that, *TIM*$^+$ uses the larger one between KPT and $\mathbb{E}[I(S_k)]$ as an improved lower bound of OPT. Although this heuristic approach tends to yield a tighter lower bound of OPT, it does not reduce the significant overhead in deriving KPT. Furthermore, in the worst case when $KPT \geq \mathbb{E}[I(S_k)]$, the heuristic approach would fail to offer any improvement over *TIM*.

3. PROPOSED SOLUTION

This section presents *IMM*, an influence maximization technique that borrows ideas from *TIM* and *TIM*$^+$ [33] but adopts a novel algorithm design to reduce computational costs. At a high level, *IMM* consists of two phases as follows:

1. **Sampling.** This phase iteratively generates random RR sets and puts them into a set $\mathcal{R}$, until a certain stopping condition is met.
2. **Node Selection.** This phase applies the standard greedy algorithm for maximum coverage [34] to derive a size-k node set S_k^* that covers a large number of RR sets in $\mathcal{R}$. It then returns S_k^* as the final result.

The node selection phase of *IMM* is similar to that of *TIM* and *TIM*$^+$; however, *TIM* and *TIM*$^+$ requires all random RR sets in $\mathcal{R}$ to be independent, which is not the case in *IMM*. In particular, the random RR sets generated by *IMM* are dependent because, in the sampling phase of *IMM*, whether the $(i+1)$-th RR set is generated relies on whether the first i RR sets satisfy the stopping condition. Allowing such dependencies among RR sets makes *IMM*'s theoretical guarantees more difficult to analyze, but it is crucial for *IMM*'s efficiency as it enables *IMM* to eliminate a considerable amount of redundant computation that is unavoidable in *TIM* and *TIM*$^+$ (see Section 3.4). Meanwhile, the sampling phase of *IMM* considerably differs from the parameter estimation phase of *TIM* and *TIM*$^+$, in that it employs a more advanced statistical method to decide the number θ of RR sets required. This method not only provides more

[3] We say that a node set S covers an RR set R if $S \cap R \neq \emptyset$.

Notation	Description
$G = (V, E)$	a social network G with a node set V and an edge set E
n, m	the numbers of nodes and edges in G, respectively
k	the size of the seed set for influence maximization
$\mathcal{T}(v)$	the triggering distribution of a node $v \in V$
$\mathcal{G}$	the distribution of social networks induced by the triggering distribution of each node (see Section 2.1)
$I(S)$	the influence of a node set S in a diffusion process on G
$\mathbb{E}(\cdot)$	the expectation of a random variable
$\mathcal{R}$	the set of RR sets generated by *IMM*'s sampling phase
θ	the number of RR sets in $\mathcal{R}$
$F_{\mathcal{R}}(S)$	the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S
OPT	the maximum expected influence of any size-k node set S

Table 1: Frequently used notations.

accurate choice of θ, but also leads to a significantly smaller cost of parameter estimation than in the case of *TIM* and *TIM*$^+$.

In the following, we first introduce the concept of *martingales* [10], which is crucial in our analysis of the dependent RR sets in $\mathcal{R}$. After that, we discuss the condition under which the node selection phase of *IMM* returns a good solution, and then we clarify the details of the sampling phase. For ease of reference, Table 1 lists the notations that we frequently use. Unless specified otherwise, all logarithms in this paper are to the base e.

3.1 A Martingale View of RR Sets

Let $R_1, R_2, \ldots, R_\theta$ be the sequence of random RR sets generated in the sampling phase of *IMM*. Let S be any set of nodes in G, and x_i ($i \in [1, \theta]$) be a random variable that equals 0 if $S \cap R_i = \emptyset$, and 1 otherwise. By Lemma 1 and the linearity of expectation,

$$\mathbb{E}[I(S)] = \frac{n}{\theta} \cdot \mathbb{E}\left[\sum_{i=1}^{\theta} x_i\right]. \qquad (1)$$

Accordingly, the node selection phase of *IMM* uses $\frac{n}{\theta} \cdot \sum_{i=1}^{\theta} x_i$ as an estimator of $\mathbb{E}[I(S)]$. To ensure that this estimation is accurate, it is essential that $\sum_{i=1}^{\theta} x_i$ should not deviate significantly from its expectation. A classic tool to prove such concentration results is the Chernoff bounds [10], but it requires all x_i to be independent, which contradicts the dependencies among x_i caused by the stopping condition in *IMM*'s sampling phase. We circumvent this issue with an analysis based on *martingales* [10], defined as follows:

DEFINITION 2 (MARTINGALE). *A sequence of random variables $Y_1, Y_2, Y_3, \ldots$ is a* martingale, *if and only if $\mathbb{E}[|Y_i|] < +\infty$ and $\mathbb{E}[Y_i \mid Y_1, Y_2, \ldots, Y_{i-1}] = Y_{i-1}$ for any i.* □

To establish the connections between x_i ($i \in [1, \theta]$) and martingales, we first introduce two properties of *IMM*'s sampling phase. First, each R_i ($i \in [1, \theta]$) is generated for a node v selected uniformly at random, and it contains nodes that can reach v on a graph g sampled from $\mathcal{G}$. Second, the choices of v and g are independent of $R_1, R_2, \ldots, R_{i-1}$ (although the decision to generate R_i depends on $R_1, R_2, \ldots, R_{i-1}$). Due to these properties, for any $i \in [1, \theta]$,

$$\mathbb{E}[x_i \mid x_1, x_2, \ldots, x_{i-1}] \;=\; \mathbb{E}[x_i] \;=\; \mathbb{E}[I(S)]/n.$$

Let $p = \mathbb{E}[I(S)]/n$ and $M_i = \sum_{j=1}^{i}(x_j - p)$. Then, we have $\mathbb{E}[M_i] = 0$, and

$$\mathbb{E}[M_i \mid M_1, M_2, \ldots, M_{i-1}] = M_{i-1}.$$

Algorithm 1: NodeSelection ($\mathcal{R}$, k)

```
Initialize a node set $S_k^* = \emptyset$.;
for $i = 1$ to $k$ do
    Identify the vertex $v$ that maximizes $F_{\mathcal{R}}(S_k^* \cup v) - F_{\mathcal{R}}(S_k^*)$;
    Insert $v$ into $S_k^*$;
return $S_k^*$;
```

Therefore, by Definition 2, $M_1, M_2, \ldots, M_\theta$ is a martingale.

The following lemma from [10] shows two concentration results for martingales that have similar flavor to the Chernoff bounds.

LEMMA 2 ([10]). *Let $Y_1, Y_2, Y_3, \ldots$ be a martingale, such that $|Y_1| \le a$, $|Y_j - Y_{j-1}| \le a$ for any $j \in [1, i]$, and*

$$\mathrm{V}ar[Y_1] + \sum_{j=2}^{i} \mathrm{V}ar[Y_j \mid Y_1, Y_2, \ldots, Y_{j-1}] \le b,$$

where $\mathrm{V}ar[\cdot]$ denotes the variance of a random variable. Then, for any $\lambda > 0$,

$$\Pr\Big[Y_i - \mathbb{E}[Y_i] \ge \gamma\Big] \quad\le\quad \exp\left(-\frac{\gamma^2}{\frac{2}{3}a\gamma + 2b}\right). \qquad \square$$

Consider the martingale $M_1, M_2, \ldots, M_\theta$. We have $|M_1| \le 1$ and $|M_j - M_{j-1}| \le 1$ for any $j \in [2, \theta]$. By the properties of R_i ($i \in [1, \theta]$), we also have

$$\begin{aligned} &\mathrm{V}ar[M_1] + \textstyle\sum_{j=2}^{\theta}\mathrm{V}ar[M_j \mid M_1, M_2, \ldots, M_{j-1}] \\ &\qquad = \textstyle\sum_{j=1}^{\theta}\mathrm{V}ar[M_j] \;=\; \theta p \cdot (1 - p). \end{aligned}$$

Then, by Lemma 2 and $M_\theta = \sum_{i=1}^{\theta}(x_i - p)$, we have the following corollary.

COROLLARY 1. *For any $\varepsilon > 0$,*

$$\Pr\left[\sum_{i=1}^{\theta} x_i - \theta p \ge \varepsilon \cdot \theta p\right] \quad\le\quad \exp\left(-\frac{\varepsilon^2}{2 + \frac{2}{3}\varepsilon} \cdot \theta p\right). \quad \square$$

In addition, by applying Lemma 2 on the martingale $-M_1, -M_2, \ldots, -M_\theta$, we have the following result.

COROLLARY 2. *For any $\varepsilon > 0$,*

$$\Pr\left[\sum_{i=1}^{\theta} x_i - \theta p \le -\varepsilon \cdot \theta p\right] \quad\le\quad \exp\left(-\frac{\varepsilon^2}{2} \cdot \theta p\right). \qquad \square$$

In Section 3.2, we will utilize the above corollaries to establish the approximation guarantees of *IMM*.

3.2 Node Selection Phase

Let $\mathcal{R} = \{R_1, R_2, \ldots, R_\theta\}$ be the set of random RR sets generated by the sampling phase of *IMM*, and $F_{\mathcal{R}}(S)$ be the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S. Algorithm 1 shows the pseudo-code of the node selection phase of *IMM*, which takes as input $\mathcal{R}$ and a positive integer k, and returns a size-k node set S_k^*. The algorithm corresponds to the standard greedy approach for the maximum coverage problem [34], which guarantees that $F_{\mathcal{R}}(S_k^*)$ is at least $(1 - 1/e)$ times the fraction of RR sets covered by any size-k node set. In what follows, we show that when the number θ of RR sets in $\mathcal{R}$ is sufficiently large, S_k^* ensures a $(1 - 1/e - \varepsilon)$-approximation to the influence maximization problem with a high probability.

Consider the size-k node set S_k° with the maximum expected influence. Let $OPT = \mathbb{E}[I(S_k^\circ)]$. By Equation 1, $n \cdot F_{\mathcal{R}}(S_k^\circ)$ is an unbiased estimator of OPT. Then, by Corollary 2, $F_{\mathcal{R}}(S_k^\circ)$ should be close to OPT when θ is sizable, as shown in the following lemma.

LEMMA 3. *Let $\delta_1 \in (0,1)$, $\varepsilon_1 > 0$, and*

$$\theta_1 = \frac{2n \cdot \log(1/\delta_1)}{OPT \cdot \varepsilon_1^2}. \tag{2}$$

If $\theta \geq \theta_1$, then $n \cdot F_{\mathcal{R}}(S_k^\circ) \geq (1-\varepsilon_1) \cdot OPT$ holds with at least $1-\delta_1$ probability. □

Suppose that $n \cdot F_{\mathcal{R}}(S_k^\circ) \geq (1-\varepsilon_1) \cdot OPT$ holds. By the properties of the greedy approach,

$$\begin{aligned} n \cdot F_{\mathcal{R}}(S_k^*) &\geq (1-1/e) \cdot n \cdot F_{\mathcal{R}}(S_k^\circ) \\ &\geq (1-1/e) \cdot (1-\varepsilon_1) \cdot OPT. \end{aligned} \tag{3}$$

Intuitively, Equation 3 indicates that the expected influence of S_k^* is likely to be large, since $n \cdot F_{\mathcal{R}}(S_k^*)$ is an indicator of $\mathbb{E}[I(S_k^*)]$. We formalize this intuition in the following lemma:

LEMMA 4. *Let $\delta_2 \in (0,1)$, $\varepsilon_1 < \varepsilon$, and*

$$\theta_2 = \frac{(2-2/e) \cdot n \cdot \log\left(\binom{n}{k}/\delta_2\right)}{OPT \cdot (\varepsilon - (1-1/e) \cdot \varepsilon_1)^2}. \tag{4}$$

If Equation 3 holds and $\theta \geq \theta_2$, then with at least $1-\delta_2$ probability, $\mathbb{E}[I(S_k^*)] \geq (1-1/e-\varepsilon) \cdot OPT$. □

Lemmas 3 and 4 lead to the following theorem.

THEOREM 1. *Given any $\varepsilon_1 \leq \varepsilon$ and any $\delta_1, \delta_2 \in (0,1)$ with $\delta_1 + \delta_2 \leq 1/n^\ell$, setting $\theta \geq \max\{\theta_1, \theta_2\}$ ensures that the node selection phase of IMM returns a $(1-1/e-\varepsilon)$-approximate solution with at least $1-1/n^\ell$ probability.* □

Parametrization. Assume that OPT is known. Then, by Theorem 1, we can obtain a $(1-1/e-\varepsilon)$-approximation from *IMM* with at least $1-1/n^\ell$ probability as follows: We first choose ε_1, δ_1, and δ_2 according to the conditions stated in Theorem 1; after that, we calculate θ_1 and θ_2 based on Equations 2 and 4, respectively, and compute $\theta = \max\{\theta_1, \theta_2\}$; finally, we ensure that $\mathcal{R}$ contains at least θ RR sets. A natural question is, how should we select ε_1, δ_1, and δ_2 to minimize θ? Getting an exact answer to this question is difficult, due to the complex relationships among ε_1, δ_1, δ_2, and θ. However, there exists a relatively simple method to derive a near-minimum θ, as explained in the following.

First, we set $\delta_1 = \delta_2 = 1/(2n^\ell)$. Under this setting, θ is minimized when $\theta_1 = \theta_2$. In turn, $\theta_1 = \theta_2$ holds if and only if $\varepsilon_1 = \varepsilon \cdot \frac{\alpha}{(1-1/e)\cdot\alpha+\beta}$, where

$$\begin{aligned} \alpha &= \sqrt{\ell \log n + \log 2}, \text{ and} \\ \beta &= \sqrt{(1-1/e) \cdot \left(\log\binom{n}{k} + \ell \log n + \log 2\right)}. \end{aligned} \tag{5}$$

In that case,

$$\theta = \frac{2n \cdot ((1-1/e) \cdot \alpha + \beta)^2}{OPT \cdot \varepsilon^2}.$$

Denote the above θ as θ^*. The following lemma shows that θ^* is near-optimal with respect to Theorem 1:

LEMMA 5. *Given ε, ℓ, let θ° be the smallest possible θ under the conditions stated in Theorem 1. We have*

$$\theta^\circ \leq \theta^* \leq \theta^\circ \cdot \frac{\ell \log n + \log 2}{\ell \log n}. \qquad \square$$

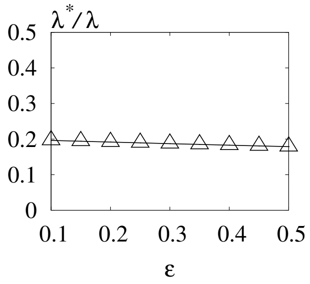


Figure 1: λ^*/λ **vs.** ε.

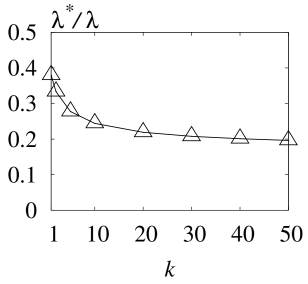


Figure 2: λ^*/λ **vs.** k.

In summary, the node selection phase of *IMM* returns a $(1-1/e-\varepsilon)$-approximation with at least $1-1/n^\ell$ probability, if we ensure that the number θ of RR sets in $\mathcal{R}$ is at least λ^*/OPT, where

$$\lambda^* = 2n \cdot \left((1-1/e) \cdot \alpha + \beta\right)^2 \cdot \varepsilon^{-2}. \tag{6}$$

Nevertheless, ensuring $\theta \geq \lambda^*/OPT$ is non-trivial, since OPT is unknown in advance. In Section 3.3, we address this issue by setting $\theta = \lambda^*/LB$, where LB is a lower bound of OPT that is almost tight.

Finally, we have the following result regarding the time complexity of Algorithm 1.

COROLLARY 3. *Algorithm 1 runs in $O\left(\sum_{R \in \mathcal{R}} |R|\right)$ time.* □

The above corollary follows from the fact that Algorithm 1 corresponds to the standard greedy approach for maximum coverage, which runs in time linear to the total size of its input [34].

Comparison with the Node Selection Phase of *TIM* and *TIM*$^+$. Previous work [33] utilizes Algorithm 1 as part of the node selection phase of *TIM* and *TIM*$^+$, and establishes two conditions for Algorithm 1 to return a $(1-1/e-\varepsilon)$-approximation with at least $1-1/n^\ell$ probability. First, all random RR sets in $\mathcal{R}$ should be independent. Second, $\theta \geq \lambda/OPT$, where

$$\lambda = (8+2\varepsilon)n \cdot \left(\ell \log n + \log\binom{n}{k} + \log 2\right) \cdot \varepsilon^{-2}. \tag{7}$$

In comparison, our results are more general in that we allow dependencies among the RR sets in $\mathcal{R}$. Furthermore, our analysis leads to a θ that is near-optimal with respect to Theorem 1, whereas the θ given in [33] roughly corresponds to a suboptimal case in Theorem 1 with $\delta_1 = 1/\left(n^\ell \cdot \binom{n}{k}\right)$, $\delta_2 = 1-\delta_1$, and $\varepsilon_1 = \varepsilon/2$.

To illustrate the differences between our results and those in [33], Figure 1 compares λ^* (in Equation 6) and λ (in Equation 7) under a realistic setting where $n = 10^6$, $\ell = 1$, $k = 50$, and ε varies from 0.1 to 0.5. Observe that $\lambda^* \leq 0.2\lambda$ in all cases. Figure 2 shows λ^*/λ as a function of k, with $\varepsilon = 0.1$, $n = 10^6$, and $\ell = 1$. Again, λ^* is considerably smaller than λ, regardless of the value of k.

3.3 Sampling Phase

Recall that *IMM*'s node selection phase requires as input a set $\mathcal{R}$ of random RR sets with $|\mathcal{R}| \geq \lambda^*/OPT$, where λ^* is as defined in Equation 6. As OPT is not known a priori, we aim to identify a lower bound LB of OPT and then set $|\mathcal{R}| = \lambda^*/LB \geq \lambda^*/OPT$. Ideally, LB should be as close to OPT as possible, so as to minimize the computational cost of generating $\mathcal{R}$. However, it is challenging to derive such an LB, given the NP-hardness of the influence maximization problem.

To address the above challenge, our idea is to design a statistical test $B(x)$, such that if $OPT < x$, then $B(x) = false$ with a high probability. With such a test, we can easily identify a lower bound of OPT by running $B(x)$ on $O(\log n)$ values of x, e.g., $x =$

Algorithm 2: Sampling (G, k, ε, ℓ)

```
1  Initialize a set $\mathcal{R} = \emptyset$ and an integer $LB = 1$;
2  Let $\varepsilon' = \sqrt{2} \cdot \varepsilon$;
3  for $i = 1$ to $\log_2 n - 1$ do
4      Let $x = n/2^i$;
5      Let $\theta_i = \lambda'/x$, where $\lambda'$ is as defined in Equation 9;
6      while $|\mathcal{R}| \le \theta_i$ do
7          Select a node $v$ from $G$ uniformly at random;
8          Generate an RR set for $v$, and insert it into $\mathcal{R}$;
9      Let $S_i$ = NodeSelection( $\mathcal{R}$ );
10     if $n \cdot F_{\mathcal{R}}(S_i) \ge (1 + \varepsilon') \cdot x$ then
11         $LB = n \cdot F_{\mathcal{R}}(S_i)/(1 + \varepsilon')$;
12         break;
13 Let $\theta = \lambda^*/LB$, where $\lambda^*$ is as defined in Equation 6;
14 while $|\mathcal{R}| \le \theta$ do
15     Select a node $v$ from $G$ uniformly at random;
16     Generate an RR set for $v$, and insert it into $\mathcal{R}$;
17 return $\mathcal{R}$
```

$n/2, n/4, n/8, \dots, 2$. (Note that $OPT \in [1, n]$.) Interestingly, the algorithm used in *IMM*'s node selection phase provides a means to implement $B(x)$, as shown in the following lemma:

LEMMA 6. *Let $x \in [1, n]$ and $\varepsilon', \delta_3 \in (0, 1)$. Suppose that we invoke Algorithm 1 on a set $\mathcal{R}$ containing θ random RR sets, with*

$$\theta \ge \frac{(2 + \frac{2}{3}\varepsilon') \cdot \left(\log \binom{n}{k} + \log(1/\delta_3)\right)}{\varepsilon'^2} \cdot \frac{n}{x}. \tag{8}$$

Let S_k^* be the output of Algorithm 1. Then, if $OPT < x$, we have $n \cdot F_{\mathcal{R}}(S_k^*) < (1 + \varepsilon') \cdot x$ with at least $1 - \delta_3$ probability. □

The rationale of Lemma 6 is as follows. First, if OPT is large, then Algorithm 1 is likely to return a node set S_k^* that covers a large portion of RR sets in $\mathcal{R}$, i.e., $F_{\mathcal{R}}(S_k^*)$ should be large. Conversely, if we observe that $F_{\mathcal{R}}(S_k^*)$ is small, then OPT is also likely to be small (modulo the sampling error in $\mathcal{R}$). Therefore, we can use $F_{\mathcal{R}}(S_k^*)$ as an indicator of the magnitude of OPT.

Based on Lemma 6, Algorithm 2 presents the pseudo-code of the sampling phase of *IMM*. Given G, k, ε, and ℓ, the algorithm first initializes a set $\mathcal{R} = \emptyset$, and an integer $LB = 1$ (Line 1). After that, it sets a parameter $\varepsilon' = \sqrt{2} \cdot \varepsilon$, and enters a for loop with at most $\log_2 n - 1$ iterations (Lines 3 - 12). (We will explain our choice of ε' shortly.)

In the i-th iteration, the algorithm computes $x = n/2^i$, and then derives $\theta_i = \lambda'/x$, where

$$\lambda' = \frac{(2 + \frac{2}{3}\varepsilon') \cdot \left(\log \binom{n}{k} + \ell \cdot \log n + \log \log_2 n\right) \cdot n}{\varepsilon'^2}. \tag{9}$$

It can be verified that θ_i equals the smallest θ that satisfies Equation 8 when $\delta_3 = n^\ell \cdot \log_2 n$. As a next step, the algorithm inserts random RR sets into $\mathcal{R}$ until $|\mathcal{R}| = \theta_i$ (Lines 6-8), after which it invokes Algorithm 1 on $\mathcal{R}$ to obtain a size-k node set S_i. By Lemma 6, if

$$n \cdot F_{\mathcal{R}}(S_i) \ge (1 + \varepsilon') \cdot x, \tag{10}$$

then $OPT \ge x$ should hold with at least $1 - 1/(n^\ell \cdot \log_2 n)$ probability. In other words, once Equation 10 holds, we can terminate the for loop and identify x as a lower bound of OPT. This explains Line 10-12 in Algorithm 2, except that the algorithm uses $LB = n \cdot F_{\mathcal{R}}(S_i)/(1 + \varepsilon')$ as a lower bound of OPT, instead of setting $LB = x$. (Notice that the former is a tighter lower bound, due to Equation 10.) Our choice of LB is justified by the following lemma.

LEMMA 7. *Let x, ε', δ_3, $\mathcal{R}$, and S_k^* be as defined in Lemma 6. If $OPT \ge x$, then we have $OPT \ge n \cdot F_{\mathcal{R}}(S_k^*)/(1 + \varepsilon')$ with at least $1 - \delta_3$ probability.* □

The above discussions assume that Equation 10 holds in a ceratin iteration of the for loop in Algorithm 2. On the other hand, if Equation 10 does not hold in any of the $\log_2 n - 1$ iterations, then the algorithm sets $LB = 1$. In either case, once LB is determined, the algorithm computes $\theta = \lambda^*/LB$, where λ^* is as defined in Equation 6, and then it adds more random RR sets into $\mathcal{R}$ until $|\mathcal{R}| = \theta$ (Lines 13-16). Finally, it terminates by returning $\mathcal{R}$.

Theoretical Analysis. Let $j = \left\lceil \log_2 \frac{n}{OPT} \right\rceil$. By Lemma 6 and the union bound, Algorithm 2 has at most $(j - 1)/(n^\ell \cdot \log_2 n)$ probability to terminate its for loop before the j-th iteration. Then, by Lemma 7 and the union bound, Algorithm 2 sets $LB \le OPT$ with at least $1 - n^\ell$ probability. Therefore, we have the following result.

THEOREM 2. *With at least $1 - n^\ell$ probability, Algorithm 2 returns a set $\mathcal{R}$ of RR sets with $|\mathcal{R}| \ge \lambda^*/OPT$, where λ^* is as defined in Equation 6.* □

In addition, we can show that LB is close to OPT with a high probability. Towards this end, we prove that Algorithm 2 is likely to stop its for loop soon after the j-th iteration, as shown in the following lemma.

LEMMA 8. *Let x, ε', δ_3, $\mathcal{R}$, and S_k^* be as defined in Lemma 6. Let $c \ge 1$. If $OPT \ge c \cdot \frac{(1+\varepsilon')^2}{1-1/e} \cdot x$, then $n \cdot F_{\mathcal{R}}(S_k^*) < (1 + \varepsilon') \cdot x$ holds with at most $(\delta_3)^c$ probability.* □

Based on Lemma 8, we have the following result regarding LB.

LEMMA 9. *Let $\mathcal{R}$ be the output of Algorithm 2, and LB be as in Line 13 of Algorithm 2. We have $LB \ge \frac{1-1/e}{(1+\varepsilon')^2} \cdot OPT$ with at least $1 - 1/n^\ell$ probability. In addition, when $1/n^\ell \le 1/2$,*

$$\mathbb{E}[|\mathcal{R}|] \le \frac{3 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT},$$

where λ' is as defined in Equation 9. □

In Section 3.4, we will utilize Lemma 9 to analyze the total running time of *IMM* and compare it with that of *TIM* and *TIM*$^+$ [33].

Parametrization. It remains to explain why we set $\varepsilon' = \sqrt{2} \cdot \varepsilon$ in Algorithm 2. As shown in Lemma 9, the expected number of RR sets generated by Algorithm 2 is bounded by a quantity proportional to $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$. Ideally, we should choose an ε' that minimizes $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$, but it is difficult given the complex expressions of λ^* and λ'. As an alternative, we roughly approximate $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$ with a simple function of ε' and ε, and then derive $\varepsilon' = \sqrt{2} \cdot \varepsilon$ as a minimizer of the function. Our experiments show that this choice of ε' leads to reasonably good performance of *IMM*.

3.4 Putting It Together

In summary, *IMM* first invokes Algorithm 2 to generate a set $\mathcal{R}$ of RR sets, and then feeds $\mathcal{R}$ to Algorithm 1 to obtain a size-k node set S_k^* as the final output. By Theorems 1 and 2 as well as the union bound, *IMM* returns a $(1 - 1/e - \varepsilon)$-approximate solution to influence maximization with at least $1 - 2/n^\ell$ probability. The success probability of *IMM* can be easily improved to $1 - 1/n^\ell$, by increasing ℓ by a factor of $1 + \log 2/\log n$. Algorithm 3 shows the pseudo-code of a version of *IMM* with the improved success probability.

Algorithm 3: IMM (G, k, ε, ℓ)

```
$\ell = \ell \cdot (1 + \log 2/\log n)$;
$\mathcal{R}$ = Sampling ( $G$, $k$, $\varepsilon$, $\ell$ );
$S_k^*$ = NodeSelection ( $\mathcal{R}$, $k$ );
return $S_k^*$
```

In what follows, we discuss the time complexity of *IMM*. We focus on the sampling phase of *IMM* (i.e., Algorithm 2), since the time complexity of the node estimation phase (i.e., Algorithm 1) has been established in Corollary 3. Observe that the major computational cost of Algorithm 2 is incurred by the construction of RR sets. On the other hand, the time required in generating a random RR set R is $O(w(R))$, where $w(R)$ denotes the number of edges in G that point to the nodes in R. Specifically, the generation of R requires (i) sampling an edge set from the triggering distribution of each node in R and (ii) traversing all edges in the edge sets sampled. The former requires $O(w(R))$ time, since a sample from a node v's triggering distribution incurs a linear cost to the number of incoming edges of v. Meanwhile, the latter takes $O(w(R))$ time, since the sampled edge sets contain $O(w(R))$ time in total.

Let EPT be the expected value of $w(R)$. We have the following result from [33].

LEMMA 10 ([33]). $n \cdot EPT \leq m \cdot OPT$. □

Now consider the set $\mathcal{R}$ of RR sets generated by Algorithm 2. By Lemma 9,

$$\begin{aligned}\mathbb{E}[|\mathcal{R}|] &\leq \frac{3\max\{\lambda^*, \lambda'\} \cdot (1+\varepsilon')^2}{(1-1/e)\cdot OPT} \\ &= O\left((k+\ell)n\log n \cdot \varepsilon^{-2}/OPT\right). \qquad (11)\end{aligned}$$

If $|\mathcal{R}|$ is independent of $w(R)$ for each $R \in \mathcal{R}$, then by Lemma 10 and Equation 11, the expected total time to generate $\mathcal{R}$ is

$$\begin{aligned}\mathbb{E}\Big[\textstyle\sum_{R\in\mathcal{R}} w(R)\Big] &= \mathbb{E}\big[|\mathcal{R}|\big]\cdot EPT \\ &= O\left((k+\ell)(n+m)\log n/\varepsilon^2\right). \quad (12)\end{aligned}$$

However, $|\mathcal{R}|$ is correlated with the RR sets in $\mathcal{R}$, because $|\mathcal{R}|$ depends on when Algorithm 2 terminates its for loop, which in turn is decided by the RR sets in $\mathcal{R}$. But interestingly, we can prove that Equation 12 still holds, by utilizing the *optional stopping theorem* [38] for martingales. For our discussion, we introduce a simplified version of the theorem[4]as follows.

THEOREM 3 ([38]). *Let $Y_1, Y_2, Y_3, \ldots$ be a martingale, and τ be a random variable such that the event $\tau = i$ is independent of any Y_j with $j > i$. If $Pr[\tau < +\infty] = 1$, then*

$$\mathbb{E}[Y_\tau] = \mathbb{E}[Y_1]. \qquad (13)$$

An intuitive way to understand Theorem 3 is as follows. Suppose that we retrieve $Y_1, Y_2, \ldots$ one by one, and we have a certain algorithm that examines the Y_i that we obtain and decides when we should stop the retrieval process. (That is, the algorithm determines the value of τ.) Then, as long as the algorithm always terminates in finite time, the expectation of the last retrieved Y_i is equal to $\mathbb{E}[Y_1]$.

Based on Theorem 3, we prove Equation 12 as follows. First, let R_i be the i-th RR set generated by Algorithm 2, and $Z_i = \sum_{j=1}^{i}\big(w(R_i) - EPT\big)$. It can be verified that $Z_1, Z_2, \ldots$ is a martingale. Let τ be a random variable that denotes the total number of RR sets generated by Algorithm 3. Observe that $\tau < +\infty$ always holds. Thus, by Theorem 3, $\mathbb{E}[Z_\tau] = \mathbb{E}[Z_1]$. Notice that

$$\begin{aligned}\mathbb{E}[Z_1] &= \mathbb{E}\big[w(R_1) - EPT\big] = 0, \text{ and} \\ \mathbb{E}[Z_\tau] &= \mathbb{E}\left[\textstyle\sum_{j=1}^{\tau}\big(w(R_i) - EPT\big)\right] \\ &= \mathbb{E}\left[\textstyle\sum_{j=1}^{\tau} w(R_i)\right] - \mathbb{E}[\tau]\cdot EPT \\ &= \mathbb{E}\left[\textstyle\sum_{R\in\mathcal{R}} w(R)\right] - \mathbb{E}\big[|\mathcal{R}|\big]\cdot EPT.\end{aligned}$$

Therefore, Equation 12 holds.

With the above discussions, we can prove that the expected running time of *IMM* equals the right hand side of Equation 12. This leads to the following theorem.

THEOREM 4. *Algorithm 3 returns a $(1-1/e-\varepsilon)$-approximate solution with at least $1 - 1/n^\ell$ probability, and runs in $O\left((k+\ell)(n+m)\log n/\varepsilon^2\right)$ expected time.* □

Comparison with *TIM* and *TIM*$^+$. *IMM*, *TIM*, and *TIM*$^+$ have the same approximation guarantee and expected time complexity, but *IMM* provides much higher efficiency in practice since it requires generating a much smaller number of RR sets. Specifically, the number of RR sets constructed in *IMM* is $O((k+\ell)n\log n \cdot \varepsilon^{-2}/OPT)$, whereas *TIM* requires $O((k+\ell)n\log n \cdot \varepsilon^{-2}/KPT)$ RR sets, where KPT is the expected influence of a node set obtained by sampling k nodes with replacement from G, with the sampling probability of each node being proportional to its indegree [33]. It can be verified that $KPT < OPT$ always holds, and $KPT/OPT = O(k/n)$ in the worst case. Therefore, *TIM* incurs a much larger overhead than *IMM*. Meanwhile, although *TIM*$^+$ improves upon *TIM* in terms of efficiency, its parameter estimation phase is identical to that of *TIM*, due to which it also requires producing a large number of RR sets to estimate KPT. As a consequence, *TIM*$^+$'s practical performance is much inferior to that of *IMM*, as we show in Section 6.

In addition, there exists another crucial difference between *IMM* and *TIM* (resp. *TIM*$^+$). In *IMM*, all RR sets generated in the sampling phase are re-used in the node selection phase. In contrast, *TIM*'s node selection phase is unable to recycle any RR sets produced in the parameter estimation phase, due to which it has to re-generate a large number of RR sets from scratch for influence maximization. The reason is that the node selection phase of *TIM* demands that all RR sets used should be independent, whereas the RR sets produced in *TIM*'s parameter estimation phase are correlated. This results in a considerable amount of redundant computation, which degrades *TIM*'s efficiency. *TIM*$^+$ suffers from the same issue since its node selection and parameter estimation phases are the same as those of *TIM*. Meanwhile, *IMM* circumvents this issue because its node selection phase accommodates dependencies among RR sets, which in turn is due to the martingale approach adopted in *IMM*.

4. EXTENSIONS

This section extends *IMM* beyond the triggering model, and demonstrates an application of *IMM* to the continuous-time model [16], a diffusion model not supported by *TIM* and *TIM*$^+$ [33].

4.1 Generalization of *IMM*

Our discussions in the previous sections have assumed the triggering model. However, observe that the algorithms used by *IMM* (i.e., Algorithms 1, 2, and 3) do not rely on anything specific to the triggering model, except that they require a method to generate

[4] In its original form, the optional stopping theorem states three sufficient conditions for Equation 13. For simplicity, we include only one condition in Theorem 3, but it suffices for our purpose.

random RR sets. To extend *IMM* to other diffusion models, we first generalize the definition of RR sets in a model-independent manner as follows.

DEFINITION 3 (REVERSE INFLUENCE SET). *Let v be a node in V. A* reverse influence (RI) set *for v is a node set $R \subseteq V$, such that for any node set $S \subseteq V$, the probability that $R \cap S \neq \emptyset$ equals the probability that S as a seed set can activate v in a diffusion process. A* random RI set *is an RI set for a node selected uniformly at random from V.* □

Notice that RR sets are a special case of RI sets under the triggering model, due to Lemma 1.

Let $\mathcal{R}$ be a set of RI sets under a certain diffusion model $\mathcal{M}$. Then, by Definition 3 and the linearity of expectation, we have

$$\mathbb{E}[I(S)] = n \cdot \mathbb{E}[F_{\mathcal{R}}(S)], \tag{14}$$

where $I(S)$ denotes the influence of S under $\mathcal{M}$, and $F_{\mathcal{R}}(S)$ denotes the fraction of RI sets in $\mathcal{R}$ that overlaps S. Given Equation 14 and the concentration results in Corollaries 1 and 2, it can be verified that all theoretical analysis in Section 3.2 holds under $\mathcal{M}$, if the input to Algorithm 1 is a set $\mathcal{R}$ of random RI sets generated under $\mathcal{M}$. It then follows that all results in Section 3.3 are still valid, even if we modify Algorithm 2 so that it generates random RI sets under $\mathcal{M}$ instead of random RR sets. As a consequence, Algorithm 3 is guaranteed to return a $(1 - 1/e - \varepsilon)$-approximation under $\mathcal{M}$, with at least $1 - 1/n^{\ell}$ probability.

On the other hand, the time complexity of *IMM* under $\mathcal{M}$ may differ from that stated in Theorem 4, since Theorem 4 assumes $n \cdot EPT \leq m \cdot OPT$ (see Lemma 10), which does not necessarily hold under $\mathcal{M}$. Nonetheless, if we re-define EPT as the expected cost to generate a random RI set under $\mathcal{M}$, then it follows from Theorem 3 that *IMM* has an expected time complexity of $O\left(\frac{EPT}{OPT} \cdot (k + \ell)(n + m)\log n/\varepsilon^2\right)$.

In summary, *IMM* can be extended to any diffusion model $\mathcal{M}$ for which random RI sets can be generated efficiently. In addition, it provides the following theoretical guarantees under $\mathcal{M}$.

THEOREM 5. *Under a diffusion model where a random RI set takes $O(EPT)$ expected time to generate, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$-approximation with at least $1 - 1/n^{\ell}$ probability, and runs in $O\left(\frac{EPT}{OPT} \cdot (k + \ell)(n + m)\log n/\varepsilon^2\right)$ expected time.* □

Remark. The above extension of *IMM* requires a procedure for generating random RI sets under the diffusion model $\mathcal{M}$. However, such a procedure may not exist. For example, consider a diffusion model $\mathcal{M}$ under which the expected influence of a node set S is not a submodular function of S. Then, it can be verified that it is impossible to generate RI sets under $\mathcal{M}$ that satisfy the condition stated in Definition 3. We leave as future work to characterize the diffusion models under which random RI sets can be produced.

4.2 Application to Continuous-Time Models

The *continuous-time independent cascade (CTIC) model* [6, 16, 30] assumes that each directed edge e in the social network G is associated with a *length distribution* $\mathcal{L}(e)$. Given a seed set S and a threshold t, a diffusion process under the CTIC model first samples a length $l(e)$ for each edge e, and then activates any node that can be reached from a node in S via a path with a length no more than t. The rationale behind this model is that $w(e)$ represents the time required to transmit information from the starting point of e to its ending point, and accordingly, the diffusion process from S with a threshold t captures the effect of information cascade within a certain time frame. Such a model is useful in practice since, in viral marketing, a marketer is often concerned about the effectiveness of a marketing campaign within a given time (e.g., one month from the start of the campaign) [15].

In what follows, we demonstrate that our generalization of *IMM* in Section 4.1 can be applied to handle the CTIC model. Towards that end, we first present an efficient algorithm to construct an RI set $R(v)$ for any given node v. Conceptually, the algorithm works in three steps. First, it decides the length of each edge e in G by taking a sample from $\mathcal{L}(e)$. Let g be the resulting weighted graph. Then, the algorithm invokes the Dijkstra's algorithm [13] on g to identify the shortest distance from each node to v. Finally, it returns the set $R(v)$ of all nodes whose distance to v is no more than t (i.e., the threshold parameter in the CTIC model).

For practical efficiency, we implement the above algorithm as follows. We invoke the Dijkstra's algorithm to traverse G with v as the starting point, following only the incoming edge of each node. Each time we encounter an edge e whose length has not been decided, we sample its length from $\mathcal{L}(e)$. By the properties of the Dijkstra's algorithm [13], it would maintain a priority queue Q during the traverse of G, and it removes a node u from Q only if the shortest path from u to v is determined. Accordingly, we monitor each node u' removed from Q by the algorithm, and we terminate the algorithm whenever the shortest path from u' to v is longer than t. After that, we return the set $R(v)$ of all nodes that have been removed from Q, except u'. We refer to this algorithm as *RI-Gen*.

It can be verified that the running time of *RI-Gen* is $O(n^* \log n^* + m^*)$, where n^* and m^* are the numbers of nodes and edges inspected by *RI-Gen* during the traversal of G. Although $n^* = n$ and $m^* = m$ in the worst case, we will show that when v is selected from G uniformly at random, the expected time complexity of *RI-Gen* can be much smaller than $O(n \log n + m)$. Before that, we first establish the correctness of *RI-Gen*.

LEMMA 11. *Suppose that we feed a node v as input to RI-Gen, and obtain a node set $R(v)$ as the output. Then, for any node set S, the probability that S overlaps $R(v)$ equals the probability that S as a seed set can activate v in a diffusion process under the CTIC model.* □

Based on Lemma 11, we have the following result related to *RI-Gen*'s running time.

LEMMA 12. *Let OPT^* be the largest expected influence of any size-1 node set in G under the CTIC model. Suppose that we choose a node v from G uniformly at random, and feed v as input to RI-Gen. Let m^* be the number of edges inspected by RI-Gen. Then,*

$$n \cdot \mathbb{E}[m^*] \leq m \cdot OPT^*. \qquad \square$$

Given Lemma 12, it can be derived that the expected time complexity of *RI-Gen* is $O\left(\frac{n \log n}{m} OPT\right)$. Meanwhile, Lemma 11 shows that we can extend *IMM* to tackle the CTIC model, by adopting *RI-Gen* to generate random RI sets. Then, by Theorem 5, we have the following result.

THEOREM 6. *Under the CTIC model, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$-approximation with at least $1 - 1/n^{\ell}$ probability, and runs in $O\left((k + \ell)(n + m)\log^2 n/\varepsilon^2\right)$ expected time.* □

Comparison with *ConTinEst* [15]. *ConTinEst* is the state-of-the-art solution for influence maximization under the CTIC model. It first generates c weighted versions of G, denoted as $g_1, g_2, \ldots, g_c$, such that in each g_i ($i \in [1, c]$), the length of each edge e is sampled from $\mathcal{L}(e)$ independently. After that, it constructs a *sketch* [11] on each g_i, and utilizes the sketches for influence maximization. Du et al. [15] provide an analysis of *ConTinEst*'s accuracy guarantee, but do not formally discuss its time complexity. In the following, we analyze the time complexity of *ConTinEst* based on the theoretical results from [15], and compare it with *IMM*.

First, the sketch on each g_i ($i \in [1, c]$) is parameterized with a positive integer d, and it takes $O(dn \log^2 n + dm \log n)$ time to construct [11]. Second, by Theorems 1 and 2 in [15], *ConTinEst* returns a $(1-1/e-\varepsilon)$-approximate solution with at least $1-1/n^\ell$ probability, if

$$c \;\geq\; \frac{\Lambda \cdot 4k^3(\ell+1)(\log n + \log 2)}{\varepsilon^2 \cdot OPT^2}, \text{ where} \qquad (15)$$

$$\Lambda = \max_{S \subseteq V, |S| \leq k} \left(\tfrac{2}{d-2}\mathbb{E}[I(S)]^2 + \tfrac{d-1}{d-2}\mathbb{V}ar[I(S)] + \tfrac{4\varepsilon \cdot r(S)}{3k} OPT \right),$$

and $r(S)$ denotes the maximum number of nodes that S can activate under the CTIC model. We observe that Λ is unknown in advance, and hence, it is unclear how a user can select an appropriate c to ensure a $(1-1/e-\varepsilon)$-approximation. However, this issue is not addressed in [15]. For our discussion, we use a loose lower bound of Λ to derive an optimistic estimation of *ConTinEst*'s time complexity, and then compare it with that of *IMM*. (Note that this is to the advantage of *ConTinEst*.)

Let S_k° be the size-k node set with the largest expected influence under the CTIC model. Then, $E[I(S_k^\circ)] = OPT$ and $r(S_k^\circ) \geq OPT$. Therefore,

$$\begin{aligned} \Lambda \;&\geq\; \left(\tfrac{2}{d-2}\mathbb{E}[I(S_k^\circ)]^2 + \tfrac{4\varepsilon \cdot r(S_k^\circ)}{3k} OPT \right) \\ &\geq\; \left(\tfrac{2}{d-2} + \tfrac{4\varepsilon}{3k} \right) OPT^2 \end{aligned}$$

By setting $\Lambda = \left(\frac{2}{d-2} + \frac{4\varepsilon}{3k} \right) OPT^2$ in Equation 15, we have

$$c \;\geq\; \left(\tfrac{8}{d-2} + \tfrac{16\varepsilon}{3k} \right) \cdot k^3(\ell+1)(\log n + \log 2)/\varepsilon^2. \qquad (16)$$

Given that *ConTinEst* needs to construct a sketch on each g_i ($i \in [1, c]$), and that each sketch takes $O(dn \log^2 n + dm \log n)$ time to generate, the time complexity of *ConTinEst* is at least

$$O\left(k^3(\ell+1)(n \log n + m)\log^2 n/\varepsilon^2\right).$$

Observe that this time complexity is larger than that of *IMM* (see Theorem 6) by a factor of more than k^2. In Section 6, we will demonstrate that the empirical efficiency of *ConTinEst* is also inferior to that of *IMM*.

5. RELATED WORK

Kempe et al. [25] presents the first algorithmic study on influence maximization. They show that the problem is NP-hard in general, but can be approximated with a factor of $1-1/e-\varepsilon$ under the triggering model using a polynomial-time greedy algorithm. The key of Kempe et al.'s algorithm is a monte-carlo approach for estimating the expected influence of any node set S. In particular, it simulates the diffusion process from S for a large number r of times, and then takes the average influence of S in the r simulations as an estimation of $\mathbb{E}[I(S)]$. With this monte-carlo approach, Kempe et al.'s algorithm greedily selects the node in G that leads to the largest increment of estimated influence (with respect to the nodes that are already selected), until k nodes are chosen.

Kempe et al.'s algorithm is simple and effective, but it incurs a high time complexity of $O(knmr)$, which restricts its application to very small graphs. This motivates numerous techniques [4–9, 15–19, 24–27, 29, 32, 33, 37] that improve upon Kempe et al.'s algorithm in terms of efficiency. Among them, the methods in [8, 20, 29] follow Kempe et al.'s greedy framework and retain its approximation ratio under the triggering model, while providing higher empirical efficiency by using branch-and-bound approaches to omit node sets whose expected influence are small. Those techniques, however, still entail $O(kmnr)$ computation time, due to which cannot scale to large graphs with millions of nodes. In contrast, techniques in [7–9, 19, 24, 27, 37] offer much better scalability by exploiting heuristics in node selection, but none of them retains the $(1-1/e-\varepsilon)$-approximation ratio achieved by Kempe et al.'s approach.

Table 2: Datasets.

Name	n	m	Type	Corresponding Site
NetHEPT	15.2K	31.4K	undirected	arxiv.org
Pokec	1.6M	30.6M	directed	pokec.azet.sk
LiveJournal	4.8M	69.0M	directed	www.livejournal.com
Orkut	3.1M	117.2M	undirected	www.orkut.com
Twitter	41.7M	1.5G	directed	twitter.com

Very recently, Borg et al. [5] make a theoretical breakthrough on influence maximization by presenting a near-linear time algorithm under the independent cascade (IC) model (which is a special case of the triggering model). In particular, Borg et al.'s algorithm runs in $O(k\ell^2(m+n)\log^2 n/\varepsilon^3)$ time, and returns a $(1-1/e-\varepsilon)$-approximate solution with at least $1-1/n^\ell$ probability. As pointed out in [33], however, Borg et al.'s algorithm incurs significant overheads in practice, due to the ε^{-3} term and the large hidden constant factor in its time complexity. Tang et al. [33] address this deficiency and propose an improved algorithm (i.e., *TIM*) that runs in $O((k+\ell)(m+n)\log n/\varepsilon^2)$ while providing the same approximation guarantee with Borg et al.'s approach. In addition, Tang et al. show that *TIM* can be extended to support the triggering model, and it offers much higher empirical efficiency than any existing method that returns a $(1-1/e-\varepsilon)$-approximation under the same model. Nevertheless, as we explain in Section 3.4, *TIM* is inferior to *IMM* in that it incurs unnecessary computational costs.

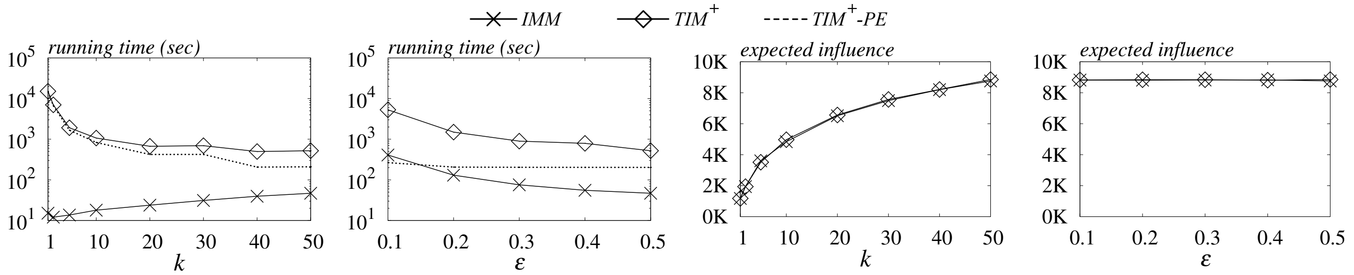
In addition, Du et al. [15] present a $(1-1/e-\varepsilon)$-approximation algorithm (i.e., *ConTinEst*) for influence maximization under the CTIC model. The algorithm adopts a greedy framework similar to that of Kempe et al.'s approach [25], but it applies an advanced sketch method [11] to estimate the expected influence of all relevant node sets in a batch manner, which considerably reduces computation overheads. Nevertheless, as we discuss on Section 4.2, *ConTinEst*'s approximation guarantees rely on parameters that are unknown in advance, and its time complexity is at least a factor of k^2 larger than that of *IMM*.

6. EXPERIMENTS

This section experimentally evaluates *IMM* against the states of the art, on a linux machine with an Intel Xeon 2.4GHz CPU and 64GB memory.

6.1 Experimental Settings

Datasets. We use five real social networks in our experiments, as shown in Table 2. Among them, *Pokec*, *LiveJournal*, *Orkut*, and *Twitter* are the largest datasets ever used in the literature of influence maximization. All datasets are publicly available: *Pokec*, *LiveJournal*, and *Orkut* can be obtained from [1], while *NetHEPT* and *Twitter* can be obtained from [2] and [3], respectively.



(a) Running time vs. k ($\varepsilon = 0.5$) (b) Running time vs. ε ($k = 50$) (c) Expected influence vs. k ($\varepsilon = 0.5$) (d) Expected influence vs. ε ($k = 50$)

Figure 3: Running time and expected influence on *Twitter* under the IC model ($l = 1$ for *IMM* and *TIM*$^+$).

Diffusion Models. We consider three well-adopted diffusion models in our experiments, namely, the *independent cascade (IC)* model [25] (see Section 2.2), the *linear threshold (LT)* model [25], and the *continuous time independent cascade (CTIC)* model [6, 16, 30] (see Section 4.2). For the IC model, we set the propagation probability $p(e)$ of each edge e to $1/i$, where i denotes the number of edges that share the same ending point with e. This setting is adopted in previous work [7, 19, 24, 36]. For the CTIC model, we set the length distribution of each edge e to be a *Weibull distribution* [28], with the following pdf:

$$f(x) = \frac{a}{b} \cdot (\frac{x}{b})^{a-1} \cdot e^{-(x/b)^a},$$

where a and b are sampled from $[0, 10]$ uniformly at random for each edge e. In addition, the threshold parameter t of the CTIC model is varied from 1 to 10 in our experiments. These settings follow from the experiments in [15] for evaluating *ConTinEst* [15].

Recall from Section 2.2 that the LT model is a special case of the triggering model where each sample from a node v's triggering distribution $\mathcal{T}(v)$ is a unit set containing only one incoming neighbor of v. Following previous work [6], we construct $\mathcal{T}(v)$ as follows. First, for each of v's incoming neighbors u, we assign to u a number selected from $[0, 1]$ uniformly at random. Then, we normalize the numbers assigned to all incoming neighbors, so that they sum up to one. Finally, we set the normalized number of each incoming neighbor as the probability that it is sampled from $\mathcal{T}(v)$.

Algorithms. We compare *IMM* with five algorithms, namely, *TIM* [33], *TIM*$^+$ [33], *IRIE* [24], *SIMPATH* [21], and *ConTinEst* [15]. In particular, *TIM* and *TIM*$^+$ are the state-of-the-art techniques that return $(1 - 1/e - \varepsilon)$-approximations under the triggering model, while *ConTinEst* is a method the yields the same approximation result under the CTIC model. On the other hand, *IRIE* and *SIMPATH* are the most advanced heuristic solutions under the IC and LT models, respectively. We implement *IMM* with C++, and we adopt the C++ implementations of the other algorithms made available by their inventors.

Parameter Settings. For *IMM*, *TIM*, *TIM*$^+$, we set $\varepsilon = 0.5$, $\ell = 1$, and $k = 50$ by default. For *IRIE* and *SIMPATH*, we set their internal parameters as recommended in [24] and [21], respectively. In each of our experiments, we run each method 5 times and report the average measurements. In addition, to compare the quality of the seed sets returned by each method, we estimate the expected influence of any seed set S by taking its average influence in 10000 simulations of the diffusion process.

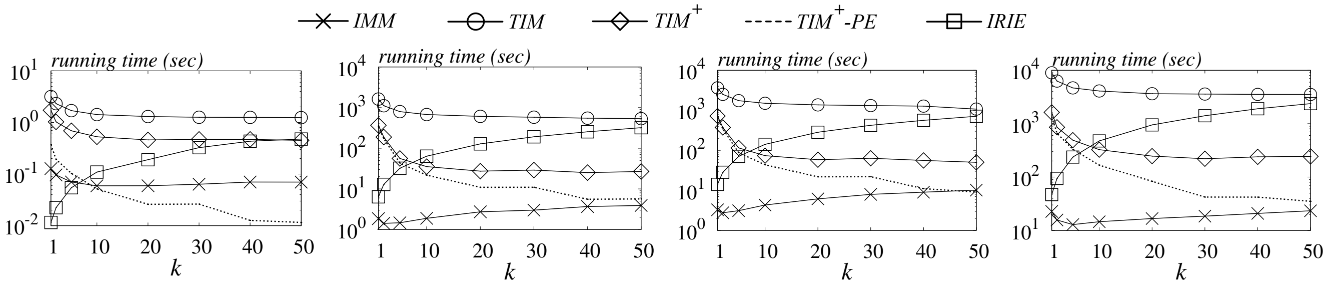
6.2 Results under the IC Model

In our first set of experiments, we compare four methods that support the IC models, namely, *IMM*, *TIM*, *TIM*$^+$, and *IRIE*. Figure 3a shows the computational costs of *IMM* and *TIM*$^+$ on *Twitter* (i.e., our largest dataset), with k varying from 1 to 50. We omit *TIM* and *IRIE*, since the former incurs prohibitive cost on *Twitter*, while the latter requires more than 64G memory to process the data. Observe that *IMM* consistently outperforms *TIM*$^+$ in terms of running time, and is faster than *TIM*$^+$ by three orders of magnitude when $k = 1$. The reason that *TIM*$^+$ is inferior to *IMM* is two-fold. First, as we point out in Section 3, the node selection phase of *TIM*$^+$ requires generating a large number of RR sets, due to the facts that (i) it cannot reuse RR sets produced in the parameter estimation phase, and (ii) it makes sub-optimal choices when setting its internal parameters. Second, the parameter estimation phase of *TIM*$^+$ (denoted as *TIM*$^+$*-PE*) incurs significant overheads when k is small. The rationale is that, when k decreases, the optimal expected influence OPT also tends to decrease, in which case it is more challenging to derive a reasonable lower bound of OPT. To illustrate *TIM*$^+$*-PE*'s computational overhead, we plot it in Figure 3 along with *IMM* and *TIM*$^+$. Observe that *TIM*$^+$*-PE* runs even slower than *IMM*. Furthermore, when k = 1, *TIM*$^+$*-PE* accounts for 97% of *TIM*'s computation time. In contrast, *IMM* does not suffer from this issue, since it adopts a more effective statistical test in its sampling phase.

Figure 3b illustrates the running time of *IMM* and *TIM*$^+$ on *Twitter* as a function of ε. Regardless of the value of ε, the processing cost of *IMM* is lower than that of *TIM*$^+$ by an order of magnitude. The computation time of both *IMM* and *TIM* decreases with ε, since a larger ε leads to a less stringent theoretical guarantee, which is easier to achieve. The overhead of *TIM*$^+$*-PE* is insensitive to ε, which is consistent with the fact that its time complexity is independent of ε. Figure 3c (resp. Figure 3d) shows the expected influence of *IMM* and *TIM*$^+$ when k (resp. ε) varies. In all cases, *IMM* and *TIM*$^+$ yield similar results.

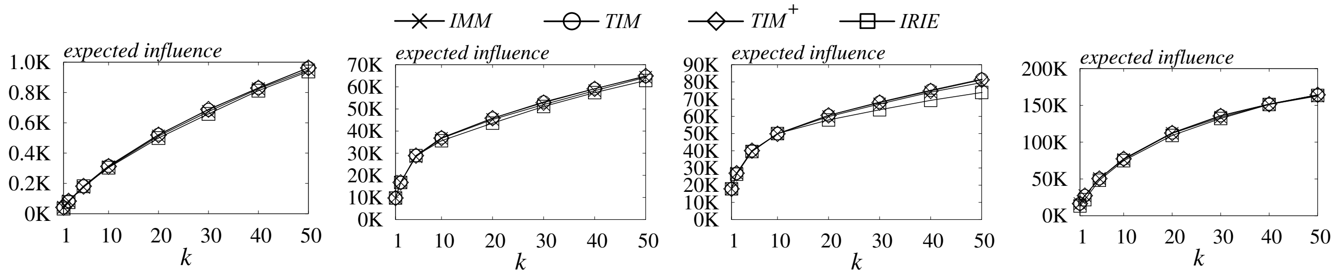
Figure 4 plots the running time of *IMM*, *TIM*, *TIM*$^+$, and *IRIE* against k for the four smaller datasets. Again, *IMM* consistently outperforms *TIM*$^+$, especially when k is small. Furthermore, the parameter estimation phase of *TIM*$^+$ (i.e., *TIM*$^+$*-PE*) alone incurs a higher cost than *IMM* in most cases. On the other hand, *TIM*$^+$ is more efficient *TIM*, due to the heuristic optimization that it adopts to reduce computational overhead of its node selection phase (see Section 2.3). *IRIE* outperforms *IMM* on *NetHEPT* when $k \leq 5$, but is inferior to *IMM* in all other cases. In particular, the running time of *IRIE* is around two orders of magnitude larger than that of *IMM* on *Pokec*, *LiveJournal*, and *Orkut*, when $k \geq 20$.

Figure 5 shows the expected influence of each method on the four smaller datasets. *IMM*, *TIM*, and *TIM*$^+$ produce comparable results in all scenarios. Meanwhile, on *LiveJournal*, the quality of seed sets returned by *IRIE* are noticeably worse than those returned by *IMM*, *TIM*, and *TIM*$^+$. The reason is that *IMM*, *TIM*, and *TIM*$^+$ all provide asymptotic guarantees in terms of the accuracy of their solutions, whereas *IRIE* is a heuristic method that does not offer any worst-case assurance.



(a) *NetHEPT* (b) *Pokec* (c) *LiveJournal* (d) *Orkut*

Figure 4: Running time vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and *TIM*$^+$).



(a) *NetHEPT* (b) *Pokec* (c) *LiveJournal* (d) *Orkut*

Figure 5: Expected influence vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and *TIM*$^+$).

Remark. We have also evaluated the effect of ε on the performance of *IMM*, *TIM*, and *TIM*$^+$ on the four smaller datasets. We vary ε from 0.1 to 0.5, and find that *IMM* consistently surpasses both *TIM* and *TIM*$^+$ in terms of efficiency, while achieving similar expected influence. We omit the results due to the space constraint.

6.3 Results under the LT model

Our second set of experiments compares four methods that support the LT model, i.e., *IMM*, *TIM*, *TIM*$^+$, and *SIMPATH*. Figure 6 shows the running time and expected influence of *IMM* and *TIM*$^+$ on *Twitter*, omitting *TIM* and *SIMPATH* since they entail prohibitive costs on the dataset. As with the case under the IC model, *IMM* considerably outperforms *TIM*$^+$ in all cases, and is even more efficient than the parameter estimation phase of *TIM*$^+$ (denoted as *TIM*$^+$*-PE*). At the same time, there is no noticeable difference between the quality of the results returned by *IMM* and *TIM*$^+$.

Figure 7 (resp. Figure 8) illustrates the computational overheads (resp. expected influence) of all methods on the four smaller datasets. The relative performance of *IMM*, *TIM*, and *TIM*$^+$ is qualitatively similar to that in Figures 4 and 5, with *IMM* surpassing *TIM* and *TIM*$^+$ in terms of efficiency. On the other hand, *SIMPATH* incurs a much larger overheads than all other methods, and its result accuracy on on *Pokec* and *LiveJournal* is considerably lower than that of *IMM*, *TIM*, and *TIM*$^+$.

6.4 Results under the CTIC Model

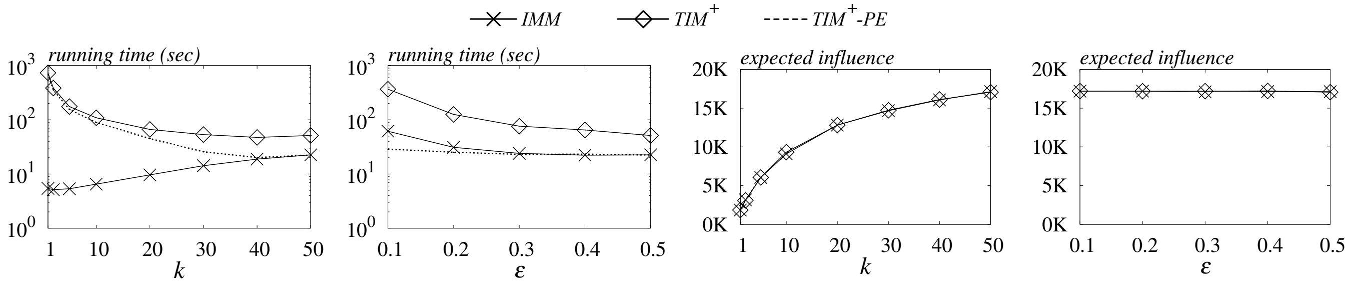
Our last set of experiments compares *IMM* with *ConTinEst* under the CTIC model. We first consider the case when both *IMM* and *ConTinEst* return $(1 - 1/e - \varepsilon)$-approximate solutions with $\varepsilon \leq 0.5$. Towards this end, we tune *ConTinEst*'s parameters c and d as follows. First, we set $d = 5$, following the experiments in [15]. Then, we let c be the smallest value that satisfies Equation 15. Note that this setting is to the advantage of *ConTinEst* because, as we analyze in Section 4.2, *ConTinEst* may require a much larger c to ensure $(1 - 1/e - \varepsilon)$-approximations.

Figure 9a illustrates the computation time of *IMM* and *ConTinEst* on *NetHEPT*, when $\varepsilon = 0.5$, $t = 10$, and k varies from 1 to 5. (Recall that t is the threshold parameter in the CTIC model.) We use only *NetHEPT* in this experiment, since *ConTinEst* incurs prohibitive overheads on the larger datasets. Observe that the running time of *ConTinEst* increases significantly with k. The reason is that, when the approximation ratio of *ConTinEst* if fixed, its computational cost is roughly proportional to k^3, as we analyze in Section 6.4. In contrast, *IMM* scales much better with k, and it is three orders of magnitude faster than *ConTinEst* when $k = 5$.

Figure 9b shows the overheads of *IMM* and *ConTinEst* when $\varepsilon = 0.5$, $k = 3$, and t varies from 1 to 10. The running time of *IMM* increases with t. To explain, recall that under the CTIC model, *IMM* generates each RI set by invoking the Dijkstra's algorithm on a weighted version of G to identify nodes that are within t distance to a given node v. When t increases, there exists a larger number of nodes in G that are within t distance to v, and hence, the construction of RI sets incurs a larger overhead. This renders *IMM* less efficient. In contrast, the computational cost of *ConTinEst* is insensitive to t. That said, even when $t = 10$, *IMM* is still more than 250 times faster than *ConTinEst*. Finally, Figure 9c plots the processing costs of *IMM* and *ConTinEst* when $k = 3$, $t = 10$, and ε varies from 0.1 to 0.5. The costs of *IMM* and *ConTinEst* both decrease with the increase of ε, but *IMM* is significantly more efficient than *ConTinEst* in all cases.

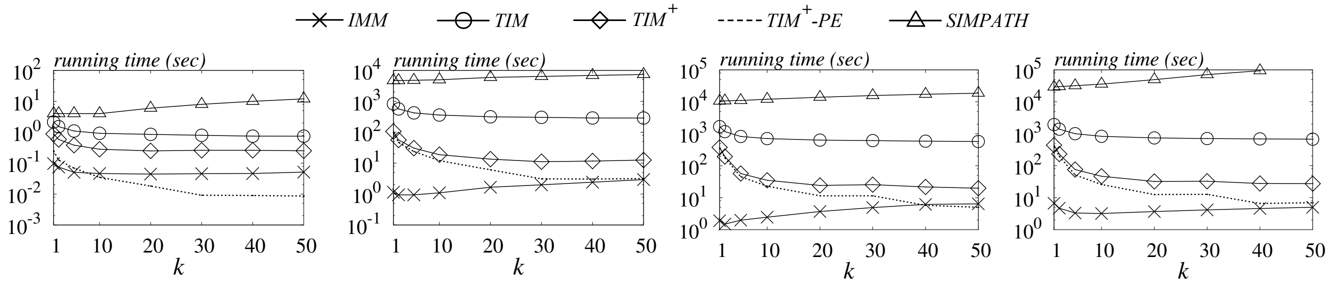
7. CONCLUSION

This paper presents *IMM*, an influence maximization algorithm runs in $O((k + \ell)(n + m)\log n/\varepsilon^2)$ expected time and returns a $(1 - 1/e - \varepsilon)$-approximation with at least $1 - 1/n^\ell$ probability under the triggering model. *IMM* has the same approximation guarantee and time complexity as the state of the art, but achieves higher empirical efficiency with a novel algorithm design based on martingales. In addition, *IMM* can be extended to a large class of diffusion models beyond the triggering model, while still retaining



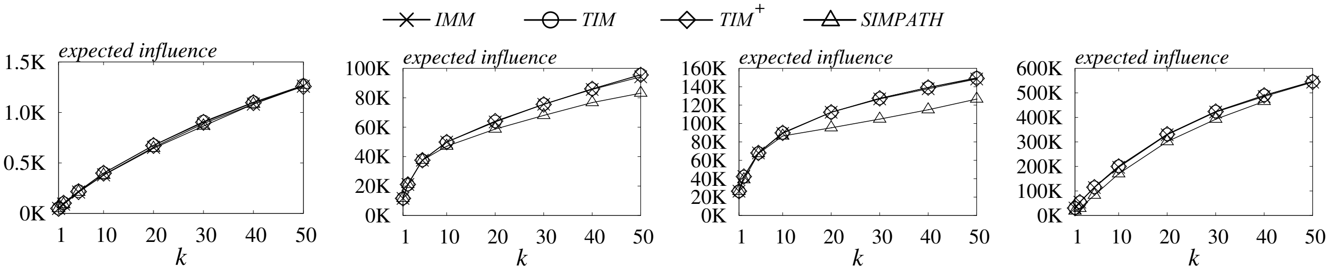
(a) Running time vs. k ($\varepsilon = 0.5$) (b) Running time vs. ε ($k = 50$) (c) Expected influence vs. k ($\varepsilon = 0.5$) (d) Expected influence vs. ε ($k = 50$)

Figure 6: Running time and expected influence on *Twitter* under the LT model ($l = 1$ for *IMM* and *TIM*$^+$).



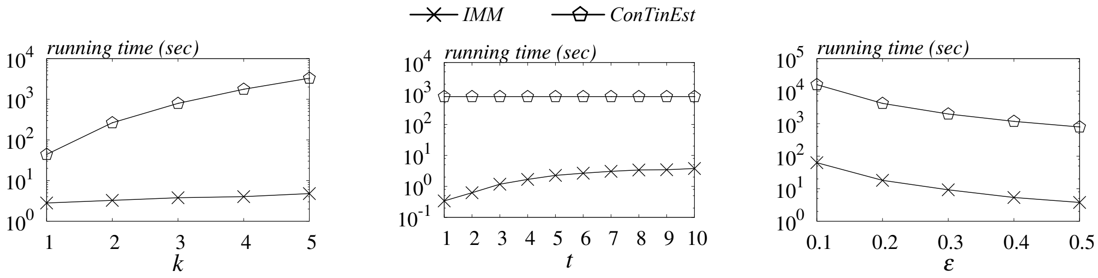
(a) *NetHEPT* (b) *Pokec* (c) *LiveJournal* (d) *Orkut*

Figure 7: Running time vs. k under the LT model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and *TIM*$^+$).



(a) *NetHEPT* (b) *Pokec* (c) *LiveJournal* (d) *Orkut*

Figure 8: Expected influence vs. k under the LT model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and *TIM*$^+$).



(a) Vary k ($t = 10$ and $\varepsilon = 0.5$). (b) Vary t ($k = 3$ and $\varepsilon = 0.5$). (c) Vary ε ($k = 3$ and $t = 10$).

Figure 9: Running time of *IMM* and *ConTinEst* under the CTIC model.

the $(1 - 1/e - \varepsilon)$-approximation ratio. We experimentally evaluate *IMM* against the state-of-the-art methods under the IC, LT, triggering, and CTIC models, using real social networks with up to 1.4 billion edges. Our experimental results show that *IMM* consistently outperforms the states of the art in terms of computation efficiency, and is often orders of magnitude faster. For future work, we plan to investigate a precise characterization of the diffusion models for which our extension of *IMM* (in Section 4.2) is applicable.

8. ACKNOWLEDGMENTS

This work was supported by AcRF Tier 2 Grant ARC19/14 from the Ministry of Education, Singapore, an SUG Grant from the Nanyang Technological University, and a gift from the Microsoft Research Asia. The authors would like to thank the anonymous reviewers for their constructive and insightful comments.

9. REFERENCES

[1] `https://snap.stanford.edu/data/`.

[2] `http://research.microsoft.com/en-us/people/weic/graphdata.zip`.

[3] `http://an.kaist.ac.kr/traces/WWW2010.html`.

[4] S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. In *WINE*, pages 306–311, 2007.

[5] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier. Maximizing social influence in nearly optimal time. In *SODA*, pages 946–957, 2014.

[6] W. Chen, W. Lu, and N. Zhang. Time-critical influence maximization in social networks with time-delayed diffusion process. In *AAAI*, 2012.

[7] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, pages 1029–1038, 2010.

[8] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, pages 199–208, 2009.

[9] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*, pages 88–97, 2010.

[10] F. R. K. Chung and L. Lu. Concentration inequalities and martingale inequalities: A survey. *Internet Mathematics*, 3(1):79–127, 2006.

[11] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Syst. Sci.*, 55(3):441–453, 1997.

[12] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, NY, USA, 1986.

[13] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, 1:269–271, 1959.

[14] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, pages 57–66, 2001.

[15] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, pages 3147–3155, 2013.

[16] M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *ICML*, pages 561–568, 2011.

[17] M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Modeling information propagation with survival theory. In *ICML*, pages 666–674, 2013.

[18] M. Gomez-Rodriguez and B. Schölkopf. Influence maximization in continuous time diffusion networks. In *ICML*, 2012.

[19] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 5(1):73–84, 2011.

[20] A. Goyal, W. Lu, and L. V. S. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *WWW*, pages 47–48, 2011.

[21] A. Goyal, W. Lu, and L. V. S. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *ICDM*, pages 211–220, 2011.

[22] E. M. J. Goldenberg, B. Libai. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12(3):211–223, 2001.

[23] E. M. J. Goldenberg, B. Libai. Using complex systems analysis to advance marketing theory development. *American Journal of Sociology*, 9:1, 2001.

[24] K. Jung, W. Heo, and W. Chen. Irie: Scalable and robust influence maximization in social networks. In *ICDM*, pages 918–923, 2012.

[25] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, pages 137–146, 2003.

[26] D. Kempe, J. M. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *ICALP*, pages 1127–1138, 2005.

[27] J. Kim, S.-K. Kim, and H. Yu. Scalable and parallelizable processing of influence maximization for large-scale social networks. In *ICDE*, pages 266–277, 2013.

[28] J. F. Lawless. *Statistical Models and Methods for Lifetime Data*. Wiley-Interscience, 2002.

[29] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.

[30] B. Liu, G. Cong, D. Xu, and Y. Zeng. Time constrained influence maximization in social networks. In *ICDM*, pages 439–448, 2012.

[31] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD*, pages 61–70, 2002.

[32] L. Seeman and Y. Singer. Adaptive seeding in social networks. In *FOCS*, pages 459–468, 2013.

[33] Y. Tang, X. Xiao, and Y. Shi. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*, pages 75–86, 2014.

[34] V. V. Vazirani. *Approximation Algorithms*. Springer, 2002.

[35] M. D. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Trans. Software Eng.*, 17(9):972–975, 1991.

[36] C. Wang, W. Chen, and Y. Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Min. Knowl. Discov.*, 25(3):545–576, 2012.

[37] Y. Wang, G. Cong, G. Song, and K. Xie. Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *KDD*, pages 1039–1048, 2010.

[38] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.

APPENDIX

Proof of Lemma 3. Define a random variable x_i for each $R_i \in \mathcal{R}$, such that $x_i = 1$ if $S_k^\circ \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. Then, $F_{\mathcal{R}}(S_k^\circ) = \sum_{i=1}^{\theta} x_i/\theta$. Let $p = \mathbb{E}\left[F_{\mathcal{R}}(S_k^\circ)\right]$. By Lemma 1,

$$p \;=\; \mathbb{E}\left[F_{\mathcal{R}}(S_k^\circ)\right] \;=\; \mathbb{E}\left[I(S_k^\circ)\right]/n \;=\; OPT/n.$$

By Corollary 2,

$$\begin{aligned}
&\Pr\left[n \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1-\varepsilon_1)\cdot OPT\right] \\
&\quad= \Pr\left[n \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1-\varepsilon_1)\cdot np\right] \\
&\quad= \Pr\left[\theta \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1-\varepsilon_1)\cdot \theta p\right] \\
&\quad= \Pr\left[\textstyle\sum_{i=1}^{\theta} x_i - \theta p \leq -\varepsilon_1 \cdot \theta p\right] \\
&\quad\leq \exp\Big(-\varepsilon_1^2 \cdot p\theta/2\Big) \;\leq\; \exp\Big(-\varepsilon_1^2 \cdot p\theta_1/2\Big) \\
&\quad= \delta_1.
\end{aligned}$$

Thus, the lemma is proved. □

Proof of Lemma 4. Let S_k be an arbitrary size-k node set. We say that S_k is *bad*, if $\mathbb{E}[I(S_k)] < (1-1/e-\varepsilon)\cdot OPT$. To prove the lemma, we will show that each bad size-k node set has at most $\delta_2/\binom{n}{k}$ probability to be returned by the node selection phase of *IMM*. This suffices to establish the lemma because (i) there exist only $\binom{n}{k}$ bad size-k node sets, and (ii) if each of them has at most $\delta_2/\binom{n}{k}$ probability to be returned, then by the union bound, there is at least $1-\delta_2$ probability that none of them is output by the node selection phase.

Consider any bad size-k node set S_k. Define a random variable x_i for each $R_i \in \mathcal{R}$, such that $x_i = 1$ if $S_k \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. We have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i/\theta$. Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n$ and $\varepsilon_2 = \varepsilon - (1-1/e)\cdot\varepsilon_1$. We have

$$\begin{aligned}&\Pr\left[n\cdot F_{\mathcal{R}}(S_k) - \mathbb{E}[I(S_k)] \geq \varepsilon_2\cdot OPT\right]\\ =\;& \Pr\left[\theta\cdot F_{\mathcal{R}}(S_k) - \theta p \geq \tfrac{\varepsilon_2\cdot OPT}{np}\cdot\theta p\right]\end{aligned} \tag{17}$$

Let $\zeta = \frac{\varepsilon_2\cdot OPT}{np}$. By Corollary 1 and $p < ((1-1/e-\varepsilon)\cdot OPT)/n$,

r.h.s of Eqn. 17

$$\begin{aligned}
&\leq \exp\left(-\frac{\zeta^2}{2+\frac{2}{3}\zeta}\cdot\theta p\right)\\
&= \exp\left(-\frac{\varepsilon_2^2\cdot OPT^2}{2n^2p+\frac{2}{3}\varepsilon_2 n\cdot OPT}\cdot\theta\right)\\
&< \exp\left(-\frac{\varepsilon_2^2\cdot OPT^2}{2n(1-1/e-\varepsilon)\cdot OPT+\frac{2}{3}\varepsilon_2 n\cdot OPT}\cdot\theta\right)\\
&< \exp\left(-\frac{(\varepsilon-(1-1/e)\varepsilon_1)^2\cdot OPT}{(2-2/e)\cdot n}\cdot\theta\right)\\
&\leq \exp\left(-\frac{(\varepsilon-(1-1/e)\varepsilon_1)^2\cdot OPT}{(2-2/e)\cdot n}\cdot\theta_2\right)\\
&\leq \delta_2/\tbinom{n}{k}.
\end{aligned}$$

Therefore, the lemma holds. □

Proof of Theorem 1. By Lemma 4, $\mathbb{E}[I(S_k^*)] \geq (1-1/e-\varepsilon)\cdot OPT$ holds with at least $1-\delta_2$ probability under the condition that Equation 3 holds. And by Lemma 3, Equation 3 holds with at least $1-\delta_1$ probability. By the union bound, $\mathbb{E}[I(S_k^*)] \geq (1-1/e-\varepsilon)\cdot OPT$ holds with at least $1-\delta_1-\delta_2 \geq 1/n^\ell$ probability. Thus, the theorem is proved. □

Proof of Lemma 5. We rewrite θ_1 (resp. θ_2) as a function f_1 (resp. f_2) as follows:

$$\begin{aligned}
f_1(\varepsilon_1,\delta_1) &= \frac{2n\cdot\log(1/\delta_1)}{OPT\cdot\varepsilon_1^2}, \text{ and}\\
f_2(\varepsilon_1,\delta_2) &= \frac{(2-2/e)\cdot n\cdot\log\left(\binom{n}{k}/\delta_2\right)}{OPT\cdot(\varepsilon-(1-1/e)\cdot\varepsilon_1)^2}.
\end{aligned}$$

Observe that f_1 monotonically decreases with the increase of ε_1 or δ_1, and f_2 monotonically increases with ε_1 but monotonically decreases with the increase of δ_2. Assume that θ^o is obtained when $\varepsilon_1 = \varepsilon_1^o$, $\delta_1 = \delta_1^o$, and $\delta_2 = \delta_2^o$. Since θ^o is the smallest possible θ, the following equations hold:

$$\begin{aligned}
f_1(\varepsilon_1^o,\delta_1^o) &= f_2(\varepsilon_1^o,\delta_2^o) = \theta^o, \quad\text{and}\\
\delta_1^o+\delta_2^o &= 1/n^\ell.
\end{aligned}$$

Meanwhile, we have $\theta^* = f_1(\varepsilon_1^*, 1/(2n^\ell)) = f_2(\varepsilon_1^*, 1/(2n^\ell))$, where $\varepsilon_1^* = \varepsilon\cdot\frac{\alpha}{(1-1/e)\cdot\alpha+\beta}$. Since $\theta^o \leq \theta$, we have

$$f_1(\varepsilon_1^o,\delta_1^o) \leq f_1\left(\varepsilon_1^*, 1/(2n^\ell)\right) \tag{18}$$

The remaining part of our proof consists of three steps. First, we prove that $f_1(\varepsilon_1', 1/n^\ell) < f_1(\varepsilon_1^o,\delta_1^o)$, where ε_1' is the positive real root of the equation $f_1(\varepsilon_1', 1/n^\ell) = f_2(\varepsilon_1', 1/n^\ell)$. For this purpose, we consider two cases based on whether $\varepsilon_1^o \leq \varepsilon_1'$. When $\varepsilon_1^o \leq \varepsilon_1'$, it can be verified that $f_1(\varepsilon_1', 1/n^\ell) < f_1(\varepsilon_1^o,\delta_1^o)$ given $\delta_1^o < 1/n^\ell$. Meanwhile, if $\varepsilon_1^o > \varepsilon_1'$, we have

$$f_1(\varepsilon_1', 1/n^\ell) = f_2(\varepsilon_1', 1/n^\ell) < f_2(\varepsilon_1^o,\delta_2^o) = f_1(\varepsilon_1^o,\delta_1^o). \tag{19}$$

Second, we show that $M \leq \frac{\ell\log n+\log 2}{\ell\log n}\cdot f_1(\varepsilon_1', 1/n^\ell)$, where $M = \max\left\{f_1(\varepsilon_1', 1/(2n^\ell)), f_2(\varepsilon_1', 1/(2n^\ell))\right\}$. We have

$$\begin{aligned}
\frac{f_1(\varepsilon_1', 1/(2n^\ell))}{f_1(\varepsilon_1', 1/n^\ell)} &= \frac{\ell\log n+\log 2}{\ell\log n}, \text{ and}\\
\frac{f_2(\varepsilon_1', 1/(2n^\ell))}{f_1(\varepsilon_1', 1/n^\ell)} &= \frac{f_2(\varepsilon_1', 1/(2n^\ell))}{f_2(\varepsilon_2', 1/n^\ell)}\\
&= \frac{\ell\log n+\log\binom{n}{k}+\log 2}{\ell\log n+\log\binom{n}{k}}\\
&< \frac{\ell\log n+\log 2}{\ell\log n}.
\end{aligned}$$

It then follows that

$$M \leq \frac{\ell\log n+\log 2}{\ell\log n}\cdot f_1(\varepsilon_1', 1/n^\ell). \tag{20}$$

Finally, we prove that $f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq M$. We consider two cases based on whether $\varepsilon_1^* \geq \varepsilon_1'$. When $\varepsilon_1^* \geq \varepsilon_1'$, we have

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq f_1(\varepsilon_1', 1/(2n^\ell)) \leq M.$$

Meanwhile, if $\varepsilon_1^* < \varepsilon_1'$, we have

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) = f_2(\varepsilon_1^*, 1/(2n^\ell)) \leq f_2(\varepsilon_1', 1/(2n^\ell)) \leq M.$$

Therefore, the following equation holds

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq M. \tag{21}$$

Combining Equations 18, 19, 20 and 21, we have

$$f_1(\varepsilon_1^o,\delta_1^o) \leq f_1(\varepsilon_1^*, 1/(2n^\ell)) < \frac{\ell\log n+\log 2}{\ell\log n}\cdot f_1(\varepsilon_1^o,\delta_1^o).$$

Thus, the lemma is proved. □

Proof of Lemma 6. Let S_k be an arbitrary size-k node set and $p = \mathbb{E}[F_{\mathcal{R}}(S_k)]$. Define a random variable x_i for each $R_i \in \mathcal{R}$ such that $x_i = 1$ if $S_k \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. Then, we have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i/\theta$ and

$$p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n \leq OPT/n < x/n. \tag{22}$$

let $\zeta = \frac{(1+\varepsilon')\cdot x}{np} - 1$. By Equation 22, we have $\zeta > \varepsilon'\cdot x/(np) > \varepsilon'$. Then, by Corollary 1,

$$\begin{aligned}
&\Pr[n\cdot F_{\mathcal{R}}(S_k) \geq (1+\varepsilon')\cdot x]\\
&= \Pr\left[\theta\cdot F_{\mathcal{R}}(S_k) - \theta p \geq (\frac{(1+\varepsilon')\cdot x}{np} - 1)\cdot\theta p\right]\\
&\leq \exp\left(-\frac{\zeta^2}{2+2\zeta/3}\cdot\theta p\right)\\
&< \exp\left(-\frac{\varepsilon'\cdot x/(np)}{2/\zeta+2/3}\cdot\frac{(2+\frac{2}{3}\varepsilon')\cdot\log(\binom{n}{k}/\delta_3)}{\varepsilon'^2}\cdot\frac{np}{x}\right)\\
&< \exp\left(-\frac{1}{2/\varepsilon'+2/3}\cdot\frac{(2+\frac{2}{3}\varepsilon')\cdot\log(\binom{n}{k}/\delta_3)}{\varepsilon'}\right)\\
&= \delta_3/\tbinom{n}{k}.
\end{aligned}$$

By the union bound, we can have $n \cdot F_{\mathcal{R}}(S_k^*) < (1+\varepsilon') \cdot x$ with at least $1 - \delta_3$ probability. Thus, the lemma is proved. □

Proof of Lemma 7. Let S_k be an arbitrary size-k node set. We say that S_k is *bad*, if $OPT < n \cdot \mathbb{E}[F_{\mathcal{R}}(S_k)]/(1+\varepsilon')$. To prove the lemma, we will show that each bad size-k node set has at most $\delta_3/\binom{n}{k}$ probability to be returned by Algorithm 1. This suffices to establish the lemma because (i) there exist only $\binom{n}{k}$ bad size-k node sets, and (ii) if each of them has at most $\delta_3/\binom{n}{k}$ probability to be returned, then by the union bound, there is at least $1-\delta_3$ probability that none of them is output by the node selection phase.

Consider any bad size-k node set S_k. Define a random variable x_i for each $R_i \in \mathcal{R}$, such that $x_i = 1$ if $S_k \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. We have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i/\theta$. Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n$. We have

$$\begin{aligned} &\Pr\left[OPT < n \cdot F_{\mathcal{R}}(S_k)/(1+\varepsilon')\right] \\ &\quad\le\ \Pr\left[\theta \cdot F_{\mathcal{R}}(S_k) - \theta p \ge \tfrac{\varepsilon' \cdot OPT}{np} \cdot \theta p\right] \end{aligned} \tag{23}$$

Let $\zeta = \frac{\varepsilon' \cdot OPT}{np}$. By Corollary 1 and $OPT \ge x$,

r.h.s of Eqn. 23

$$\begin{aligned} &\le\ \exp\left(-\frac{\zeta^2}{2+\frac{2}{3}\zeta}\cdot\theta p\right) \\ &=\ \exp\left(-\frac{\varepsilon'^2 \cdot OPT^2}{2n^2 p + \frac{2}{3}\varepsilon' n \cdot OPT}\cdot\theta\right) \\ &<\ \exp\left(-\frac{\varepsilon'^2 \cdot OPT}{2n + \frac{2}{3}\varepsilon' n}\cdot\frac{2+\frac{2}{3}\varepsilon'}{\varepsilon'^2}\cdot\frac{n}{x}\cdot\log(\tbinom{n}{k}/\delta_3)\right) \\ &<\ \exp\left(-(OPT/x)\cdot\log(\tbinom{n}{k}/\delta_3)\right) \\ &\le\ \delta_3/\tbinom{n}{k}. \end{aligned}$$

Therefore, the lemma holds. □

Proof of Theorem 2. Let $j = \lceil \log_2 \frac{n}{OPT} \rceil$. By Lemma 6 and the union bound, Algorithm 2 has at most $(j-1)/(n^\ell \cdot \log_2 n)$ probability to terminate its for loop before the j-th iteration. Then, by Lemma 7 and the union bound, Algorithm 2 sets $LB \le OPT$ with at least $1 - n^\ell$ probability. Therefore, the theorem holds. □

Proof of Lemma 8. Let S_k^+ be the size-k node set that maximizes $F_{\mathcal{R}}(S_k^+)$. By the properties of the greedy algorithm for maximum coverage, we have

$$F_{\mathcal{R}}(S_k^*)\ \ge\ (1-1/e)\cdot F_{\mathcal{R}}(S_k^+)\ \ge\ (1-1/e)\cdot F_{\mathcal{R}}(S_k^\circ).$$

Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k^\circ)]$. We have $p = OPT/n$. Since $OPT \ge c \cdot \frac{(1+\varepsilon')^2}{1-1/e}\cdot x$,

$$\begin{aligned} &\Pr[n \cdot F_{\mathcal{R}}(S_k^*) \le (1+\varepsilon')\cdot x] \\ &\quad\le\ \Pr[n(1-1/e)\cdot F_{\mathcal{R}}(S_k^\circ) \le (1+\varepsilon')\cdot x] \\ &\quad\le\ \Pr[n \cdot F_{\mathcal{R}}(S_k^\circ) \le OPT/(1+\varepsilon')] \\ &\quad\le\ \Pr[\theta_i \cdot F_{\mathcal{R}}(S_k^\circ) - \theta_i p \le -\varepsilon' \cdot \theta_i p/(1+\varepsilon')] \end{aligned} \tag{24}$$

Let $\zeta = \varepsilon'/(1+\varepsilon')$. By Corollary 2,

$$\begin{aligned} &\text{r.h.s of Eqn. 24} \le \exp\left(-\frac{\zeta^2}{2}\cdot\theta_i p\right) \\ &=\ \exp\left(-\frac{\varepsilon'^2}{2\cdot(1+\varepsilon')^2}\cdot\frac{2+2\varepsilon'/3}{\varepsilon'^2}\cdot\frac{np}{x}\cdot\log(\tbinom{n}{k}/\delta_3)\right) \\ &=\ \exp\left(-\frac{1+\varepsilon'/3}{(1+\varepsilon')^2}\cdot\frac{OPT}{x}\cdot\log(\tbinom{n}{k}/\delta_3)\right) \\ &\le\ \exp\left(-\frac{1+\varepsilon'/3}{(1+\varepsilon')^2}\cdot\frac{(1+\varepsilon')^2}{1-1/e}\cdot\log(\tbinom{n}{k}/\delta_3)\cdot c\right) \\ &<\ (\delta_3/\tbinom{n}{k})^c \end{aligned}$$

By the union bound, for all possible size-k node set S_k^* we have $n \cdot F_{\mathcal{R}}(S_k^*) \le (1+\varepsilon')\cdot x$ holds with at most $(\delta_3)^c$ probability. Thus, the lemma is proved. □

Proof of Lemma 9. Assume that $OPT \ge \frac{(1+\varepsilon')^2}{1-1/e}\cdot x$ first holds in the j-th iteration. Let $z \ge j$, and $c = 2^{z-j}$. Then, in the z-th iteration, we have $OPT \ge c \cdot \frac{(1+\varepsilon')^2}{1-1/e}\cdot\frac{n}{2^z}$. Let I_z be the event that *IMM* executes the z-th iteration of its for loop. By Lemma 8,

$$P[I_z]\ \le\ 1/(n^\ell \cdot \log_2 n)^c\ \le\ 1/(n^\ell \cdot \log_2 n)$$

Taking into account all iterations from the j-th one to the $(\log_2 n - 1)$-th one, we have

$$\textstyle\sum_{z \ge j} P[I_z]\ \le\ 1/n^\ell$$

Thus, Algorithm 2 terminates before the j-th iteration with at least $1 - 1/n^\ell$ probability. In that case, we have $OPT \le \frac{(1+\varepsilon')^2}{1-1/e}\cdot x$, which leads to

$$LB\ =\ n \cdot F_{\mathcal{R}}(S_i)/(1+\varepsilon')\ \ge\ x\ \ge\ \frac{1-1/e}{(1+\varepsilon')^2}\cdot OPT.$$

Meanwhile, the expected total number of RR sets generated in the first $j-1$ iterations is at most

$$\begin{aligned} &\max\left\{\frac{\lambda^*}{LB}, \frac{\lambda'}{n/2^{j-1}}\right\}\cdot\left(1 - \textstyle\sum_{z\ge j} P[I_z]\right) \\ &\le\ \frac{2\max\{\lambda^*, \lambda'\}\cdot(1+\varepsilon')^2}{(1-1/e)\cdot OPT}\cdot(1-1/n^\ell) \\ &<\ \frac{2\max\{\lambda^*, \lambda'\}\cdot(1+\varepsilon')^2}{(1-1/e)\cdot OPT}. \end{aligned} \tag{25}$$

In addition, when $1/n^\ell \le 1/2$, the expected total number of RR sets from the j-th to the $(\log_2 n - 1)$-th iteration is at most

$$\begin{aligned} &\sum_{z \ge j} \max\left\{\frac{\lambda^*}{LB}, \frac{\lambda'}{n/2^z}\right\}\cdot P[I_z] \\ &=\ \textstyle\sum_{z\ge j}\dfrac{\max\{\lambda^*, \lambda'\}}{n/2^z}\cdot P[I_z] \\ &\le\ \frac{1}{\log_2 n}\cdot\textstyle\sum_{z\ge j}\dfrac{\max\{\lambda^*, \lambda'\}}{(n^\ell)^c \cdot n/2^z} \\ &\le\ \frac{\max\{\lambda^*, \lambda'\}}{n \cdot \log_2 n}\cdot\textstyle\sum_{z\ge j} 2^{-2^{z-j}+z} \\ &<\ \frac{\max\{\lambda^*, \lambda'\}}{n \cdot \log_2 n}\cdot\log_2 n \\ &\le\ \frac{\max\{\lambda^*, \lambda'\}\cdot(1+\varepsilon')^2}{(1-1/e)\cdot OPT}. \end{aligned} \tag{26}$$

Combining Equations 25 and 26, we have

$$\mathbb{E}[|\mathcal{R}|]\ \le\ \frac{3\max\{\lambda^*, \lambda'\}\cdot(1+\varepsilon')^2}{(1-1/e)\cdot OPT},$$

which proves the lemma. □

Proof of Theorem 4. By combining Theorems 1 and 2, we obtain that Algorithm 3 returns a $(1 - 1/e - \varepsilon)$-approximate solution with at least $1 - 1/n^\ell$. In the following, we focus on *IMM*'s time complexity. Observe that the computation cost of *IMM* mainly consists of two parts, namely, the cost of generating the set $\mathcal{R}$ of random RR sets, and the cost of invoking Algorithm 1. As discussed in Section 3.4, the expected total cost of constructing $\mathcal{R}$ is $O\left((k+\ell)(n+m)\log n/\varepsilon^2\right)$. Therefore, it suffices to prove that the total cost incurred by Algorithm 1 is no more than $O\left((k+\ell)(n+m)\log n/\varepsilon^2\right)$.

Observe that *IMM* invokes Algorithm 1 at most $\log_2 n$ times, i.e., once in each iteration of the for loop in Algorithm 2, and once for the node selection phase. Let R_i be the version of $\mathcal{R}$ in the i-th iteration of Algorithm 2. By Corollary 3, when given $\mathcal{R}_i$ as input, Algorithm 1 runs in $O(\sum_{R\in\mathcal{R}_i}|R|)$ time. Notice that $|\mathcal{R}_i|$ doubles after each iteration, i.e., $|\mathcal{R}_1|, |\mathcal{R}_2|, \ldots$ is a geometric sequence with a common ratio 2. Combining this with Theorem 3, it can be verified that the expected total cost incurred by invoking Algorithm 1 in Algorithm 2 is $O(\mathbb{E}[\sum_{R\in\mathcal{R}}|R|])$. On the other hand, the node selection phase executes Algorithm 1 once on the $\mathcal{R}$ generated by Algorithm 2, which takes $O(\sum_{R\in\mathcal{R}}|R|)$ time. In summary, the expected total cost incurred by Algorithm 1 is $O(\mathbb{E}[\sum_{R\in\mathcal{R}}|R|])$.

Note that $|R| \le w(R)$ for any $R \in \mathcal{R}$. Then, by Equation 12,

$$\begin{aligned} O(\mathbb{E}[\textstyle\sum_{R\in\mathcal{R}}|R|]) &\le O(\mathbb{E}[\textstyle\sum_{R\in\mathcal{R}}w(R)]) \\ &= O\left((k+\ell)(n+m)\log n/\varepsilon^2\right), \end{aligned}$$

which proves the theorem. □

Proof of Theorem 5. The approximation guarantees of *IMM* under $\mathcal{M}$ follow from the discussions in Section 4.1. In the following, we focus on *IMM*'s time complexity. Similar to the case of the triggering models, the computation cost of *IMM* under $\mathcal{M}$ also mainly consists of two parts, namely, the cost of generating the set $\mathcal{R}$ of random RI sets, and the cost of invoking Algorithm 1.

Let $w(R)$ be a random variable that denote the cost of generating a specific RI set R under $\mathcal{M}$. Using a derivation similar to the proof of Equation 12 in Section 3.4, we can prove that

$$\begin{aligned} \mathbb{E}\Big[\textstyle\sum_{R\in\mathcal{R}}w(R)\Big] &= \mathbb{E}\big[|\mathcal{R}|\big]\cdot EPT \\ &= O\left(\tfrac{EPT}{OPT}(k+\ell)(n+m)\log n/\varepsilon^2\right). \end{aligned}$$

In other words, the expected total cost of generating $\mathcal{R}$ is $O\left(\frac{EPT}{OPT}(k+\ell)(n+m)\log n/\varepsilon^2\right)$. To prove the theorem, it suffices to show that the expected total cost incurred by Algorithm 1 is also no more than $O\left(\frac{EPT}{OPT}(k+\ell)(n+m)\log n/\varepsilon^2\right)$.

Following from the analysis in the proof of Theorem 4, the expected total cost incurred by Algorithm 1 is $O(\mathbb{E}[\sum_{R\in\mathcal{R}}|R|])$. Note that $|R| \le w(R)$ for any $R \in \mathcal{R}$. Therefore, we have

$$\begin{aligned} O(\mathbb{E}[\textstyle\sum_{R\in\mathcal{R}}|R|]) &\le O(\mathbb{E}[\textstyle\sum_{R\in\mathcal{R}}w(R)]) \\ &= O\left(\tfrac{EPT}{OPT}(k+\ell)(n+m)\log n/\varepsilon^2\right), \end{aligned}$$

which completes the proof. □

Proof of Lemma 11. First, we show that if $R(v)$ is constructed by the conceptual algorithm in Section 4.2, then the lemma holds. Let p_1 be the probability that S overlaps $R(v)$, and p_2 be the probability that S as a seed set can activate v in the diffusion process under the CTIC model. Observe that the conceptual algorithm first constructs g and then identifies nodes within t distance to v. Hence, p_1 equals the probability that g contains a directed path that starts from a node in S, ends at v, and has a length no more than t. In addition, p_2 equals the probability that v is reachable from S within t distance. Thus, we have $p_1 = p_2$.

Second, we prove that our implementation of the conceptual algorithm does not alter its output. In our implementation, we only sample the length of an edge at the first time that we encounter it. For any edge inspected in the algorithm, the effect of sampling its length on the fly is the same as sampling its length at the very beginning of the algorithm. Therefore, the node set $R(v)$ returned by our implementation is identical to the output of the conceptual algorithm. Thus, the lemma is proved. □

Proof of Lemma 12. Let R be the RI set generated from v by ***RI-Gen***, and $w(R)$ be the number of edges in G that point to the nodes in R. $\mathbb{E}[m^*] \le \mathbb{E}[w(R)]$ holds because every edge inspected by ***RI-Gen*** must point to a node in R. Define a probability distribution $\mathcal{V}'$ over all nodes, such that each node has the probability mass that proportional to its in-degree in G. Let v' be a random sample from $\mathcal{V}'$, and p_R be the probability that an edge selected uniformly at random from G would point to a node in R. By the definition of $w(R)$ and p_R, we have

$$m \cdot \mathbb{E}[p_R] \;=\; \mathbb{E}[p_R \cdot m] \;=\; \mathbb{E}[w(R)].$$

Define a function $f(v', R)$ for each v' sampled from $\mathcal{V}'$, such that $f(v', R) = 1$ if $v' \in R$, and $f(v', R) = 0$ otherwise. Regarding v' as a random variable and R as a fixed set, we have

$$p_R \;=\; \textstyle\sum_{v'} \Pr[v'] \cdot f(v', R).$$

Next, we fix v' and consider R as a random variable. Define $p_{v'}$ to be the probability that R overlaps $\{v'\}$. We have

$$p_{v'} \;=\; \textstyle\sum_{R} Pr[R] \cdot f(v', R).$$

By Lemma 11, we have $p_{v'}$ equals the probability that $\{v'\}$ as a seed set can activate v. Thus, we have

$$n \cdot \mathbb{E}[p'_v] \;=\; \mathbb{E}[I(\{v'\})].$$

Since OPT^* is the largest expected influence of any size-1 node set, $\mathbb{E}[I(\{v'\})] \le OPT^*$ holds. Combining all equations above, we have

$$\begin{aligned} n \cdot \mathbb{E}[m^*] &\le n \cdot \mathbb{E}[w(R)] \\ &= n \cdot m \cdot \mathbb{E}[p_R] \\ &= n \cdot m \cdot \textstyle\sum_R \Pr[R] \cdot p_R \\ &= n \cdot m \cdot \textstyle\sum_R \left(\Pr[R] \cdot \sum_{v'} \Pr[v'] \cdot f(v', R)\right) \\ &= n \cdot m \cdot \textstyle\sum_{v'} \left(\Pr[v'] \cdot \sum_R \Pr[R] \cdot f(v', R)\right) \\ &= n \cdot m \cdot \textstyle\sum_{v'} \Pr[v'] \cdot p_{v'} \\ &= n \cdot m \cdot \mathbb{E}[p_{v'}] \\ &= m \cdot \mathbb{E}[I(\{v'\})] \\ &\le m \cdot OPT^*. \end{aligned}$$

Thus, the lemma is proved. □

Proof of Theorem 6. Let n^* and m^* be the number of nodes and edges inspected by ***RI-Gen*** respectively. Recall that ***RI-Gen*** runs in $O(n^* \log n^* + m^*)$ time. We have $n^* \log n^* + m^* = O(m^* \log n)$. By Lemma 12, we have

$$\mathbb{E}[m^* \log n] \;\le\; \frac{m}{n} \cdot OPT^* \log n.$$

Observe that OPT^* is no more than the largest expected influence of a size-k node set, i.e., $OPT^* \le OPT$ for any $k \ge 1$. Then, by Theorem 5, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$-approximation with at least $1 - 1/n^\ell$ probability under the CTIC model, and runs in $O\left((k+\ell)(n+m)\log^2 n/\varepsilon^2\right)$ expected time. Thus, the theorem is proved. □