Contents lists available at ScienceDirect
Performance Evaluation
journal homepage: www.elsevier.com/locate/peva

Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework

Yishi Lin [*], John C.S. Lui
Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Competitive influence maximization
Information diffusion
Viral marketing
Social networks

ABSTRACT

Given the popularity of the viral marketing campaign in online social networks, finding a computationally efficient method to identify a set of most influential nodes so as to compete well with others is of the utmost importance. In this paper, we propose a general model to describe the influence propagation of multiple competing sources in the same network. We formulate the *Competitive Influence Maximization with Partial information* (CIMP) problem: given an influence propagation model and the probability of a node being in the competitor's seed set, how to find a set of k seeds so to trigger the largest expected influence cascade under the presence of other competitors? We propose a general algorithmic framework, *Two-phase Competitive Influence Maximization* (TCIM), to address the CIMP problem. TCIM returns a $(1 - 1/e - \epsilon)$-approximate solution with probability of at least $1 - n^{-\ell}$, where $\ell \geq 1/2$ is a parameter controlling the trade-off between the success probability and the computational efficiency. TCIM has an efficient expected time complexity of $O(c(k+\ell)(m+n)\log n/\epsilon^2)$, where n and m are the number of nodes and edges in the network, and c is a function of the given propagation model (which may depend on k and the underlying network). To the best of our knowledge, this is the first work which provides a general framework for the competitive influence maximization problem where the seeds of the competitor could be given as an *explicit* set of seed nodes or a *probability distribution* of seed nodes. Moreover, our algorithmic framework provides both quality guarantee of solution and practical computational efficiency. We conduct extensive experiments on real-world datasets under three specific influence propagation models, and show the efficiency and accuracy of our framework. In particular, for the case where the seed set of the competitor is given explicitly, we achieve up to four orders of magnitude speedup as compared to previous algorithms with the same quality guarantee. When the competitor's seed set is not given explicitly, running TCIM using the *probability distribution* of the competitor's seeds returns nodes with higher expected influence than those nodes returned by TCIM using an *explicit guess* of the competitor's seeds.

1. Introduction

With the popularity of online social networks (OSNs), viral marketing has become a powerful method for companies to promote sales. In 2003, Kempe et al. [1] formulated the influence maximization problem: given a network G and an integer

[*] Corresponding author.
E-mail addresses: yslin@cse.cuhk.edu.hk (Y. Lin), cslui@cse.cuhk.edu.hk (J.C.S. Lui).

http://dx.doi.org/10.1016/j.peva.2015.06.012
0166-5316/© 2015 Elsevier B.V. All rights reserved.

k, how to select a set of k nodes in G so that they can trigger the largest influence cascade under a predefined influence propagation model. The selected nodes are often referred to as *seed nodes*. Kempe et al. proposed the *Independent Cascade* (IC) model and the *Linear Threshold* (LT) model to describe the influence propagation process. They proved that the influence maximization problem under these two models is NP-hard and a natural greedy algorithm could return $(1 - 1/e - \epsilon)$-approximate solutions for any $\epsilon > 0$. Recently, Tang et al. [2] presented an algorithm with a high probability of finding an approximate solution, and at the same time, with a low computational overhead.

Recognizing that companies are competing in a viral marketing, a thread of work studied the competitive influence maximization problem under a series of competitive influence propagation models, where multiple sources spread the information in a network simultaneously (e.g., [3–5]). Many of these work assumed that there are two companies competing with each other and studied the problem from the "follower's perspective". For example, in the viral marketing, a company introducing a new product into an existing market can be regarded as the follower, and consumers of the existing products can be treated as nodes influenced by this company's competitor. Formally, the problem of *Competitive Influence Maximization* (CIM) is defined as follows: suppose we are given a network G and the set of *seed nodes* selected by our competitor, how to select *seed nodes* for our product so as to trigger the largest influence cascade? In this work, we propose a more general problem, the *Competitive Influence Maximization with Partial information* (CIMP) problem, where we are given the probability of each node being in the competitor's seed set. We refer to this given probability as the *seed distribution* of the competitor. The assumption of knowing the *seed distribution* in the CIMP problem is much milder than the assumption in the CIM problem where one needs to know the explicit set of seed nodes of the competitor. Note that the CIM problem is essentially a special case of our CIMP problem.

Contributions: We believe that there can be many influence propagation models representing different viral marketing scenarios. However, existing works are usually restricted to specific propagation models (e.g., [4,6]). In this work, we propose a *general framework* that can solve the competitive influence maximization problem under a variety of propagation models. Our contributions are:

- We present a *General Competitive Independent Cascade* (GCIC) model which can accommodate different influence propagation models, and apply it to the *Competitive Influence Maximization* (CIM) problem. We proceed to formulate a more general *Competitive Influence Maximization with Partial information* (CIMP) problem, which generalizes the CIM problem as a special case. Both the CIM and CIMP problems are NP-hard.
- For the CIMP problem under the GCIC model, we propose a *Two-phase Competitive Influence Maximization* (TCIM) algorithmic framework. With probability of at least $1-n^{-\ell}$, TCIM guarantees a $(1-1/e-\epsilon)$-approximate solution. It runs in $O(c(\ell+k)(m+n)\log n/\epsilon^2)$ expected time, where $n = |V|$, $m = |E|$, $\ell \geq 1/2$ is a quality control knob, and c depends on the specific propagation model, seed-set size k and the network $G = (V, E)$. To the best of our knowledge, this is the first general algorithm with *both* $(1 - 1/e - \epsilon)$ approximation guarantee and practical efficiency. Moreover, as we will explain later, the $(1-1/e-\epsilon)$ approximation guarantee is in fact the best guarantee one could obtain in polynomial time.
- To demonstrate the generality, accuracy and efficiency of our framework, we analyze the performance of TCIM using three popular influence propagation models: *Campaign-Oblivious Independent Cascade model* [6], *Distance-based model* [4] and *Wave propagation model* [4].
- We conduct extensive experiments using real datasets to demonstrate the efficiency and effectiveness of TCIM. For the CIM problem, when $k = 50$, $\epsilon = 0.5$ and $\ell = 1$, TCIM returns solutions comparable with those returned by baseline algorithms with the same quality guarantee, but runs *up to four orders of magnitude faster*. We also illustrate via experiments the benefits of taking the "seed distribution" of the competitor as input to our TCIM framework.

The outline of our paper is as follows. Background and related work are given in Section 2. We define the *General Competitive Independent Cascade* model, the *Competitive Influence Maximization* problem and the *Competitive Influence Maximization problem with Partial information* in Section 3. We present the TCIM framework in Section 4 and analyze its performance under various influence propagation models in Section 5. We compare TCIM with the greedy algorithm with performance guarantee in Section 6, and show experimental results in Section 7. Section 8 concludes.

2. Background and related work

Single source influence maximization. In the seminal work [1], Kempe et al. proposed the *Independent Cascade* (IC) and the *Linear-Threshold* (LT) propagation model and formally defined the *influence maximization problem*. In the IC model, a network $G = (V, E)$ is given and each edge $e_{uv} \in E$ is associated with a probability p_{uv}. Initially, a set of nodes S is *active* and S is referred to as the *seed nodes*. Each active node u has a single chance to influence its inactive neighbor v and succeeds with probability p_{uv}. Let $\sigma(S)$ be the expected number of nodes in G that S could activate, the influence maximization problem is defined as how to select a set of k nodes such that $\sigma(S)$ is maximized. Kempe et al. showed that this problem is NP-hard under both proposed models. Moreover, they showed that $\sigma(S)$ is a monotone and submodular function of S under both models. Therefore, for any $\epsilon > 0$, a greedy algorithm returns a solution whose expected influence is at least $(1 - 1/e - \epsilon)$ times the expected influence of the optimal solution. The research on this problem went on for around ten years (e.g., [7–12]). Recently, Borgs et al. [13] made a breakthrough and presented an algorithm that simultaneously maintains the performance guarantee and significantly reduces the time complexity. Tang et al. [2] further improved the method in [13] and presented an algorithm TIM/TIM$^+$, where TIM stands for *Two-phase Influence Maximization*. It returns a $(1 - 1/e - \epsilon)$-approximate solution with probability at least $1 - n^{-\ell}$ and runs in time $O((\ell + k)(m + n) \log n/\epsilon^2)$, where $n = |V|$ and $m = |E|$. Note that the approximation factor $(1 - 1/e - \epsilon)$ is in fact optimal for two reasons. First, the evaluation of $\sigma(S)$ for $S \neq \emptyset$ is #P-hard for both the IC model [8] and the LT model [9]. Second, for any monotone and submodular function $\sigma(S)$, $(1 - 1/e)$ is the optimal approximation guarantee one could obtain by evaluating σ at a polynomial number of set [14].

Competitive influence maximization. We review some work that modeled the competition between two sources and studied the influence maximization problem from the "*follower's perspective*". The majority of these works considered competition between two players (e.g., two companies), and the "follower" is the player who selects a set of *seed nodes* with the full knowledge of seed nodes selected by its competitor. Carnes et al. [4] proposed the *Distance-based model* and the *Wave propagation model* to describe the influence spread of competing sources and considered the influence maximization problem from the follower's perspective. Bharathi et al. [3] proposed an extension of the single source IC model and utilized the greedy algorithm to compute the best response to the competitor. Motivated by the need to limit the spread of rumor in the social networks, there is a thread of work focusing on how to maximize rumor containment (e.g., [15,6,16]). For example, Budak et al. [6] modeled the competition between the "bad" and "good" sources. They focused on minimizing the number of nodes influenced by the "bad" source. The majority of these works assume that seeds of one source, or the "first-mover", is known. This assumption is sometimes too *restrictive* because competitors do not always reveal their seed set. Our work considers that one can estimate the "*distribution*" of the seed set of the first source, this relaxes the above restrictive assumption. But this relaxation also implies that one needs an accurate and efficient algorithm to solve the CIMP, and this is the aim of our work. Note that Budak et al. [6] also considered the influence limitation problem in the presence of missing data. However, both of their assumption and strategy differ from ours. They assumed that only the status (influenced, uninfluenced, newly influenced) of some nodes are revealed. And, their strategy was to predict the status of remaining nodes before running the influence limitation algorithm.

3. Competitive influence maximization problem

In this section, we introduce the *General Competitive Independent Cascade* (GCIC) model which models the influence propagation of competing sources in the same network. Based on the GCIC model, we formally define the *Competitive Influence Maximization problem* and the *Competitive Influence Maximization problem with Partial information*.

3.1. General Competitive Independent Cascade Model

Let us first define the *General Competitive Independent Cascade* (GCIC) model. A network is modeled as a directed graph $G = (V, E)$ with $n = |V|$ nodes and $m = |E|$ edges. Users in the network are modeled as nodes, and directed edges between nodes represent the interaction between users. A node v is a neighbor of node u if there is a directed edge $e_{uv} \in E$. Every edge $e_{uv} \in E$ is associated with a length $d_{uv} > 0$ and a probability p_{uv} denoting the influence node u has on v. We assume $p_{uv} = 0$ for $e_{uv} \notin E$. For simplicity, we assume that the length of all edges is 1. Our algorithm and analysis can be easily extended when edges have non-uniform lengths.

We first consider two influence sources. Denote A and B as two sources that simultaneously spread information in the network G. A node $v \in V$ could be in one of three states: S, I_A and I_B. Nodes in state S, the *susceptible state*, have not been influenced by any source. Nodes in state I_A (resp. I_B) are influenced by source A (resp. B). Once a node is influenced, it cannot change its state. Initially, sources A and B can each specify a set of seed nodes, which we denote as $S_A \subseteq V$ and $S_B \subseteq V$. We refer to nodes in S_A (resp. S_B) as *seeds* or *initial adopters* of source A (resp. B). We also assume $S_A \cap S_B = \emptyset$. As in the single source *Independent Cascade* (IC) model, an influenced node u influences its neighbor v with probability p_{uv}, and we say each directed edge $e_{uv} \in E$ is *active* with probability p_{uv}. We use $E_a \subseteq E$ to denote a set of *active* edges. Let $d_{E_a}(v, u)$ be the shortest distance from v to u through active edges in E_a and assume $d_{E_a}(v, u) = +\infty$ if v cannot reach u through active edges. Moreover, let $d_{E_a}(S_A \cup S_B, u) = \min_{v \in S_A \cup S_B} d_{E_a}(v, u)$ be the shortest distance from the nodes in $S_A \cup S_B$ to node u through active edges in E_a. For a given E_a, we say a node v is the *nearest initial adopter* of u if $v \in S_A \cup S_B$ and $d_{E_a}(v, u) = d_{E_a}(S_A \cup S_B, u)$. In the GCIC model, for a given E_a, a node u will be in the same state as that of one of its *nearest initial adopters* at the end of the influence propagation process. The expected influence of S_B is the expected number of nodes in state I_B at the end of the influence propagation process, where the expectation is taken over the randomness of E_a. A specific influence propagation model of the GCIC model will specify how the influence propagates in detail, including the tie-breaking rule for the case where both nodes in S_A and S_B are the nearest initial adopters of a node.

Moreover, we make the following assumptions. Let $\sigma_u(S_B|S_A)$ be the conditional probability that, given S_A, node u will be influenced by source B when S_B is the seed set for source B. We assume that $\sigma_u(S_B|S_A)$ is a monotone and submodular function of $S_B \subseteq V \setminus S_A$ for all $u \in V$. Let $\sigma(S_B|S_A) = \sum_{u \in V} \sigma_u(S_B|S_A)$ be the expected influence of S_B given S_A, it is also monotone and submodular of $S_B \subseteq V \setminus S_A$. Finally, one can easily extend the above model to more than two sources. This can be done by assuming that source B has n competitors $A_1, \ldots, A_n$. Let $A = A_1 \cup \cdots \cup A_n$, from the perspective of source B, he is facing a single large competitor A.

We call this the *General Competitive Independent Cascade* model because for any graph $G = (V, E)$ and $S_B \subseteq V$, the expected influence of S_B given $S_A = \emptyset$ is equal to the expected influence of S_B in the single source IC model. Note that there are a number of specific instances of the GCIC model, e.g., the *Campaign-Oblivious Independent Cascade Model* [6], the *Wave propagation Model* [4], and the *Distance-based Model* [4]. The influence spread function of these specific instances is in fact monotone and submodular.[1] We will elaborate on these models in later sections.

3.2. Problem definition

We now formally define the *Competitive Influence Maximization* (CIM) problem.

Definition 1 (*Competitive Influence Maximization Problem*). Suppose we are given a specific instance of the *General Competitive Independent Cascade* model (i.e., the Distance-based Model), a directed graph $G = (V, E)$ and the seed set $S_A \subseteq V$ for source A. Find a set S_B^* of k nodes for source B such that the expected influence of S_B^* given S_A is maximized. Formally, we have: $S_B^* = \arg\max_{S_B \in \{S \subseteq V \setminus S_A,\ |S|=k\}} \sigma(S_B|S_A)$.

As stated previously, knowing S_A is not always realistic since competitors are not willing to reveal the information. We now define a more general problem where we are given the *seed distribution* $\mathcal{D}_A$ of source A, instead of the explicit seed set S_A. For each node $u \in V$, $\mathcal{D}_A(u)$ specifies the probability that u is a seed of source A. We write $S_A \sim \mathcal{D}_A$ to indicate that the seed set S_A is drawn from the random seed distribution $\mathcal{D}_A$. Given the set S_B of seed nodes selected by source B, we define the expected influence spread of S_B given the seed distribution $\mathcal{D}_A$ as $\sigma(S_B|\mathcal{D}_A) = \mathbb{E}_{S_A \sim \mathcal{D}_A}[\sigma(S_B \setminus S_A|S_A)]$. Note that by this definition, we are essentially considering the "worst case" influence spread of S_B such that if source B happens to select a node in S_A, source B would fail to influence that node. One could also specify a tie-breaking rule for the situation where both sources A and B select the same node. In this work, for the ease of presentation, we assume *A dominates B* (or if there is a tie, A wins). Recall that for any given S_A, $\sigma(S_B \setminus S_A|S_A)$ is a monotone and submodular function of S_B. One could easily verify that, for any given $\mathcal{D}_A$, $\sigma(S_B|\mathcal{D}_A)$ is also a monotone and submodular function of S_B.

Definition 2 (*Competitive Influence Maximization Problem with Partial Information*). Suppose we are given a specific instance of the General Competitive Independent Cascade model (i.e., the Distance-based Model), a graph $G = (V, E)$ and the seed distribution $\mathcal{D}_A$ for source A. Find a set S_B^* of k nodes for source B such that the expected influence of S_B^* given $\mathcal{D}_A$ is maximized, i.e., $S_B^* = \arg\max_{S_B \in \{S \subseteq V, |S|=k\}} \sigma(S_B|\mathcal{D}_A)$.

For the distribution $\mathcal{D}_A$, we assume that if $\mathcal{D}_A(u) < 1$ holds for a node $u \in V$, the probability $\mathcal{D}_A(u)$ is upper bounded by a constant C $(0 \le C < 1)$. Moreover, we are only interested in the non-trivial case that $|V \setminus S_A| \ge k$ for the CIM problem and $|V| \ge k$ for the CIMP problem. It is easy to see that CIM problem is a special case of the CIMP problem where the probability of each node being a seed of source A is either zero or one.

Note that the IC model is essentially the GCIC model where the competitor does not select seed nodes. Because the influence maximization problem under the IC model is NP-hard [1], both the CIM and the CIMP problem under the GCIC model are also NP-hard. Moreover, because computing the expected influence of a seed set is #P-hard under the IC model [8], computing $\sigma(S_B|\mathcal{D}_A)$ (resp. $\sigma(S_B|S_A)$) exactly under the GCIC model is also #P-hard, and one could only estimate the expected influence in polynomial time. Given the above analysis and the fact that $\sigma(S_B|\mathcal{D}_A)$ (resp. $\sigma(S_B|S_A)$) is monotone and submodular of S_B, $(1 - 1/e - \epsilon)$ is the optimal approximation guarantee one could obtain for the CIMP (resp. CIM) problem in a polynomial time [14]. In this paper, we provide a solution to the general CIMP problem with the $(1 - 1/e - \epsilon)$ approximation guarantee and at the same time, with practical run time complexity.

4. Proposed solution framework to the CIMP problem

In this section, we present our *Two-phase Competitive Influence Maximization* (TCIM) algorithm to solve the CIMP problem. Note that our work is different from [2], which was designed for the single source influence maximization problem. Our work is a general framework for the CIMP problem under different instances of the *General Competitive Independent Cascade* model, while maintaining the $(1 - 1/e - \epsilon)$ approximation guarantee and practical efficiency. We first provide basic definitions and the high level idea of TCIM. Then, we present a detailed description and analysis of two phases of the TCIM algorithm, the *Parameter estimation and refinement* phase, and the *Node selection* phase.

4.1. Basic definitions and high level idea

Motivated by the definition of "RR sets" in [13,2], we first define the *Reverse Accessible Pointed Graph* (RAPG). We then design a *scoring system* such that for a large number of random RAPG instances generated based on a given seed distribution $\mathcal{D}_A$ and a given seed set S_B, the average score of S_B for each RAPG instance is a good approximation of the expected influence of S_B given $\mathcal{D}_A$. Let $d_g(u, v)$ be the shortest distance from u to v in a graph g and assume $d_g(u, v) = +\infty$ if u cannot reach v in g. Let $d_g(S, v)$ be the shortest distance from the nodes in set S to node v through edges in g, and assume $d_g(S, v) = +\infty$ if $S = \emptyset$ or $S \ne \emptyset$ but there are no paths from nodes in S to v. We define the *Reverse Accessible Pointed Graph (RAPG)* and the random RAPG instance as follows.

[1] We like to point out that while there does exist influence spread functions which are non-submodular, it is still an open research question as to how to solve these non-submodular optimization problems in general.

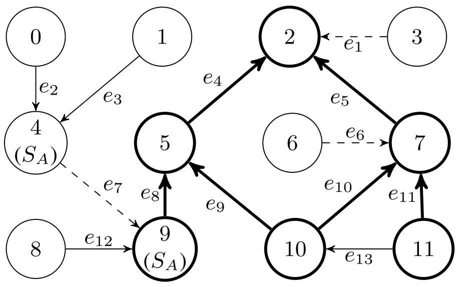


Fig. 1. Example of a random RAPG instance: The graph G contains 12 nodes and 13 directed edges each represented by an arrow. The random subgraph g is obtained from G by removing 3 edges represented by dashed arrows, i.e., e_1, e_6 and e_7. The random seed set of source A is $S_A = \{4, 9\}$. From g, S_A, and the randomly selected "root" node 2, we get the random RAPG instance $R = (V_R, E_R, S_{R,A})$ where $V_R = \{2, 5, 7, 9, 10, 11\}$, $E_R = \{e_4, e_5, e_8, e_9, e_{10}, e_{11}\}$ and $S_{R,A} = \{9\}$.

Definition 3 (*Reverse Accessible Pointed Graph*). For a given node r in G, a given seed set S_A drawn from $\mathcal{D}_A$, and a subgraph g of G obtained by removing each edge e_{uv} in G with probability $1 - p_{uv}$, let $R = (V_R, E_R, S_{R,A})$ be the Reverse Accessible Pointed Graph (RAPG) obtained from r, S_A, and g. The node set V_R contains $u \in V$ if $d_g(u, r) \le d_g(S_A, r)$. The edge set E_R contains edges on all shortest paths from nodes in V_R to r through edges in g. And the node set $S_{R,A} = S_A \cap V_R$ is the set of initial adopters of source A in R. We refer to r as the "root" of R.

Definition 4 (*Random RAPG Instance*). Let $\mathcal{G}$ be the distribution of g induced by the randomness in edge removals from G. A random RAPG instance R is a Reverse Accessible Pointed Graph (RAPG) obtained from a randomly selected node $r \in V$, an instance of g randomly sampled from $\mathcal{G}$, and an instance S_A randomly sampled from $\mathcal{D}_A$.

Fig. 1 shows an example of a random RAPG instance $R = (V_R, E_R, S_{R,A})$ with $V_R = \{2, 5, 7, 9, 10, 11\}$, $E_R = \{e_4, e_5, e_8, e_9, e_{10}, e_{11}\}$ and $S_{R,A} = \{9\}$. The "root" of R is node 2.

Let us now present our *scoring system*. For a random RAPG instance $R = (V_R, E_R, S_{R,A})$ obtained from a random node $r \in V$, $g \sim \mathcal{G}$ and $S_A \sim \mathcal{D}_A$, the score of a node set S_B in R is defined as follows.

Definition 5 (*Score*). Suppose we are given a random RAPG instance $R = (V_R, E_R, S_{R,A})$ obtained from a random node r, $g \sim \mathcal{G}$ and $S_A \sim \mathcal{D}_A$. The score of a node set S_B in R, denoted by $f_R(S_B)$, is defined as the probability that node r will be influenced by source B when (1) the influence propagates in graph g with all edges being "active"; and (2) $S_A \cap V_R$ and $(S_B \setminus S_A) \cap V_R$ are seed sets for sources A and B respectively.

Recall that for the *General Competitive Independent Cascade* model, we assume that for any node $u \in V$, the conditional probability $\sigma_u(S_B|S_A)$ is a monotone and submodular function of $S_B \subseteq V \setminus S_A$. It follows that, for any given $R = (V_R, E_R, S_{R,A})$, $f_R(S_B)$ is also a monotone and submodular function of $S_B \subseteq V$. Furthermore, we define the marginal gain of the score as follows.

Definition 6 (*Marginal Gain of Score*). For a random RAPG instance R with root v, we denote

$$\Delta_R(w|S_B) = f_R(S_B \cup \{w\}) - f_R(S_B) \tag{1}$$

as the marginal gain of score if we add w to the seed set S_B.

From the definition of GCIC model and that of the RAPG, we know that for any RAPG instance R obtained from r, $g \sim \mathcal{G}$ and $S_A \sim \mathcal{D}_A$, R contains all nodes that can influence r and all shortest paths from these nodes to r. Hence, for any given $\mathcal{D}_A$, S_B and node w, once an instance R is constructed, the evaluation of $f_R(S_B)$ and $\Delta_R(w|S_B)$ can be done based on R without the knowledge of g and $S_A \setminus S_{R,A}$, where S_A is drawn from $\mathcal{D}_A$.

From Definition 5, for any $S_B \subseteq V$, the expected value of $f_R(S_B)$ over the randomness of R equals the probability that a randomly selected node in G can be influenced by S_B. Formally, we have the following lemma.

Lemma 1. *Given a seed distribution $\mathcal{D}_A$ and a seed set S_B, we have $\sigma(S_B|\mathcal{D}_A) = n \cdot \mathbb{E}[f_R(S_B)]$, where the expectation of $\mathbb{E}[f_R(S_B)]$ is taken over the randomness of R, and n is the number of nodes in G.*

From Lemma 1, for any given set S_B, a high expected "score" implies a high expected influence spread. By Lemma 1 and the Chernoff–Hoeffding bound, for a sufficiently large number of random RAPG instances, the average score of a set S_B in those RAPG instances could be a good approximation to the expected influence of S_B in G. The main challenge is how to determine the number of RAPG instances required, and how to select seed nodes for source B with a high expected influence spread based on a set of random RAPG instances. TCIM consists of the following two phases, which will be described in detail in later subsections.

1. *Parameter estimation and refinement*: Suppose S_B^* is the optimal solution to the CIMP problem. Let $OPT = \sigma(S_B^*|\mathcal{D}_A)$ be the expected influence spread of S_B^* given $\mathcal{D}_A$. In this phase, TCIM estimates and refines a lower bound of OPT and uses the lower bound to derive a parameter θ.
2. *Node selection*: In this phase, TCIM first generates a set $\mathcal{R}$ of θ random RAPG instances of G and $\mathcal{D}_A$, where θ is a sufficiently large number obtained in the previous phase. Using the greedy approach, TCIM returns a set of seed nodes S_B for source B with the goal of maximizing the summation of "score" over all RAPG instances generated.

4.2. Node selection

We first describe the *Node selection* phase, because we would like to explain why we need to estimate the lower bound of OPT first. Algorithm 1 shows the pseudo-code of the *node selection* phase. Given a graph G, the seed distribution $\mathcal{D}_A$ of source A, the seed set size k for source B and a constant θ, the algorithm returns a seed set S_B of k nodes for source B with a large expected influence spread. In Lines 1–2, the algorithm generates θ random RAPG instances and initializes the marginal gain of score for all nodes $u \in V$. Then, in Lines 3–12, the algorithm selects seed nodes S_B iteratively using the greedy approach with the goal of maximizing the summation of "score", i.e., $\sum_{R\in\mathcal{R}} f_R(S_B)$.

Algorithm 1 NodeSelection $(G, \mathcal{D}_A, k, \theta)$

```
1:  Generate a set $\mathcal{R}$ of $\theta$ random RAPG instances.
2:  Let $MG_{\mathcal{R}}(u) = \sum_{R\in\mathcal{R}} f_R(\{u\})$ for all $u \in V$.
3:  Initialize the seed set $S_B = \emptyset$.
4:  for $i = 1$ to $k$ do
5:      Identity the node $v_i \in V\backslash S_B$ with largest $MG_{\mathcal{R}}(v_i)$.
6:      Add $v_i$ to $S_B$.
7:      if $i < k$ then
8:          // Update $MG_{\mathcal{R}}(u)$ as $\sum_{R\in\mathcal{R}} \Delta_R(u|S_B)$ for all $u \in V\backslash S_B$.
9:          Let $\mathcal{R}' = \{R | R \in \mathcal{R}, \Delta_R(v_i|S_B\backslash\{v_i\}) > 0\}$.
10:         for all $R \in \mathcal{R}'$ and $u \in V_R\backslash(S_B \cup S_{R,A})$ do
11:             $MG_{\mathcal{R}}(u) = MG_{\mathcal{R}}(u) - \Delta_R(u|S_B\backslash\{v_i\})$.
12:             $MG_{\mathcal{R}}(u) = MG_{\mathcal{R}}(u) + \Delta_R(u|S_B)$.
13: return $S_B$
```

Generation of RAPG instances. We adapt the randomized breadth-first search (BFS) [13,2] here. To generate one random RAPG instance, we first randomly pick a node r as its "root". Then, we create a queue containing a single node r and initialize the RAPG instance under construction as $R = (V_R = \{r\}, E_R = \emptyset, S_{R,A} = \emptyset)$. For a node u, let $d_R(u, r)$ be the shortest distance from u to r in the current R and let $d_R(u, r) = +\infty$ if u cannot reach r in R. We iteratively pop the node v at the top of the queue and examine its incoming edges. For each incoming neighbor u of v satisfying $d_R(u, r) \geq d_R(v, r)+1$, with probability p_{uv}, we insert e_{uv} into R and push node u into the queue if it has not been pushed into the queue before. Whenever we push a node u into the queue, including the "root" node r, we add u to $S_{R,A}$ with probability $\mathcal{D}_A(u)$. If set $S_{R,A}$ first becomes non-empty when we push a node u into the queue and $d_R(u, r) = d$, we terminate the BFS after we have examined incoming edges of all nodes whose distance to r in R is d. Otherwise, the BFS terminates naturally when the queue becomes empty. If reverse the direction of all edges in R, we obtain an accessible pointed graph, in which all nodes are reachable from r. For this reason, we refer to r as the "root" of R.

Greedy approach. Let $F_{\mathcal{R}}(S_B) = \sum_{R\in\mathcal{R}} f_R(S_B)$ for all $S_B \subseteq V$. Since $f_R(S_B)$ is a monotone and submodular function of $S_B \subseteq V$ for any RAPG instance R, we can conclude that $F_{\mathcal{R}}(S_B)$ is also a monotone and submodular function of $S_B \subseteq V$ for any $\mathcal{R}$. Hence, the greedy approach selecting a set of nodes S_B with the goal of maximizing $F_{\mathcal{R}}(S_B)$ could return a $(1 - 1/e)$ approximate solution [17]. Formally, let S_B^* be the optimal solution. The greedy approach in Lines 3–12 of Algorithm 1 returns a solution S_B such that $F_{\mathcal{R}}(S_B) \geq (1 - 1/e)F_{\mathcal{R}}(S_B^*)$. From Lemma 1, $F_{\mathcal{R}}(S_B)$ being large implies that S_B has a large expected influence spread when used as a seed set for source B.

The "marginal gain vector". During the greedy selection process, we maintain a vector $MG_{\mathcal{R}}$ such that $MG_{\mathcal{R}}(u) = F_{\mathcal{R}}(S_B \cup \{u\}) - F_{\mathcal{R}}(S_B)$ holds for current S_B and all $u \in V \setminus S_B$. We refer to $MG_{\mathcal{R}}$ as the "*marginal gain vector*". The initialization of $MG_{\mathcal{R}}$ could be done during or after the generation of random RAPG instances, whichever is more efficient. At the end of each iteration of the greedy approach, we update $MG_{\mathcal{R}}$. Suppose in one iteration, we expand the previous seed set S_B' by adding a node v_i and the new seed set is $S_B = S_B' \cup \{v_i\}$. For any RAPG instance R such that $v_i \notin V_R \setminus S_{R,A}$, we have $\Delta_R(u|S_B') = \Delta_R(u|S_B)$ for all $u \in V \setminus S_B$. And, for any RAPG instance R such that $f_R(S_B') = 1$, for all $u \in V \setminus S_B'$, we have $\Delta_R(u|S_B') = 0$ and the marginal gain of score cannot be further decreased. To conclude, for a given $R = (V_R, E_R, S_{R,A})$ and a node $u \in V_R \setminus (S_{R,A} \cup S_B)$, $\Delta_R(u|S_B)$ differs from $\Delta_R(u|S_B')$ only if node $v_i \in V_R \setminus S_{R,A}$ and $f_R(S_B') < 1$. Hence, to update $MG_{\mathcal{R}}(u)$ as $\sum_{R\in\mathcal{R}} \Delta_R(u|S_B)$ for all $u \in V \setminus S_B$, it is not necessary to compute $\Delta_R(u|S_B)$ for all $R \in \mathcal{R}$ and $u \in V \setminus (S_{R,A} \cup S_B)$. Note that for any RAPG instance R and node v_i, $\Delta_R(v_i|S_B \setminus \{v_i\}) > 0$ implies $v_i \in V_R \setminus S_{R,A}$ and $f_R(S_B') > 1$. Hence, Lines 9–12 do the update correctly.

Time complexity analysis. Let $\mathbb{E}[N_R]$ be the expected number of random numbers required to generate a random RAPG instance, the expected time complexity of generating θ random RAPG instances is $O(\theta \cdot \mathbb{E}[N_R])$. Let $\mathbb{E}[|E_R|]$ be the expected number of edges in a random RAPG instance. We assume that the initialization and update of $MG_{\mathcal{R}}$ takes time $O(c\theta\cdot\mathbb{E}[|E_R|])$. Here, $c = \Omega(1)$ depends on specific influence propagation model and may also depend on k and G. In each iteration, we select a node from $V\setminus S_B$ with the largest marginal gain of score, which takes time $O(n)$. Hence, Algorithm 1 runs in $O(kn+\theta\cdot\mathbb{E}[N_R]+c\theta\cdot\mathbb{E}[|E_R|])$ expected time. Moreover, from the fact that $\mathbb{E}[|E_R|]\le\mathbb{E}[N_R]$ and $c=\Omega(1)$, the expected running time can be written in a more compact form as

$$O(kn+c\theta\cdot\mathbb{E}[N_R]). \tag{2}$$

In Section 5, we will show the value of c and provide the expected running time of the TCIM algorithm for several influence propagation models, e.g., the *Distance-based model*.

The approximation guarantee. From Lemma 1, we see that the larger θ is, the more accurate is the estimation of the expected influence. The key challenge is how to determine the value of θ, i.e., the number of RAPG instances required, so as to achieve certain accuracy of the estimation. Specifically, we like to find a θ such that the node selection algorithm returns a $(1-1/e-\epsilon)$-approximation solution. At the same time, we want θ to be as small as possible because it has a direct impact on the running time of Algorithm 1. Applying the Chernoff–Hoeffding bound, one can show that for a sufficiently large set $\mathcal{R}$ of random RAPG instances, $n\cdot F_{\mathcal{R}}(S_B)/\theta = n\cdot\left(\sum_{R\in\mathcal{R}}f_R(S_B)\right)/\theta$ could be an accurate estimate of the influence spread of S_B given $\mathcal{D}_A$. Then, the set S_B of k nodes returned by the greedy algorithm has a large $F_{\mathcal{R}}(S_B)$ and also a large expected influence spread. Specifically, for Algorithm 1, we have the following theorem.

Theorem 1. *Given that θ satisfies*

$$\theta\ge(8+2\epsilon)n\cdot\frac{\ell\ln n+\ln\binom{n}{k}+\ln 2}{OPT\cdot\epsilon^2}. \tag{3}$$

Algorithm 1 returns a solution with $(1-1/e-\epsilon)$ approximation with probability at least $1-n^{-\ell}$.

Proof. Please refer to Appendix. □

By Theorem 1, let $\lambda=(8+2\epsilon)n(\ell\ln n+\ln\binom{n}{k}+\ln 2)/\epsilon^2$, Algorithm 1 returns a $(1-1/e-\epsilon)$-approximate solution for any $\theta\ge\lambda/OPT$.

4.3. Parameter estimation

The goal of our parameter estimation algorithm is to find a lower bound LB_e of OPT so that $\theta=\lambda/LB_e\ge\lambda/OPT$. Here, the subscript "e" of LB_e is short for "estimated".

Lower bound of ***OPT***. We define a probability distribution $\mathcal{V}^+$ over the nodes in V. Recall that $\mathcal{D}_A(u)$ is the probability of node u being a seed of source A. And, we denote the indegree of u as $d^-(u)$. In the distribution $\mathcal{V}^+$, the probability mass for each node u is proportional to $(1-\mathcal{D}_A(u))d^-(u)$. The intuition is that we are trying to select nodes with high indegree but have a low probability of being a seed of A. Suppose we take k samples from $\mathcal{V}^+$ and use them to form a node set S_B^+ with duplicated nodes eliminated. A natural lower bound of OPT is the expected influence of S_B^+ given the seed distribution $\mathcal{D}_A$ of source A, i.e., $\sigma(S_B^+|\mathcal{D}_A)$. Furthermore, a lower bound of $\sigma(S_B^+|\mathcal{D}_A)$ is also a lower bound of OPT. In the following lemma, we present a lower bound of $\sigma(S_B^+|\mathcal{D}_A)$.

Lemma 2. *Let R be a random RAPG instance. And, let $V_R'=\{u|u\in V_R, f_R(\{u\})=1\}$. We define the width of R, denoted by $w(R)$, as $w(R)=\sum_{u\in V_R'}(1-\mathcal{D}(u))\,d^-(u)$. We define $m'=\sum_{u\in V}(1-\mathcal{D}(u))\,d^-(u)$ and $\alpha(R)=1-\left(1-\frac{w(R)}{m'}\right)^k$. We have $n\cdot\mathbb{E}[\alpha(R)]\le\sigma(S_B^+|S_A)$, where the expectation of $\mathbb{E}[\alpha(R)]$ is taken over the randomness of R.*

Proof. Please refer to Appendix. □

Let $LB_e := n\cdot\mathbb{E}[\alpha(R)]$. Then, Lemma 2 shows that LB_e is a lower bound of OPT.

Estimation of the lower bound. By Lemma 2, we can estimate LB_e by first measuring $n\cdot\alpha(R)$ on a number of random RAPG instances and then take the average of the estimation. By Chernoff–Hoeffding bound, to obtain an estimation of LB_e within $\delta\in[0,1]$ relative error with probability at least $1-n^{-\ell}$, the number of measurements required is $\Omega(n\ell\log n\epsilon^{-2}/LB_e)$. The difficulty is that we usually have no prior knowledge about LB_e. Tang et al. [2] provided an *adaptive sampling approach* which dynamically adjusts the number of measurements based on the observed sample value. We apply Tang et al.'s approach directly and Algorithm 3 shows the pseudo-code that estimates LB_e.

For Algorithm 2, the theoretical analysis in [2] can be applied directly and the following theorem holds.

Theorem 2. *When $n\ge 2$ and $\ell\ge 1/2$, Algorithm 2 returns $LB_e^*\in[LB_e/4, OPT]$ with at least $1-n^{-\ell}$ probability, and has expected running time $O(\ell(m+n)\log n)$. Furthermore, $\mathbb{E}[1/LB_e^*]<12/LB_e$.*

Algorithm 2 EstimateLB(G, $\mathcal{D}_A$, ℓ) [20]

```
1: for $i = 1$ to $\log_2 n - 1$ do
2:     Let $c_i = (6\ell \ln n + 6\ln(\log_2 n)) \cdot 2^i$.
3:     Let $s_i = 0$.
4:     for $j = 1$ to $c_i$ do
5:         Generate a random RAPG instance $R$ and calculate $\alpha(R)$.
6:         Update $s_i = s_i + \alpha(R)$.
7:     if $s_i > c_i/2^i$ then
8:         return $LB^*_e = n \cdot s_i/(2 \cdot c_i)$.
9: return $LB^*_e = 1$.
```

Running time of the node selection phase. We now analyze Algorithm 1 assuming $\theta = \lambda/LB^*_e$. From $\theta \geq \lambda/OPT$ and Theorem 1, we know Algorithm 1 returns a $(1 - 1/e - \epsilon)$-approximate solution with high probability. Now we analyze the running time of Algorithm 1. The running time of building θ random RAPG instances is $O(\theta \cdot \mathbb{E}[N_R]) = O(\frac{\lambda}{LB^*_e} \cdot \mathbb{E}[N_R])$ where $\mathbb{E}[N_R]$ is the expected number of random numbers generated for building a random RAPG instance. The following lemma shows the relationship between LB^*_e and $\mathbb{E}[N_R]$.

Lemma 3. *For $\mathbb{E}[N_R]$ and LB^*_e, we have $O\left(\frac{\mathbb{E}[N_R]}{LB^*_e}\right) = O\left(1 + \frac{m}{n}\right)$.*

Proof. Please refer to Appendix. □

Recall that the greedy selection process in Algorithm 1 has expected time complexity $O(kn + c\theta \cdot \mathbb{E}[N_R])$. Let $\theta = \lambda/LB^*_e$. Applying Lemma 3, the expected running time of Algorithm 1 becomes $O\big(kn + c\lambda\mathbb{E}[N_R]/LB^*_e\big) = O\left(c(\ell + k)(m + n)\log n/\epsilon^2\right)$.

4.4. Parameter refinement

As discussed before, if the lower bound of OPT is tight, our algorithm will have a small running time. The current lower bound LB_e is no greater than the expected influence spread of a set of k independent samples from $\mathcal{V}^+$, with duplicates eliminated. Hence, LB_e is often much smaller than the OPT. To narrow the gaps between OPT and the lower bound we get in Algorithm 2, we use a greedy algorithm to find a seed set S'_B based on the limited number of RAPG instances we generated in Algorithm 2, and estimate the influence of S'_B with a reasonable accuracy. Then, the intuition is that we can use a creditable lower bound of $\sigma(S'_B|\mathcal{D}_A)$ or LB^*_e, whichever is larger, as the *refined* bound.

Algorithm 3 refines the lower bound. Lines 2–8 greedily find a seed set S'_B based on the RAPG instances generated in Algorithm 2. Intuitively, S'_B should have a large influence spread when used as seed set for source B. Lines 9–13 estimate the expected influence of S'_B. By Lemma 1, let $\mathcal{R}''$ be a set of RAPG instances, $F := n(\sum_{R\in\mathcal{R}''} f_R(S'_B))/|\mathcal{R}''|$ is an unbiased estimation of $\sigma(S'_B|\mathcal{D}_A)$. Algorithm 3 generates a set $\mathcal{R}''$ of sufficiently large number of RAPG instances such that $F \leq (1+\epsilon')\sigma(S'_B|\mathcal{D}_A)$ holds with high probability. Then, with high probability, we have $F/(1+\epsilon') \leq OPT$, meaning that $F/(1+\epsilon')$ is a creditable lower bound of OPT. We use $LB_r = \max\{F/(1+\epsilon'), LB^*_e\}$ as the refined lower bound of OPT, which will be used to derive θ in Algorithm 1. The subscript "r" of LB_r stands for "refined".

Algorithm 3 RefineLB(G, k, $\mathcal{D}_A$, LB^*_e, ϵ, ℓ)

```
1: Let $\mathcal{R}'$ be the set of RAPG instances generated in Algorithm 2.
2: Let $MG_{\mathcal{R}'}(u) = \sum_{R\in\mathcal{R}'} f_R(\{u\})$ for all $u \in V$.
3: Initialize the seed set $S'_B = \emptyset$.
4: for $i = 1$ to $k$ do
5:     Identity the node $v_i \in V\backslash S_B$ with largest $MG_{\mathcal{R}'}(v_i)$.
6:     Add $v_i$ to $S'_B$.
7:     if $i < k$ then
8:         Update $MG_{\mathcal{R}'}(u)$ as $\sum_{R\in\mathcal{R}} \Delta_R(u|S_B)$ for all $u \in V\backslash S_B$.
9: $\epsilon' = 5 \cdot \sqrt[3]{\ell \cdot \epsilon^2/(\ell + k)}$
10: $\lambda' = (2 + \epsilon')\ell n \ln n/\epsilon'^2$
11: $\theta' = \lambda'/LB^*_e$
12: Generate a set $\mathcal{R}''$ of $\theta'$ random RAPG instances.
13: Let $F = n \cdot \left(\sum_{R\in\mathcal{R}''} f_R(S'_B)\right)/\theta'$.
14: return $LB_r = \max\{F/(1+\epsilon'), LB^*_e\}$
```

Theoretical analysis. The following lemma shows that Algorithm 3 returns $LB_r \in [LB^*_e, OPT]$ with high probability.

Lemma 4. *Given $LB^*_e \in [LB_e/4, OPT]$, Algorithm 3 returns $LB_r \in [LB^*_e, OPT]$ with at least $1 - n^{-\ell}$ probability.*

Proof. Please refer to Appendix. □

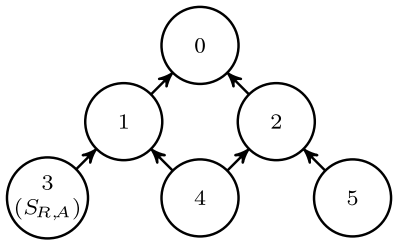


Fig. 2. Example of a random RAPG instance with 6 nodes and 6 edges. The set $S_{R,A}$ contains a single node 3.

In the following theorem, we formally show the performance guarantee and the time complexity of Algorithm 3.

Theorem 3. *Given $\mathbb{E}[1/LB_e^*] = O(12/LB_e)$ and $LB_e^* \in [LB_e/4, OPT]$, Algorithm 3 returns $LB_r \in [LB_e^*, OPT]$ with at least $1 - n^{-\ell}$ probability and runs in $O(c(m+n)(\ell+k)\log n/\epsilon^2)$ expected time.*

Proof. Please refer to Appendix. □

4.5. TCIM: the full algorithm

We now put Algorithms 1–3 together and present the complete TCIM algorithm. Given a network G, the seed distribution $\mathcal{D}_A$ for the source A together with parametric values k, ℓ and ϵ, TCIM returns a $(1-1/e-\epsilon)$ solution with probability at least $1-n^{-\ell}$. First, Algorithm 2 returns the estimated lower bound of OPT, denoted by LB_e^*. Then, we feed LB_e^* to Algorithm 3 and get a refined lower bound LB_r. Finally, $\theta = \lambda/LB_r$ and Algorithm 1 returns a set S_B of k seeds for source B based on $\theta = \lambda/LB_r$ random RAPG instances. Algorithm 4 describes the pseudo-code of TCIM as a whole.

Algorithm 4 TCIM $(G, \mathcal{D}_A, k, \ell, \epsilon)$

```
1: $\ell' = \ell + \ln 3/\ln n$
2: $LB_e^* =$ EstimateLB$(G, \mathcal{D}_A, \ell')$
3: $LB_r =$ RefineLB$(G, k, \mathcal{D}_A, LB_e^*, \epsilon, \ell')$
4: $\lambda = (8+2\epsilon)n\left(\ell'\ln n + \ln\binom{n}{k} + \ln 2\right)/\epsilon^2$
5: $\theta = \lambda/LB_r$
6: $S_B =$ NodeSelection$(G, \mathcal{D}_A, k, \theta)$
7: return $S_B$
```

The following theorem states the solution quality and expected time complexity of the TCIM framework.

Theorem 4 (*TCIM*). *When $n \geq 2$ and $\ell \geq 1/2$, TCIM returns $(1-1/e-\epsilon)$-approximate solution with probability at least $1-n^{-\ell}$. The expected time complexity is $O(c(\ell+k)(m+n)\log n/\epsilon^2)$.*

Proof. We use $\ell' = \ell + \ln 3/\ln n$ as the input parameter value of ℓ for Algorithms 1–3 . By setting this, Algorithms 1–3 each fails with probability at most $n^{-\ell}/3$. Hence, by union bound, TCIM succeeds in returning a $(1-1/e-\epsilon)$ approximation solution with probability at least $1-n^{-\ell}$. Moreover, the expected running time of TCIM is $O(c(\ell+k)(m+n)\log n/\epsilon^2)$, because Algorithms 1–3 each has expected running time at most $O(c(\ell+k)(m+n)\log n/\epsilon^2)$. □

5. Applying and analyzing various propagation models under GCIC

In this section, we describe some special cases of the GCIC model and provide detailed analysis about TCIM for these models. To illustrate the generality of the GCIC model, we use the following propagation models: the *Campaign-Oblivious Independent Cascade Model* in [6], the *Distance-based model* and the *Wave propagation model* in [4] as specific propagation models. For each model, we first briefly describe how the influence propagates given explicit seed sets of both sources. Then, we give examples of the score in a simple RAPG instance shown in Fig. 2, and analyze the time complexity of the TCIM algorithm for the *Competitive Influence Maximization problem with Partial information*.

5.1. Campaign-Oblivious Independent Cascade model

Budak et al. [6] introduced the *Campaign-Oblivious Independent Cascade model* (COICM) extending the single source IC model. The influence propagation process starts with two sets of active nodes S_A and S_B, and then unfolds in discrete steps. At step 0, nodes in S_A (resp. S_B) are activated and are in the state I_A (resp. I_B). When a node u first becomes activated in step t, it gets a single chance to activate each of its currently uninfluenced neighbor v and succeeds with the probability p_{uv}. Budak et al. assumed that one source is prioritized over the other one in the propagation process, and nodes influence by the dominant source always attempt to influence its uninfluenced neighbors first. Here we assume that if there are two or more nodes trying to activate a node v at a given time step, nodes in the state I_B (i.e., nodes influenced by source B) attempt first, which means source B is prioritized over source A.

Examples of score. Suppose we are given seed sets S_A and S_B and a set of active edges E_a. In COICM, source B ends up influencing a node u if and only if $d_u(S_B, E_a) \le d_u(S_A, E_a)$. For the RAPG instance R in Fig. 2 where node 3 is a seed of source A, we have $f_R(S_B) = 1$ if $S_B \cap \{0, 1, 2, 4, 5\} \neq \emptyset$ and $f_R(S_B) = 0$ otherwise.

Analysis of TCIM algorithm. Recall that while analyzing the running time of TCIM, we assume that if $|\mathcal{R}| = \theta$, the expected time complexity for the initialization and update of the "marginal gain vector" $MG_{\mathcal{R}}$ is $O(c\theta \cdot \mathbb{E}[|E_R|])$. We now show that $c = O(1)$ for COICM. Suppose we are selecting nodes based on a set $\mathcal{R}$ of θ RAPG instances. The initialization of $MG_{\mathcal{R}}$ takes time $O(\theta \cdot |E_R|)$ for any RAPG instance $R = (V_R, E_R, S_{R,A})$, we have $f_R(\{u\}) = 1$ for all $u \in V_R \setminus S_{R,A}$ and $f_R(\{u\}) = 0$ otherwise. Suppose in one iteration, we add a node v_i to the set S'_B and obtain a new seed set $S_B = S'_B \cup \{v_i\}$. Recall that we define $\mathcal{R}' = \{R | R \in \mathcal{R}, \Delta_R(v_i|S'_B) > 0\}$ in the greedy approach. For every RAPG instance $R \in \mathcal{R}'$ and for all $u \in V \setminus (S_{R,A} \cup S_B)$, we would have $\Delta_R(u|S_B) = 0$ and $\Delta_R(u|S'_B) = 1$ and hence we need to update $MG_{\mathcal{R}}(u)$ correspondingly. For each RAPG instance R, it only appears in $\mathcal{R}'$ in at most one iteration. Hence, the expected total time complexity for the initialization and update of the "marginal gain vector" is $O(\theta \cdot \mathbb{E}[|E_R|])$. It then follows that the expected running time of TCIM is $O((\ell + k)(m + n) \log n/\epsilon^2)$.

5.2. Distance-based model

Carnes et al. proposed the *Distance-based model* [4]. The idea is that a consumer is more likely to be influenced by the early adopters if their distance in the network is small. The model governs the diffusion of sources A and B given the initial adopters for each source and a set $E_a \subseteq E$ of active edges. Let $d_u(E_a, S_A \cup S_B)$ be the shortest distance from u to $S_A \cup S_B$ along edges in E_a and let $d_u(E_a, S_A \cup S_B) = +\infty$ if there u cannot reach any node in $S_A \cup S_B$. For set $S \subseteq V$, we define $h_u(S, d_u(E_a, S_A \cup S_B))$ as the number of nodes in S at distance $d_u(E_a, S_A \cup S_B)$ from u along edges in E_a. Given S_A, S_B and a set of active edges E_a, the probability that node u will be influenced by source B is $\frac{h_u(S_B, d_u(E_a, S_A \cup S_B))}{h_u(S_A \cup S_B, d_u(E_a, S_A \cup S_B))}$. Thus, the expected influence of S_B is $\sigma(S_B|S_A) = \mathbb{E}\left[\sum_{u \in V} \frac{h_u(S_B, d_u(E_a, S_A \cup S_B))}{h_u(S_A \cup S_B, d_u(E_a, S_A \cup S_B))}\right]$, where the expectation is taken over the randomness of E_a.

Examples of the score. Suppose we are given a random RAPG instance R shown in Fig. 2. If $S_B \cap \{0, 1, 2\} \neq \emptyset$, we would have $f_R(S_B) = 1$. Suppose $S_B = \{4, 5\}$, we have $d_0(E_R, S_{R,A} \cup S_B) = 2$, $h_0(S_B, 2) = 2$ and $h_0(S_{R,A} \cup S_B, 2) = 3$. Hence, node 0 will be influenced by S_B with probability $\frac{2}{3}$, and $f_R(\{4, 5\}) = \frac{2}{3}$. For $S_B = \{4\}$ or $S_B = \{5\}$, one can verify that $f_R(S_B) = \frac{1}{2}$.

Analysis of TCIM algorithm. We now show that $c = O(k)$ for the *Distance-based Model*. In the implementation of TCIM under the *Distance-based Model*, for each RAPG instance $R = (V_R, E_R, S_{R,A})$ with "root" r, we keep $d_R(v, r)$ for all $v \in V_R$ and $d_R(S_{R,A}, r)$ in memory. Moreover, we keep track of the values $h_r(S_{R,A} \cup S_B, d_R(S_{R,A}, r))$ and $h_r(S_B, d_R(S_{R,A}, r))$ for the current S_B and store them in memory. Then, for any given RAPG instance $R = (V_R, E_R, S_{R,A})$ and a node $u \in V_R \setminus (S_{R,A} \cup S_B)$, if $d_R(u, r) = d_R(S_{R,A}, r)$, we have $f_R(S_B \cup \{u\}) = \frac{h_r(S_B, d_R(S_{R,A}, r)) + 1}{h_r(S_{R,A} \cup S_B, d_R(S_{R,A}, r)) + 1}$. Otherwise, we have $f_R(S_B \cup \{u\}) = 1$. In each iteration, for each RAPG instance R, the update of $h_r(S_{R,A} \cup S_B, d_R(S_{R,A}, r))$ and $h_r(S_B, d_R(S_{R,A}, r))$ after expanding previous seed set S_B by adding a node could be done in $O(1)$. Moreover, for any R and $u \in V_R \setminus (S_{R,A} \cup S_B)$, we could evaluate $\Delta_R(u|S_B)$ in $O(1)$. There are $O(\theta)$ RAPG instances with the expected total number of nodes being $O(\theta \cdot \mathbb{E}[|E_R|])$. Hence, in k iterations, the expected total time complexity of the initialization and update of the marginal gain vector is $O(k\theta \cdot \mathbb{E}[|E_R|])$. Substituting c with $O(k)$ in $O(c(\ell + k)(m + n) \log n/\epsilon^2)$, the running time of the TCIM algorithm is $O(k(\ell + k)(m + n) \log n/\epsilon^2)$.

5.3. Wave propagation model

Carnes et al. also proposed the *Wave Propagation* model in [4]. Suppose we are given S_A, S_B and a set of active edges E_a. Let $p(u|S_A, S_B, E_a)$ be the probability that source B influences node u. We also let $d_{E_a}(S_A \cup S_B, u)$ be the shortest distance from seed nodes to u through edges in E_a. Let N_u be the set of neighbors of u whose shortest distance from seed nodes through edges in E_a is $d_{E_a}(S_A \cup S_B, u) - 1$. Then, Carnes et al. [4] defined $p(u|S_A, S_B, E_a) = \left(\sum_{v \in N_u} p(v|S_A, S_B, E_a)\right) / |N_u|$. The expected number of nodes S_B can influence given S_A is $\sigma(S_B|S_A) = \mathbb{E}\left[\sum_{v \in V} p(v|S_A, S_B, E_a)\right]$, where the expectation is taken over the randomness of E_a.

Examples of score. For a random RAPG instance R shown in Fig. 2, as for the Distance-based Model, we have $f_R(S_B) = 1$ if $S_B \cap \{0, 1, 2\} \neq \emptyset$. Suppose $S_B = \{4\}$, source B would influence nodes 4 and 2 with probability 1, influence node 1 with probability 1/2 and influence node 0 with probability 3/4. Hence, $f_R(\{4\}) = 3/4$. Suppose $S_B = \{5\}$, source B would influence nodes 5 and 2 with probability 1, influence node 0 with probability 1/2. Hence, $f_R(\{5\}) = 1/2$. Moreover, one can verify that $f_R(\{4, 5\}) = 3/4$.

Analysis of TCIM algorithm. We now show that for a greedy approach based on a set of θ random RAPG instances, the expected time complexity for the initialization and update of the "marginal gain vector" is $O(kn \cdot \theta \cdot \mathbb{E}[|E_R|])$. In each iteration of the greedy approach, for each RAPG instance $R = (V_R, E_R, S_{R,A})$ and each node $u \in V_R \setminus (S_{R,A} \cup S_B)$, it takes $O(|E_R|)$ to update the marginal gain vector. Since there are θ RAPG instances each having at most n nodes and the greedy approach runs in k

iteration, it takes at most $O(kn \cdot \theta \cdot \mathbb{E}[|E_R|])$ in total to initialize and update the marginal gain vector. Substituting $c = O(kn)$ into $O(c(\ell + k)n(m + n) \log n/\epsilon^2)$, the expected running time of TCIM is $O(k(\ell + k)n(m + n) \log n/\epsilon^2)$.

6. Comparison with the greedy algorithm

In this section, we compare TCIM to the greedy approach with Monte-Carlo method and we consider the *Competitive Influence Maximization* (CIM) problem where the seed set S_A for source A is given. We denote the greedy algorithm as *GreedyMC*, and it works as follows. The seed set S_B is empty initially and the greedy selection approach runs in k iterations. In the ith iteration, *GreedyMC* identifies a node $v_i \in V \setminus (S_A \cup S_B)$ that maximizes the marginal gain of influence spread of source B, i.e., maximizes $\sigma(S_B \cup \{v_i\}|S_A) - \sigma(S_B|S_A)$, and puts it into S_B. Every estimation of the marginal gain is done by θ Monte-Carlo simulations. Hence, *GreedyMC* runs in $O(kmn\theta)$ time. Tang et al. [2] provided the lower bound of θ that ensures the $(1 - 1/e - \epsilon)$ approximation ratio of the greedy method for single source influence maximization problem. We extend their analysis on *GreedyMC* and give the following theorem.

Theorem 5. *For the Competitive Influence Maximization problem, GreedyMC returns a $(1 - 1/e - \epsilon)$-approximate solution with at least $1 - n^{-\ell}$ probability, if*

$$\theta \geq (8k^2 + 2k\epsilon) \cdot n \cdot \frac{(\ell + 1) \ln n + \ln k}{\epsilon^2 \cdot OPT}. \tag{4}$$

Proof. Please refer to Appendix. □

Remark. Now we compare TCIM with *GreedyMC*. For the simplicity of comparison, we assume that $\ell \geq 1/2$ is a constant shared by both algorithms, and $n/2 \leq m$. We believe these assumptions are realistic in practice. Suppose we are able to set θ to the smallest value satisfying Inequality (4), the time complexity of *GreedyMC* is $O(k^3n^2m \log n \cdot \epsilon^{-2}/OPT)$. Given that $OPT \leq n$, the time complexity of *GreedyMC* is at least $O(k^3nm \log n/\epsilon^2)$. Under the assumption that ℓ is a constant and $O(n + m) = O(m)$, TCIM runs in $O(c(\ell + k)(m + n) \log n/\epsilon^2) = O(ckm \log n/\epsilon^2)$. Therefore, for the case where there exist two non-negative constants δ_1 and δ_2 so that $\delta_1 + \delta_2 > 0$ and $c = O(k^{2-\delta_1}n^{1-\delta_2})$, TCIM is more efficient than *GreedyMC*. Note that for all three models described in Section 5, TCIM is more efficient than GreedyMC. For the case where $c = O(k^2n)$, TCIM may still be a better choice than *GreedyMC*. For example, the time complexity of *GreedyMC* depends on the value of *OPT* or a lower bound of *OPT*. However, in practice, we usually have no prior knowledge about *OPT*. Suppose we are not able to get a tight lower bound of *OPT*, or the lower bound we get is much smaller than n, TCIM may outperform *GreedyMC*, because the time complexity of TCIM is independent of the value of *OPT*.

7. Experimental results

We perform experiments on real datasets. First, we consider the *Competitive Influence Maximization* (CIM) problem where the seeds of the competitive source are explicitly given, and we demonstrate the effectiveness and efficiency of our TCIM framework. Then, we present results on the general *Competitive Influence Maximization problem with Partial information* (CIMP) to show the benefit of taking seed distributions of the competitor as input for the TCIM framework.

Datasets. Our datasets contain three real networks: (i) A *Facebook-like social network* containing 1899 users and 20,296 directed edges [18]. (ii) The *NetHEPT network*, an academic collaboration network including 15,233 nodes and 58,891 undirected edges [10]. (iii) An *Epinions social network* of the who-trusts-whom relationships from the consumer review site Epinions [19]. The network contains 508,837 directed "trust" relationships among 75,879 users. As the *weighted IC* model in [1], for each edge $e_{uv} \in E$, we set $p_{uv} = 1/d_v^-$ where d_v^- is the indegree of v.

Propagation models. For each dataset, we use the following propagation models: the *Campaign-Oblivious Independent Cascade Model* (COICM), the *Distance-based model* and the *Wave propagation model* as described in Section 5.

7.1. Effectiveness and efficiency

In this subsection, we demonstrate the effectiveness and efficiency of the TCIM framework. We only report results for the CIM problem for two reasons. First, for all propagation models we tested, the TCIM framework solves the CIM problem exactly the same way as it solves the CIMP problem. Second, we compare TCIM with classical greedy methods and a heuristic algorithm. The heuristic algorithm cannot be easily extended to solve the CIMP problem.

Baselines. We compare TCIM with the following three algorithms. CELF [7] is a greedy approach based on a "lazy-forward" optimization technique. It exploits the monotone and submodularity of the object function to accelerate the algorithm. CELF++ [11] is a variation of CELF which further exploits the submodularity of the influence propagation models. It avoids some unnecessary re-computations of marginal gains in future iterations at the cost of introducing more computation for each candidate seed set considered in the current iteration. SingleDiscount [10] is a degree discount heuristic initially proposed for single source influence maximization problem. For the CIM problem, we adapt this heuristic method and select

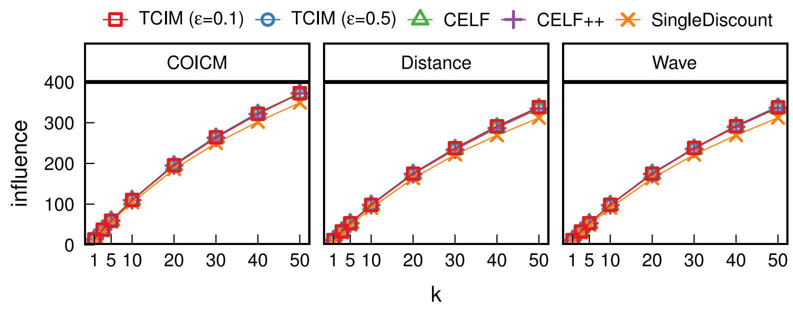


Fig. 3. Results on the *Facebook-like network*: Influence vs. k under three propagation models. ($|S_A| = 50$, $\ell = 1$).

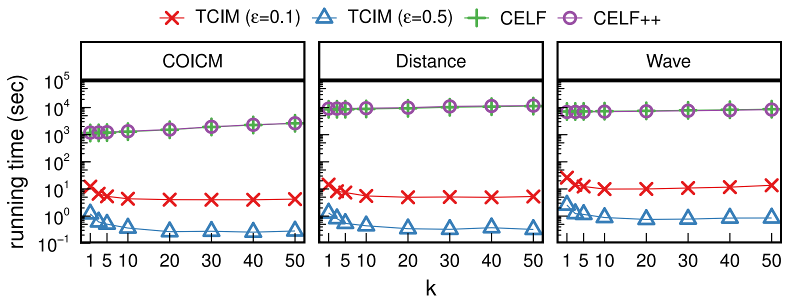


Fig. 4. Results on the *Facebook-like network*: Running time vs. k under three propagation models. ($|S_A| = 50$, $\ell = 1$).

k nodes iteratively. In each iteration, for a given set S_A and current S_B, we select a node u such that it has the maximum number of outgoing edges targeting nodes not in $S_A \cup S_B$.

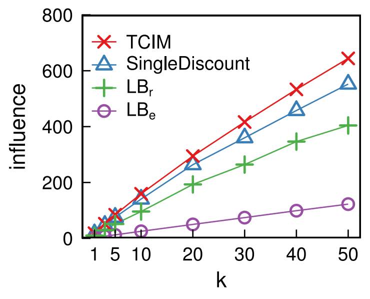
For the TCIM algorithm, let $\mathcal{R}$ be all RAPG instances generated in Algorithm 1 and let S_B be the returned seed set for source B, we report $n \cdot \left(\sum_{R\in\mathcal{R}} f_R(S_B)\right) / |\mathcal{R}|$ as the estimation of $\sigma(S_B|S_A)$. For other algorithms tested, we estimate the influence spread of the returned solution S_B using 50,000 Monte-Carlo simulations. In each experiment, we run each algorithm three times and report the average results.

Parametric values. For TCIM, the default parametric values are $|S_A| = 50$, $\epsilon = 0.1$, $k = 50$, $\ell = 1$. For CELF and CELF++, we run 10,000 Monte-Carlo simulations to estimate the expected influence spread of each candidate seed set S_B under consideration, following the setting in the literature (e.g., [1]). One should note that the number of Monte-Carlo simulations required in all of our experiments is much larger than 10,000 by Theorem 5. For each dataset, the seed set S_A for source A is returned by the TCIM algorithm with parametric values $S_A = \emptyset$, $\epsilon = 0.1$ and $\ell = 1$.

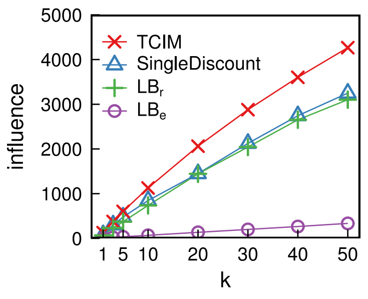
Results on small network: We first compare TCIM to CELF, CELF++ and the SingleDiscount heuristic on the *Facebook-like social network*. Fig. 3 shows the expected influence spread of S_B selected by TCIM and other methods. The influence spread of S_B returned by TCIM, CELF and CELF++ are comparable. The influence spread of the seeds selected by SingleDiscount is slightly less than other methods. Interestingly, there is no significant difference between the expected influence spread of the seeds returned by TCIM with $\epsilon = 0.1$ and $\epsilon = 0.5$, which shows that the quality of solution does not degrade too quickly with the increasing of ϵ. Fig. 4 shows the running time of TCIM, CELF and CELF++, with k varying from 1 to 50. Note that we did not show the running time of SingleDiscount because it is a heuristic method and the expected influence spread of the seeds returned is inferior to the influence spread of the seeds returned by the other three algorithms. Fig. 4 shows that among three influence propagation models, as compared to CELF and CELF++, TCIM runs *two to three orders of magnitude faster* if $\epsilon = 0.1$ and *three to four orders of magnitude faster* when $\epsilon = 0.5$. CELF and CELF++ have similar running time because most time is spent to select the first seed node for source B and CELF++ differs from CELF starting from the selection of the second seed.

Results on large networks: For *NetHEPT* and *Epinion*, we experiment by varying k, $|S_A|$ and ϵ to demonstrate the efficiency and effectiveness of the TCIM. We compare the influence spread of TCIM to SingleDiscount heuristic only, since CELF and CELF++ do not scale well on larger datasets.

Fig. 5 shows the influence spread of the solution returned by TCIM and SingleDiscount, where the influence propagation model is the *Wave propagation model*. We also show the value of LB_e and LB_r returned by the lower bound estimation and refinement algorithm. On both datasets, the expected influence of the seeds returned by TCIM exceeds the expected influence of the seeds return by SingleDiscount. Moreover, for every k, the lower bound LB_r improved by Algorithm 3 is significantly larger than the lower bound LB_e returned by Algorithm 2. For the TCIM framework, generating RAPG instances consumes the largest fraction of running time. Recall that the number of RAPG instances we generate is inversely proportional to the lower bound we estimate. Therefore, the significant decrease of the estimated lower bound shows the

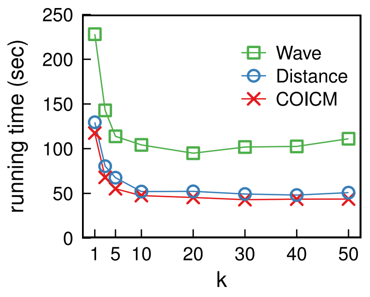


(a) *NetHEPT*.

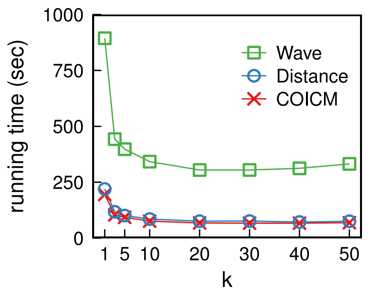


(b) *Epinion*.

Fig. 5. Results on large datasets: Influence vs. k under the *Wave propagation model*. ($|S_A| = 50, \epsilon = 0.1, \ell = 1$).

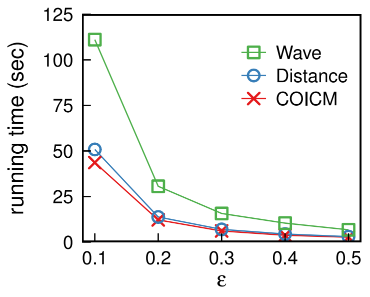


(a) *NetHEPT*.

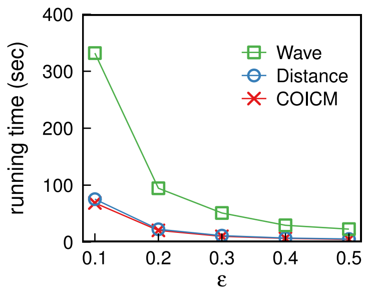


(b) *Epinion*.

Fig. 6. Results on large datasets: Running time vs. k under three propagation models. ($|S_A| = 50, \epsilon = 0.1, \ell = 1$).



(a) *NetHEPT*.



(b) *Epinion*.

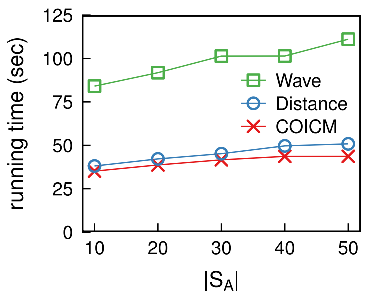
Fig. 7. Results on large datasets: Running time vs. ϵ under three propagation models. ($|S_A| = 50, k = 50, \ell = 1$).

important role of Algorithm 3 in reducing the total running time. When the influence propagation model is COICM or the *Distance-based model*, the results are similar to that in Fig. 5.

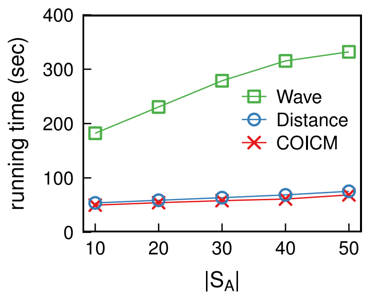
Fig. 6 shows the running time of TCIM, with k varying from 1 to 50. For every influence propagation model, when $k = 1$, the running time of TCIM is the largest. With the increase of k, the running time tends to drop first, and it may increase slowly after k reaches a certain number. This is because the running time of TCIM is mainly related to the number of RAPG instances generated in Algorithm 1, which is $\theta = \lambda/LB_r$. When k is small, LB_r is also small as OPT is small. With the increase of k, if LB_r increases faster than the decrease of λ, θ decreases and the running time of TCIM also tends to decrease. From Fig. 6, we see that TCIM is especially efficient for large k. Moreover, for every k, TCIM based on COICM is the smallest while the running time of TCIM based on the *Wave propagation model* is the largest. This is consistent with the analysis of the running time of TCIM in Section 5.

Fig. 7 shows that the running time of TCIM decreases quickly with the increase of ϵ, which is consistent with its $O(c(\ell + k)(m + n) \log n/\epsilon^2)$ time complexity. When $\epsilon = 0.5$, TCIM finishes within 7 s for *NetHEPT* dataset and finishes within 23 s for *Epinion* dataset. This implies that if we do not require a very tight approximation ratio, we could use a larger ϵ as input and the performance of TCIM could improve significantly.

Fig. 8 shows the running time of TCIM as a function of the seed-set size of source A. For every propagation model, when $|S_A|$ increases, OPT decreases and LB_r tends to decrease. As a result, the total number of RAPG instances required in the node selection phase increases and consequently, the running time of TCIM also increases.

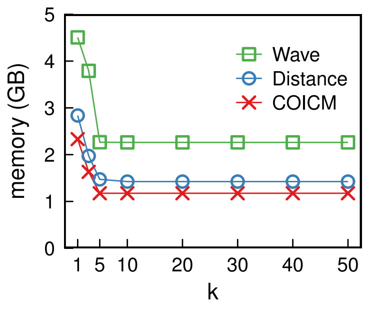


(a) *NetHEPT*.

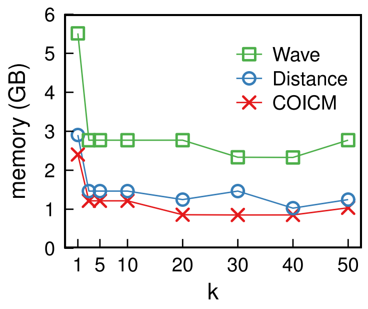


(b) *Epinion*.

Fig. 8. Results on large datasets: Running time vs. $|S_A|$ under three propagation models. ($k = 50, \epsilon = 0.1, \ell = 1$).



(a) *NetHEPT*.



(b) *Epinion*.

Fig. 9. Results on large datasets: Memory consumption vs. k under propagation models. ($|S_A| = 50, \epsilon = 0.1, \ell = 1$).

Fig. 9 shows the memory consumption of TCIM as a function of k. For any k, TCIM based on COICM consumes the least amount of memory because we only need to store nodes for each RAPG instance. TCIM based on *Wave propagation model* consumes the largest amount of memory because we need to store both nodes and edges of each RAPG instance. In the *Distance-based model*, for each RAPG instance, we do not need to store the edges, but need to store some other information; therefore, the memory consumption is in the middle. For all three models and on both datasets, the memory usage drops when k increases because the number of RAPG instances required tends to decrease.

7.2. *TCIM with partial information*

In this subsection, we report results for the general *Competitive Influence Maximization problem with Partial information* (CIMP) and demonstrate the importance of allowing the seed of a competitor being a "*distribution*" over all nodes. We also report the running time and memory consumption for each experiment. Here, the default parametric values of TCIM are $\epsilon = 0.1$, $k = 50$ and $\ell = 1$. We run each experiment ten times and report the average results.

We consider the scenario where we know the competitor has the budget to influence 50 nodes as its initial adopters. Moreover, we "guess" that the seed set of our competitor is a set of nodes selected by the single source influence maximization algorithm [2], or nodes with the highest degree, or nodes with highest closeness centrality. We denote the above mentioned three sets by S_g, S_d and S_c. Let $\mathcal{D}_A$ be our estimated seed distribution. For each node u, define $\mathcal{D}_A(u)$ as the probability of u being in a set randomly chosen from the above mentioned three possible sets. We refer to the seed distribution $\mathcal{D}_A$ as the "*mixed method distribution*".

Table 1 shows the results of this scenario for datasets *NetHEPT* and *Epinion*. For each dataset, we run the TCIM framework given seed distribution $\mathcal{D}_A$, given explicit seed set S_g, S_d and S_c. For each seed set S_B returned, we run 50,000 Monte-Carlo simulations to compute the influence given that the true seed set of competitor being S_g, S_d and S_c. We also report the average influence given different true seed set S_A. Denote the seed set returned by the TCIM framework given the *mixed method distribution* by S_B^*. For example, given that the network is *NetHEPT* and the propagation model is COICM, Table 1 shows that $\sigma(S_B^*|S_g) = 599.82$, $\sigma(S_B^*|S_d) = 632.23$ and $\sigma(S_B^*|S_c) = 657.49$. And, on average, the expected influence of S_B^* is 629.85. From Table 1, we can see that the seed set S_B^* has higher average influence than seed sets returned by running TCIM given S_g, S_d and S_c as seeds of the competitor. Moreover, we observe from Table 1 that, for any true seed set S_A, S_B^* has a larger influence than any other seed set returned by TCIM given a wrong guess of the explicit seed set. For example, suppose the network is *NetHEPT*, the propagation model is COICM, and the "true" seed set of our competitor is S_g. Let S_B' be the set returned by TCIM given S_d as a (wrong) guess of S_A. In this case, we have $\sigma(S_B'|S_g) = 400.18 < \sigma(S_B^*|S_g) = 599.82$. When the influence propagation model is the *Distance-based model*, the results are similar to those in Table 1. This indicates

Table 1
Expected influence of seeds S_B returned by the TCIM framework given the "*mixed method distribution*" (mixed method) as seed distribution for source A or given the guess of explicit seeds of A. Seeds "greedy" for source A is the set of nodes selected by single source influence maximization algorithm. The set "degree" for source A (resp. "centrality") denotes the top 50 nodes ranked by (out)degree (resp. closeness centrality). ($k = 50, \epsilon = 0.1, \ell = 1$).

Dataset	Estimated $\mathcal{D}_A/S_A$	Influence given explicit S_A selected by different methods ($\lvert S_A\rvert = 50$)							
		COICM				*Wave propagation model*			
		Greedy	Degree	Centrality	Average	Greedy	Degree	Centrality	Average
NetHEPT	Mixed method	599.82	632.23	657.49	**629.85**	586.58	624.41	650.39	**620.46**
	Greedy	658.38	515.72	519.50	**564.53**	644.53	525.70	515.37	**561.87**
	Degree	400.18	702.93	622.15	**575.09**	372.58	693.95	613.98	**560.17**
	Centrality	233.14	478.74	763.43	**491.77**	201.72	462.66	752.97	**472.45**
Epinion	Mixed method	2781.71	4603.63	10 683.26	**6022.87**	2773.17	4494.80	10 517.00	**5928.32**
	Greedy	4440.93	3958.87	6372.13	**4923.98**	4265.87	3813.06	6377.30	**4818.74**
	Degree	3130.99	5473.33	7283.28	**5295.87**	2983.56	5299.18	7258.24	**5180.33**
	Centrality	224.93	2809.74	12 078.70	**5037.79**	204.01	2721.87	12 075.78	**5000.55**

Table 2
Running time and memory consumption of the TCIM framework given the "*mixed method distribution*" (mixed method) as seed distribution for source A or given the guess of explicit seeds of A. ($k = 50, \epsilon = 0.1, \ell = 1$).

Dataset	Estimated $\mathcal{D}_A/S_A$	COICM		*Wave propagation model*	
		Time (s)	Memory (GB)	Time (s)	Memory (GB)
NetHEPT	Mixed method	24.73	1.16	107.41	5.63
	Greedy	25.46	1.16	97.75	5.18
	Degree	23.29	1.16	90.15	4.82
	Centrality	21.42	1.11	82.11	4.42
Epinion	Mixed method	38.61	0.77	250.12	8.63
	Greedy	47.05	0.82	273.67	9.46
	Degree	35.07	0.61	250.72	4.45
	Centrality	30.62	0.36	93.91	0.78

that if one is not confident of "guessing" the competitor's seed set correctly, using a *mixed method distribution* as the seed distribution and run TCIM can be a good strategy.

Table 2 shows that, for each datasets, the running time and memory consumption of the TCIM framework given the seed distribution are comparable to those of TCIM given the explicit seed distribution. Because running TCIM given the *mixed method distribution* returns a seed set with significantly higher average influence for both datasets and for both propagation models reported, we conclude that running TCIM given properly estimated seed distribution is a good strategy when we have no prior knowledge about the explicit seed set of our competitor.

8. Conclusion

In this work, we introduce a *General Competitive Independent Cascade* (GCIC) model, define the *Competitive Influence Maximization* (CIM) problem and the *Competitive Influence Maximization problem with Partial information* (CIMP). We show that the CIMP problem is a generalization of the CIM problem with a milder assumption about the knowledge of the competitor's initial adopters. We then present a *Two-phase Competitive Influence Maximization* (TCIM) framework to solve the CIMP problems under the GCIC model. TCIM returns $(1-1/e-\epsilon)$-approximate solutions with probability of at least $1-n^{-\ell}$ and has an efficient expected time complexity $O(c(\ell+k)(m+n)\log n/\epsilon^2)$, where c depends on specific influence propagation model and may also depend on k and graph G. To the best of our knowledge, this is the first general algorithmic framework for both the Competitive Influence Maximization (CIM) problem and the Competitive Influence Maximization problem with Partial information (CIMP) with both performance guarantee and practical running time. We analyze TCIM under the *Campaign-Oblivious Independent Cascade model* [6], the *Distance-based model* and the *Wave propagation model* [4]. We show that, under these three models, the value of c is $O(1)$, $O(k)$ and $O(kn)$ respectively. We provide extensive experimental results to demonstrate the efficiency and effectiveness of TCIM. For the CIM problem, the experimental results show that TCIM returns solutions comparable with those returned by the previous algorithms with the same quality guarantee, but it runs *up to four orders of magnitude faster* than them. In particular, when $k = 50$, $\epsilon = 0.1$ and $\ell = 1$, given the set of 50 nodes selected by the competitor, TCIM returns the solution within 6 min for a dataset with 75,879 nodes and 508,837 directed edges. We also present extensive experimental results for the CIMP problem, which demonstrate the importance of allowing the TCIM framework to take the "*seed distribution*" as an input. We show via experiments that when we have no prior knowledge of the competitor's seed set, running the TCIM framework given an estimated "seed distribution" of competitor is a good and effective strategy.

Appendix

In the following lemma, we first state the Chernoff–Hoeffding bound in the form that we will use throughout our derivation.

Lemma 5 (*Chernoff–Hoeffding Bound*). *Let X be the summation of θ i.i.d. random variables bounded in $[0, 1]$ with a mean value μ. Then, for any $\delta > 0$,*

$$\Pr[X > (1+\delta)\theta\mu] \le \exp\Big(-\frac{\delta^2}{2+\delta}\cdot\theta\mu\Big), \tag{5}$$

$$\Pr[X < (1-\delta)\theta\mu] \le \exp\Big(-\frac{\epsilon^2}{2}\cdot\theta\mu\Big). \tag{6}$$

To prove Theorem 1, we first provide and prove the following lemma.

Lemma 6. *Suppose we are given a set $\mathcal{R}$ of θ random RAPG instances, where θ satisfies Inequality (3) as follows:*

$$\theta \ge (8+2\epsilon)n\cdot\frac{\ell\ln n + \ln\binom{n}{k} + \ln 2}{OPT\cdot\epsilon^2}.$$

Then, with probability at least $1-n^{-\ell}$,

$$\left|\frac{n}{\theta}\cdot F_{\mathcal{R}}(S_B) - \sigma(S_B|\mathcal{D}_A)\right| < \frac{\epsilon}{2}OPT \tag{7}$$

holds for all $S_B \subseteq V$ with k nodes.

Proof of Lemma 6. First, let S_B be a given seed set with k nodes. Let $\mu = \mathbb{E}[f_R(S_B)]$, $F_{\mathcal{R}}(S_B) = \sum_{R\in\mathcal{R}} f_R(S_B)$ can be regarded as the sum of θ i.i.d. variables with a mean μ. By Lemma 1, we have $\mu = \sigma(S_B|\mathcal{D}_A)/n \le OPT/n$. Thus, by Chernoff–Hoeffding bound,

$$\begin{aligned}\Pr\left[\left|\frac{n}{\theta}\cdot F_{\mathcal{R}}(S_B) - \sigma(S_B|\mathcal{D}_A)\right| \ge \frac{\epsilon OPT}{2}\right] &= \Pr\left[|F_{\mathcal{R}}(S_B) - \theta\cdot\mu| \ge \frac{\epsilon OPT}{2n\mu}\cdot\theta\mu\right]\\ &\le 2\exp\left(-\frac{\left(\frac{\epsilon OPT}{2n\mu}\right)^2}{2+\frac{\epsilon OPT}{2n\mu}}\cdot\theta\mu\right) = 2\exp\left(-\frac{\epsilon^2 OPT^2}{8n^2\mu + 2\epsilon n OPT}\cdot\theta\right)\\ &\le 2\exp\left(-\frac{\epsilon^2 OPT}{(8+2\epsilon)\cdot n}\cdot\theta\right) \le n^{-\ell}/\binom{n}{k}.\end{aligned}$$

The last step follows by Inequality (3). There are at most $\binom{n}{k}$ node set $S_B \subseteq V$ with k nodes. By union bound, with probability at least $1-n^{-\ell}$, Inequality (7) holds for all $S_B \subseteq V$ with k nodes. □

Proof of Theorem 1. Suppose we are given a set $\mathcal{R}$ of θ random RAPG instances where θ satisfies Inequality (3). Let S_B be the set of nodes returned by Algorithm 1 and let S_B^* be the set of nodes that maximizes $F_{\mathcal{R}}(S_B^*)$. As we are using a greedy approach achieving an approximation ratio of $(1-1/e)$ to select S_B, we have $F_{\mathcal{R}}(S_B) \ge (1-1/e)F_{\mathcal{R}}(S_B^*)$.

Let S_B^{opt} be the optimum seed set for source B, i.e., the set of nodes that maximizes the influence spread of B. Then, we have $F_{\mathcal{R}}(S_B^{\text{opt}}) \le F_{\mathcal{R}}(S_B^*)$. By Lemma 6, with probability at least $1-n^{-\ell}$, we have $\sigma(S_B|\mathcal{D}_A) \ge F_{\mathcal{R}}(S_B)\cdot n/\theta - OPT\cdot\epsilon/2$ holds for all $S_B \subseteq V$ with k nodes. Thus, we can conclude

$$\begin{aligned}\sigma(S_B|\mathcal{D}_A) &\ge \frac{n}{\theta}\cdot F_{\mathcal{R}}(S_B) - \frac{\epsilon}{2}OPT\\ &\ge \frac{n}{\theta}\cdot(1-1/e)F_{\mathcal{R}}(S_B^*) - \frac{\epsilon}{2}OPT\\ &\ge \frac{n}{\theta}\cdot(1-1/e)F_{\mathcal{R}}(S_B^{\text{opt}}) - \frac{\epsilon}{2}OPT\\ &\ge (1-1/e)\left(OPT - \frac{\epsilon}{2}OPT\right) - \frac{\epsilon}{2}OPT \ge (1-1/e-\epsilon)OPT,\end{aligned}$$

which completes the proof. □

Proof of Lemma 2. Let S_B^+ be a set formed by k samples from $\mathcal{V}^+$ with duplicated nodes eliminated. Let R be a random RAPG instance R, and let $p_1(R)$ be the probability that S_B^+ overlaps with V_R'. For any S_B^+, we have $f_R(S_B^+) \ge 0$ by the definition of the scoring system. Moreover, if S_B^+ overlaps with V_R', we would have $f_R(S_B^+) = 1$. Hence, $p_1(R) \le f_R(S_B^+)$ holds and $n\cdot\mathbb{E}[p_1(R)] \le n\cdot\mathbb{E}[f_R(S_B^+)] = \sigma(S_B^+|\mathcal{D}_A)$ follows from Lemma 1. Furthermore, we define a probability distribution $\mathcal{E}^+$ over the edges in G where the probability mass of each edge e_{uv} is to $(1-\mathcal{D}(v))$. Let E^+ be a set of edges sampled from $\mathcal{E}^+$ with duplicates removed, and let $p_2(R)$ be the probability that at least one edge in E^+ points to a node in V'_R. It can be verified that $p_1(R) = p_2(R)$. From the definition of $w(R)$, we have $p_2(R) = \alpha(R) = 1 - (1 - w(R)/m')^k$. Therefore, we can conclude that

$$\mathbb{E}[\alpha(R)] = \mathbb{E}[p_2(R)] = \mathbb{E}[p_1(R)] \le \sigma(S_B^+|\mathcal{D}_A)/n,$$

which completes the proof. □

Proof of Lemma 3. We denote N_{RV} as the number of random numbers we generate to decide whether nodes are seeds of source A. Because we want to upper bound the number of random numbers we generate, here we assume that we generate a random number to decide whether node u is a seed of A even if $\mathcal{D}_A(u) = 1$. And, we denote the random numbers we generate to decide whether edges are "active" as N_{RE}. Then, we have $N_R = N_{RV} + N_{RE}$. For each edge e_{uv} we examine during the construction, we generate one random number to decide whether it is "active", and we generate at most one random number to decide whether u is a seed of source A. Therefore, we have $N_{RV} - 1 \le N_{RE}$ and $N_R \le 2N_{RE} + 1$.

We first focus on finding the upper bound of N_{RE}. Recall that V'_R is defined as $V'_R = \{u | u \in V_R, f_R(\{u\}) = 1\}$. If we generate a random number for an edge e_{uv} during the construction of R, we know it must point to a node in V'_R. Hence, let $w'(R)$ be the number of edges in G pointing to any node $u \in V'_R$, we have $N_{RE} \le w'(R)$. Note that $\mathcal{D}_A(u) < 1$ for all $u \in V'_R$. And, we have made the assumption that $\max_{u\in V}\{\mathcal{D}_A(u)|\mathcal{D}_A(u) < 1\} < C$ where C is a non-negative constant less than 1. Hence, we have $N_{RE} \le w'(R) \le w(R)/(1 - C)$.

Now, we can conclude that

$$\begin{aligned}
\frac{n}{m}\mathbb{E}[N_R] &\le \frac{n}{m}\mathbb{E}[2N_{RE} + 1] \le 2n\mathbb{E}\left[\frac{N_{RE}}{m'}\right] + \frac{n}{m} \\
&\le \frac{2n}{1-C} \cdot \sum_R \left(\Pr(R) \cdot \frac{w(R)}{m'}\right) + \frac{n}{m} \\
&\le \frac{2n}{1-C} \cdot \sum_R (\Pr(R) \cdot \alpha(R)) + \frac{n}{m} \\
&= \frac{2n}{1-C} \cdot \mathbb{E}[\alpha(R)] + \frac{n}{m} = \frac{2}{1-C} LB_e + \frac{n}{m},
\end{aligned}$$

and therefore $LB_e \ge (\mathbb{E}[N_R] - 1) \cdot (1 - C)n/(2m)$.

If $\mathbb{E}[N_R] \ge 2$, we have $LB_e \ge \mathbb{E}[N_R] \cdot (1 - C)n/(4m)$ and $\mathbb{E}[N_R]/LB_e \le 4m/((1 - C)n)$. For the case where $\mathbb{E}[N_R] < 2$, we have $\mathbb{E}[N_R]/LB_e^* < 2$ because $LB_e^* \ge 1$. We can now conclude that $O(\mathbb{E}[N_R]/LB_e^*) = O(1 + m/n)$, which completes the proof. □

Proof of Lemma 4. As $LB_r = \max\{F/(1 + \epsilon'), LB_e^*\}$ and $LB_e^* \le OPT$, it suffices to show $F/(1 + \epsilon') \le OPT$ holds with probability at least $1 - n^{-\ell}$. By Line 13 in Algorithm 3, we know $F/(1 + \epsilon') \le OPT$ if and only if $\sum_{R\in\mathcal{R}''} f_R(S'_B) \le OPT \cdot \theta'(1 + \epsilon')/n$. Let $\mu = \mathbb{E}[f_R(S'_B)]$, by Lemma 1, we have $n\mu = \sigma(S'_B|\mathcal{D}_A) \le OPT$. Since $\sum_{R\in\mathcal{R}''} f_R(S'_B)$ is the summation of $|\mathcal{R}''| = \theta'$ i.i.d. random variable with mean μ, by $n\mu \le OPT$, $LB_e^* \le OPT$ and Chernoff bound,

$$\begin{aligned}
\Pr\left[\sum_{R\in\mathcal{R}''} f_R(S'_B) \ge \frac{OPT(1+\epsilon')}{n} \cdot \theta'\right] &\le \Pr\left[\sum_{R\in\mathcal{R}''} f_R(S'_B) - \mu\theta' \ge \frac{OPT\epsilon'}{n\mu} \cdot \mu\theta'\right] \\
&\le \exp\left(-\frac{\left(\frac{OPT\epsilon'}{n\mu}\right)^2}{2 + \frac{OPT\epsilon'}{n\mu}} \cdot \mu\theta'\right) = \exp\left(-\frac{OPT^2\epsilon'^2}{2n^2\mu + OPT\epsilon' n} \cdot \theta'\right) \\
&\le \exp\left(-\frac{OPT\epsilon'^2}{(2+\epsilon')n} \cdot \frac{\lambda'}{LB_e^*}\right) \le \exp\left(-\frac{\epsilon'^2\lambda'}{(2+\epsilon')n}\right) \le \frac{1}{n^\ell}.
\end{aligned}$$

The last inequality holds since $\lambda' = (2 + \epsilon')\ell n \ln n/\epsilon'^2$ and this completes the proof. □

Proof of Theorem 3. From Lemma 4, it remains to analyze the time complexity of Algorithm 3. Theorem 2 shows that the expected running time of Algorithm 2 is $O(\ell(m + n) \log n)$, which means that the total number of edges in $\mathcal{R}'$ is at most $O(\ell(m+n) \log n)$. Hence, the running time of Lines 2–8 is $O(c\ell(m+n) \log n + kn) = O(c(\ell + k)(m + n) \log n)$. The running time of the Lines 9–13 is $O\left(\lambda'/LB_e^* \cdot \mathbb{E}[N_R]\right)$, because we generate λ'/LB_e^* RAPG instances and the running time of computing $f_R(S'_B)$ for an RAPG instance $R = (V_R, E_R, S_{R,A})$ is linear with $|V_R|$. From Lemma 3, $O\left(\frac{\lambda'}{LB_e^*} \cdot \mathbb{E}[N_R]\right) = O\left(\lambda'(1 + \frac{m}{n})\right) = O(\ell(m + n) \log n/\epsilon'^2)$. To make sure that Algorithm 3 has the time complexity $O(c(m + n)(\ell + k) \log n/\epsilon^2)$, the value of ϵ' must satisfy $\epsilon' \ge \epsilon\sqrt{\ell/(c(\ell + k))}$. In TIM/TIM$^+$ [2] that returns approximation solution for single source influence maximization problem under the IC model, Tang et al. set $\epsilon' = 5\sqrt[3]{\ell \cdot \epsilon^2/(k + \ell)}$ for any $\epsilon \le 1$. Because the GCIC model with $S_A = \emptyset$ is actually the single source IC model, we also set $\epsilon' = 5\sqrt[3]{\ell \cdot \epsilon^2/(k + \ell)}$ for any given $\epsilon \le 1$. Note that $c \ge 1$, it could be verified that $5\sqrt[3]{\ell \cdot \epsilon^2/(k+\ell)} \geq \sqrt{\ell/(c(\ell+k))}\epsilon$ holds for any $\epsilon \leq 1$. Therefore, Algorithm 3 has expected time complexity $O(c(m+n)(\ell+k)\log n/\epsilon^2)$, which completes the proof. □

Proof of Theorem 5. Let S_B be any node set that contains at most k nodes in V and let $\sigma'(S_B|S_A)$ be the estimation of $\sigma(S_B|S_A)$ computed by r Monte-Carlo simulations. Then, $r\sigma'(S_B|S_A)$ can be regarded as the sum of r i.i.d. random variable bounded in $[0, 1]$ with the mean value $\sigma(S_B|S_A)$. By Chernoff–Hoeffding bound, if r satisfies Inequality (4), it could be verified that

$$\Pr\left[|\sigma'(S_B|S_A) - \sigma(S_B|S_A)| > \frac{\epsilon}{2k}OPT\right] \leq \frac{1}{k \cdot n^{\ell+1}}.$$

Given G and k, *GreedyMC* considers at most kn node sets with sizes at most k. Applying the union bound, with probability at least $1 - n^{-\ell}$, we have

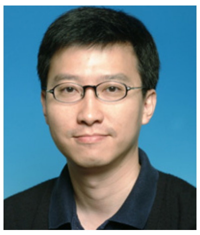
$$|\sigma'(S_B|S_A) - \sigma(S_B|S_A)| > \frac{\epsilon}{2k} \cdot OPT \tag{8}$$

holds for all sets S_B considered by the greedy approach. Under the assumption that $\sigma'(S_B|S_A)$ for all sets S_B considered by *GreedyMC* satisfies Inequality (8), *GreedyMC* returns a $(1 - 1/e - \epsilon)$-approximate solution. For the detailed proof of the accuracy of *GreedyMC*, we refer interested readers to [2] (Proof of Lemma 10). □

References

[1] D. Kempe, J. Kleinberg, É Tardos, Maximizing the spread of influence through a social network, in: Proc. of KDD'03, 2003.
[2] Y. Tang, X. Xiao, Y. Shi, Influence maximization: Near-optimal time complexity meets practical efficiency, in: Proc. of SIGMOD'14, 2014.
[3] S. Bharathi, D. Kempe, M. Salek, Competitive influence maximization in social networks, in: Internet and Network Economics, Springer, 2007.
[4] T. Carnes, C. Nagarajan, S.M. Wild, A. Van Zuylen, Maximizing influence in a competitive social network: a follower's perspective, in: Proc. of ICEC'07, 2007.
[5] A. Borodin, Y. Filmus, J. Oren, Threshold models for competitive influence in social networks, in: Internet and Network Economics, Springer, 2010.
[6] C. Budak, D. Agrawal, A. El Abbadi, Limiting the spread of misinformation in social networks, in: Proc. of WWW'11, 2011.
[7] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: Proc. of ICDM'07, 2007.
[8] W. Chen, C. Wang, Y. Wang, Scalable influence maximization for prevalent viral marketing in large-scale social networks, in: Proc. of KDD'10, 2010.
[9] W. Chen, Y. Yuan, L. Zhang, Scalable influence maximization in social networks under the linear threshold model, in: Proc. of ICDM'10, 2010.
[10] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: Proc. of KDD'09, 2009.
[11] A. Goyal, W. Lu, L.V. Lakshmanan, Celf++: optimizing the greedy algorithm for influence maximization in social networks, in: Proc. of WWW'11, 2011.
[12] K. Jung, W. Heo, W. Chen, IRIE: Scalable and robust influence maximization in social networks, in: Proc. of ICDM'12, 2012.
[13] C. Borgs, M. Brautbar, J. Chayes, B. Lucier, Maximizing social influence in nearly optimal time, in: Proc. of SODA'14, Vol. 14, SIAM, 2014.
[14] G.L. Nemhauser, L.A. Wolsey, Best algorithms for approximating the maximum of a submodular set function, Math. Oper. Res. 3 (3) (1978).
[15] J. Kostka, Y.A. Oswald, R. Wattenhofer, Word of mouth: Rumor dissemination in social networks, in: Structural Information and Communication Complexity, Springer, 2008.
[16] X. He, G. Song, W. Chen, Q. Jiang, Influence blocking maximization in social networks under the competitive linear threshold model, in: Proc. of SDM'12, SIAM, 2012.
[17] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functions—I, Math. Program. 14 (1) (1978).
[18] T. Opsahl, P. Panzarasa, Clustering in weighted networks, Soc. Networks 31 (2) (2009).
[19] M. Richardson, R. Agrawal, P. Domingos, Trust management for the semantic web, in: The Semantic Web-ISWC 2003, Springer, 2003.

Yishi Lin received her B.E. degree from the School of Computer Science and Technology at the University of Science and Technology of China in 2013. She is currently a Ph.D. candidate in the Department of Computer Science and Engineering at the Chinese University of Hong Kong, under the supervision of Prof. John C.S. Lui. Her main interests include social influence and applications, graph algorithms and network economics.



John C.S. Lui is currently the Choh-Ming Li Professor in the Department of Computer Science & Engineering at The Chinese University of Hong Kong (CUHK). He received his Ph.D. in Computer Science from UCLA. After his graduation, he joined the IBM Almaden Research Laboratory/San Jose Laboratory and participated in research and development projects on file systems and parallel I/O architectures. He later joined the Department of Computer Science and Engineering at CUHK. His current research interests are in Internet, network sciences, machine learning on large data analytics, network/system security, network economics, large scale distributed systems and performance evaluation theory. John received various departmental teaching awards and the CUHK Vice-Chancellor's Exemplary Teaching Award. John also received the CUHK Faculty of Engineering Research Excellence Award (2011–2012), he is a co-recipient of the best paper award in the IFIP WG 7.3 Performance 2005, IEEE/IFIP NOMS 2006, and SIMPLEX 2013. He is an elected member of the IFIP WG 7.3, Fellow of ACM, Fellow of IEEE, Senior Research Fellow of the Croucher Foundation.