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

A R T I C L E   I N F O

Article history:
Available online xxxx

Keywords:
Competitive influence maximization
Information diffusion
Viral marketing
Social networks

A B S T R A C T

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 \((1 - 1/e - \varepsilon)\)-approximate solution with probability of at least \((1 - n^{-\lambda})\), 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 \(O(c(k + \ell)(m + n) \log n/e^2)\), 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.

© 2015 Elsevier B.V. All rights reserved.

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 $1 - n^{-c}$, 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 [5], 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 \( 1 - n^{-1} \) and runs in time \( \Omega((\epsilon + k)(m + n) \log n/\epsilon^2) \), 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 \( \sigma \) 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, un-influenced, 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: \( S_1, S_2 \) and \( S_3 \). Nodes in state \( S_1 \) (or \( S_2 \)) are influenced by source \( A \) (resp. \( B \)). Initially, nodes in \( S_2 \) and \( S_3 \) have not been influenced by any source. Nodes in state \( S_3 \) are called the susceptible state, and nodes in state \( S_2 \) are the influenced state, and nodes in state \( S_1 = \emptyset \) are the un-influenced state. However, in this section, we treat all nodes in state \( S_2 \) as influenced.

In the GCIC model, for a given active edges in \( E \), a node \( v \) is influenced by the source \( S \) if there is a directed edge \( e_{uv} \in E \) and \( v \in S \). Moreover, let \( d_{uv} = d(v, u) \) be the shortest distance from \( u \) to \( v \) through active edges in \( E \), and assume \( d_{uv} = +\infty \) if \( u \) and \( v \) are not connected.

We introduce the notion of the nearest initial adopter. Given a set of nodes in state \( S \), let \( \sigma(S) \) be the expected number of nodes influenced by source \( S \) at the end of the influence propagation process. The expected influence of \( S \) is the expected number of nodes in state \( S \) at the end of the influence propagation process, where the expectation is taken over the randomness of \( E \). 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 \( \sigma_1(S_A | S_B) \) 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 \( \sigma_1(S_A | S_B) \) is a monotone and submodular function of \( S_A \subseteq V \setminus S_B \) for all \( u \in V \). Let \( \sigma(S_A | S_B) = \sum_{u \in A} \sigma_1(S_A | S_B) \) be the expected influence of \( S_A \). It is also monotone and submodular of \( S_A \subseteq V \setminus S_B \). 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 \( S_A \) given \( S_B = \emptyset \) is equal to the expected influence of \( S_B \) in the single source IC model. Note that there

Please cite this article in press as: Y. Lin, J.C.S. Lui, Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework, Performance Evaluation (2015), http://dx.doi.org/10.1016/j.peva.2015.08.012
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. 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 \( S_B^k \) of \( k \) nodes for source \( B \) such that the expected influence of \( S_B^k \) given \( S_A \) is maximized. Formally, we have:

\[
S_B^k = \arg \max_{S_B \subseteq V \setminus S_A} \mathbb{E}_{S_A \sim D_A}[\sigma(S_B|S_A)].
\]

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 \( D_A \) of source \( A \), instead of the explicit seed set \( S_A \). For each node \( u \in V \), \( D_A(u) \) specifies the probability that \( u \) is a seed of source \( A \). We write \( S_A \sim D_A \) to indicate that the seed set \( S_A \) is drawn from the random seed distribution \( D_A \). 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 \( D_A \) as

\[
\sigma(S_B|D_A) = \mathbb{E}_{S_A \sim D_A}[\sigma(S_B \setminus S_A|S_A)].
\]

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 \( D_A \), \( \sigma(S_B|D_A) \) 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 \( D_A \) for source \( A \). Find a set \( S_B^k \) of \( k \) nodes for source \( B \) such that the expected influence of \( S_B^k \) given \( D_A \) is maximized, i.e.,

\[
S_B^k = \arg \max_{S_B \subseteq V \setminus S_A} \mathbb{E}_{S_A \sim D_A}[\sigma(S_B|D_A)].
\]

For the distribution \( D_A \), we assume that if \( D_A(u) < 1 \) holds for a node \( u \in V \), the probability \( D_A(u) \) is upper bounded by a constant \( C \) with 0 ≤ \( C < 1 \). Moreover, we are only interested in the non-trivial case that \( |V \setminus S_A| \geq k \) for the CIM problem and \( |V| \geq k \) 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 \( \sigma(S_B|D_A) \) (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 \( \sigma(S_B|D_A) \) (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 CIM (resp. CIMP) problem in a polynomial time [14]. In this paper, we provide a solution to the general CIM 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 \( D_A \) 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 \( D_A \). 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 \( S \neq \emptyset \) 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.
Later subsections.

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 $d_R(u, r) \leq d_R(S_A, r)$. 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 $g$ 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 $g$, 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 $V_R = \{2, 5, 7, 9, 10, 11\}$, $E_R = \{e_4, e_5, e_6, e_{10}, e_{11}\}$ 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 $r \in V$, $g \sim \tilde{g}$ 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 $r$, $g \sim \tilde{g}$ 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 $(S_{A,B} \cap S_A) \cap V_R$ 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

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

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 $r$, $g \sim \tilde{g}$ 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 $\mathcal{D}_A$, $S_A$ 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.

Please cite this article in press as: Y. Lin, J.C.S. Lui, Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework, Performance Evaluation (2015), http://dx.doi.org/10.1016/j.peva.2015.08.012
1. Parameter estimation and refinement: Suppose $S^*_p$ is the optimal solution to the CIMP problem. Let $OPT = \sigma(S^*_p \mid D_x)$ be the expected influence spread of $S^*_p$ given $D_x$. In this phase, TCIM estimates and refines a lower bound of $OPT$ and uses the lower bound to derive a parameter $\theta$.

2. Node selection: In this phase, TCIM first generates a set $\mathcal{R}$ of $\theta$ random RAPG instances of $G$ and $D_x$, where $\theta$ 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 $D_x$ of source $A$, the seed set size $k$ for source $B$ and a constant $\theta$, 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 $\mathcal{R}$ random RAPG instances. 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 ($G$, $D_x$, $k$, $\theta$)

1. Generate a set $\mathcal{R}$ of $\theta$ random RAPG instances.
2. Let $MG_R(u) = \sum_{R \in \mathcal{R}} f(R(u))$ for all $u \in V$.
3. Initialize the seed set $S_B = \emptyset$.
4: for $i = 1$ to $k$ do
5: Identify the node $v_i \in V \setminus S_B$ with largest $MG_R(v_i)$.
6: Add $v_i$ to $S_B$.
7: if $i < k$ then
8: // Update $MG_R(u)$ as $\sum_{R \in \mathcal{R}} d_{R}(u) |S_B|$ for all $u \in V \setminus S_B$.
9: Let $\mathcal{R} = \{ R \in \mathcal{R} : d_R(u \setminus S_B) > 0 \}$.
10: for all $R \in \mathcal{R}$ and $u \in V \setminus (S_B \cup S_R)$ do
11: $MG_R(u) = MG_R(u) - d_R(u \setminus V) |S_B|
12: $MG_R(u) = MG_R(u) + d_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 $R = (V_R = \{ r \}, E_R = \emptyset, S_R = \emptyset)$. 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 $S_R$ with probability $D_x(u)$.

**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 $MG_R$ such that $MG_R(u) = F_{\mathcal{R}}(S_B \cup \{ u \}) - F_{\mathcal{R}}(S_B)$ holds for current $S_B$ and all $u \in V \setminus S_B$. We refer to $MG_R$ as the “marginal gain vector”. The initialization of $MG_R$ 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 $MG_R$. 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 $v_i \not\in V_R \setminus S_R$, we have $\Delta_R(u |S_B\setminus\{v_i\})$ for all $u \not\in V \setminus S_B$. And, for any RAPG instance $R$ such that $f_R(S_B) = 1$, for all $u \not\in V \setminus S_B$, we have $\Delta_R(u |S_B\setminus\{v_i\}) = 0$ and the marginal gain of score cannot be further decreased. To conclude, for a given $R = (V_R, E_R, S_R)$ and a node $u \not\in V_R \setminus (S_R \cup S_B)$, $\Delta_R(u |S_B\setminus\{v_i\})$ differs not to compute $\Delta_R(u |S_B\setminus\{v_i\})$ for all $u \in \mathcal{R}$ and $u \not\in V \setminus (S_R \cup S_B)$. Note that for any RAPG instance $R$ and node $v_i$, $\Delta_R(v_i |S_B\setminus\{v_i\}) > 0$ implies $v_i \in V_R \setminus S_R$ 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 $\theta$ 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 \( MG_R \) 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 \( V \setminus S_0 \) with the largest marginal gain of score, which takes time \( O(n) \). Hence, Algorithm 1 runs in \( O(kn + \theta \cdot \mathbb{E}[|N_0|] + c\theta \cdot \mathbb{E}[|E_R|]) \) expected time. Moreover, from the fact that \( \mathbb{E}[|E_R|] \leq \mathbb{E}[|N_0|] \) and \( c = \Omega(1) \), the expected running time can be written in a more compact form as

\[
O(kn + c\theta \cdot \mathbb{E}[|N_0|]).
\]  

(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 \( \theta \) is, the more accurate is the estimation of the expected influence. The key challenge is how to determine the value of \( \theta \), i.e., the number of RAPG instances required, so as to achieve certain accuracy of the estimation. Specifically, we like to find a \( \theta \) such that the node selection algorithm returns a \( (1 - 1/e - \epsilon) \)-approximation solution. At the same time, we want \( \theta \) 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, \( n \cdot F_R(S_B)/\theta = n \cdot (\sum_{R \in \mathcal{R}} I_R(S_B)) / \theta \) could be an accurate estimate of the influence spread of \( S_B \) given \( D_R \). Then, the set \( S_B \) of \( k \) nodes returned by the greedy algorithm has a large \( F_R(S_B) \) and also a large expected influence spread. Specifically, for Algorithm 1, we have the following theorem.

**Theorem 1.** Given that \( \theta \) satisfies

\[
\theta \geq (8 + 2\epsilon)n \cdot \frac{\ell \ln n + \ln \binom{n}{\ell} + \ln 2}{OPT \cdot e^2}.
\]

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

**Proof.** Please refer to Appendix. \( \square \)

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

4.3. Parameter estimation

The goal of our parameter estimation algorithm is to find a lower bound \( LB_e \) of \( OPT \) so that \( \theta = \lambda/LB_e \geq \lambda/OPT \). 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 \( D_A(u) \) 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 \((1 - D_A(u))d^-(u)\). 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 \( S_B^k \) with duplicated nodes eliminated. A natural lower bound of \( OPT \) is the expected influence of \( S_B^k \) given the seed distribution \( D_A \) of source \( A \), i.e., \( \sigma(S_B^k | D_A) \). Furthermore, a lower bound of \( \sigma(S_B^k | D_A) \) is also a lower bound of \( OPT \). In the following lemma, we present a lower bound of \( \sigma(S_B^k | D_A) \).

**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 \( w(R) = \sum_{v \in V_R} (1 - D_A(u))d^-(u) \). We define \( m' = \sum_{u} (1 - D_A(u))d^-(u) \) and \( \alpha(R) = 1 - \left( 1 - \frac{w(R)}{m'} \right)^k \). We have \( n \cdot \mathbb{E}[\alpha(R) \leq \sigma(S_B^k | D_A) \), where the expectation of \( \mathbb{E}[\alpha(R) \) is taken over the randomness of \( R \).

**Proof.** Please refer to Appendix. \( \square \)

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 \( \Omega(n\ell \log ne^{-2}/LB_e) \). 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 \( n \geq 2 \) and \( \ell \geq 1/2 \), 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, \( \mathbb{E}[1/LB_e'] < 12/LB_e \).

Please cite this article in press as: Y. Lin, J.C.S. Lui, Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework, Performance Evaluation (2015), http://dx.doi.org/10.1016/j.peva.2015.08.012
Algorithm 2 EstimateLB(G, \(\mathcal{D}_k\), \(\ell\)) [20]

1: for \(i = 1 \text{ to } \log_2 n - 1\) do
2: \[\text{Let } c_i = (6\ell \ln n + 6 \ln(\log_2 n)) \cdot 2^i.\]
3: \(s_i = 0.\)
4: for \(j = 1 \text{ to } c_i\) do
5: \(\text{Generate a random RAPG instance } R \text{ and calculate } \alpha(R).\)
6: \(s_i + \alpha(R).\)
7: if \(s_i > c_i/2\) then
8: \(\text{return } LB^* = n \cdot s_i/(2 \cdot c_i).\)
9: \(\text{return } LB^*_2 = 1.\)

Running time of the node selection phase. We now analyze Algorithm 1 assuming \(\theta = \lambda/LB^*_2.\) 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 \(\theta\) random RAPG instances is \(O(\theta \cdot E[N_k]) = O(n^{1+\ell} \cdot E[N_k])\) where \(E[N_k]\) is the expected number of random numbers generated for building a random RAPG instance. The following lemma shows the relationship between \(LB^*_2\) and \(E[N_k].\)

**Lemma 3.** For \(E[N_k]\) and \(LB^*_2\), we have \(O\left(\frac{E[N_k]}{LB^*_2}\right) = O\left(1 + \frac{m}{n}\right).\)

**Proof.** Please refer to Appendix. □

Recall that the greedy selection process in Algorithm 1 has expected time complexity \(O(kn + c\theta \cdot E[N_k]).\) Let \(\theta = \lambda/LB^*_2.\) Applying Lemma 3, the expected running time of Algorithm 1 becomes \(O(kn + c\lambda E[N_k]/LB^*_2) = O(\epsilon(\ell + k)(m + n) \log n/\epsilon^2).\)

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 \(LB^*_c\) is no greater than the expected influence spread of a set of \(k\) independent samples from \(V^+,\) with duplicates eliminated. Hence, \(LB^*_c\) is often much smaller than the \(OPT.\) To narrow the gaps between \(OPT\) and the lower bound \(LB^*_c\) we get in Algorithm 2, we use a greedy algorithm to find a seed set \(S_0\) based on the limited number of RAPG instances we generated in Algorithm 2, and estimate the influence of \(S_0\) with a reasonable accuracy. Then, the intuition is that we can use a creditable lower bound of \(\sigma(S_0|D_k)\) or \(LB^*_c,\) whichever is larger, as the refined bound.

Algorithm 3 refines the lower bound. Lines 2–8 greedily find a seed set \(S_0^*\) based on the RAPG instances generated in Algorithm 2. Intuitively, \(S_0^*\) should have a large influence spread when used as seed set for source \(S.\) Lines 9–13 estimate the expected influence of \(S_0^*\) by Lemma 1, let \(\mathcal{R}_0^*\) be a set of RAPG instances, \(F := n(\sum_{R \in \mathcal{R}_0^*} f_k(S_0^*) / |\mathcal{R}_0^*|)\) is an unbiased estimation of \(\sigma(S_0|D_k).\) Algorithm 3 generates a set \(\mathcal{R}_0\) of sufficiently large number of RAPG instances such that \(F \leq (1 + \epsilon')\sigma(S_0^*|D_k)\) 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 \(LB^*_c = \max(F/(1 + \epsilon'), LB^*_c)\) as the refined lower bound of \(OPT,\) which will be used to derive \(\theta\) in Algorithm 1. The subscript “r” of \(LB^*_c\) stands for “refined”.

Algorithm 3 RefineLB(G, k, \(\mathcal{D}_k\), \(LB^*_c, \epsilon, \ell\))

1: Let \(\mathcal{R}_0\) be the set of RAPG instances generated in Algorithm 2.
2: Let \(MG_{\mathcal{R}_0}(u) = \sum_{R \in \mathcal{R}_0} f_k(S_0^*)\) for all \(u \in V.\)
3: Initialize the seed set \(S_0 = \emptyset.\)
4: for \(i = 1 \text{ to } k\) do
5: \(\text{Identity the node } v_i \in V \setminus S_0 \text{ with largest } MG_{\mathcal{R}_0}(v_i).\)
6: \(\text{Add } v_i \text{ to } S_0.\)
7: if \(i < k\) then
8: \(\text{Update } MG_{\mathcal{R}_0}(u) = \sum_{R \in \mathcal{R}_0} \Delta_R(u|S_0) \text{ for all } u \in V \setminus S_0.\)
9: \(\epsilon' = 5 \cdot \sqrt{\ell + \epsilon^2}/(\ell + k)\)
10: \(\ell' = (2 + \epsilon')/(\ell + k)\)
11: \(\theta' = \lambda/LB^*_c\)
12: \(\text{Generate a set } \mathcal{R}'_0 \text{ of } \theta' \text{ random RAPG instances.}\)
13: \(\text{Let } F = n \cdot \left(\sum_{R \in \mathcal{R}_0} f_k(S_0^*)\right) / \theta'.\)
14: \(\text{return } LB^*_c = \max(F/(1 + \epsilon'), LB^*_c)\)

Theoretical analysis. The following lemma shows that Algorithm 3 returns \(LB^*_c \in [LB^*_c, OPT]\) with high probability.

**Lemma 4.** Given \(LB^*_c \in [LB^*_c/4, OPT],\) Algorithm 3 returns \(LB^*_c \in [LB^*_c, OPT]\) with at least \(1 - n^{-\ell}\) probability.

**Proof.** Please refer to Appendix. □
Budak et al. assumed that one source is prioritized over the other one in the propagation process, and nodes influence by
At step 0, nodes in the model. The influence propagation process starts with two sets of active nodes.

5.1. Campaign-Oblivious Independent Cascade model

Then, we give examples of the score in a simple RAPG instance shown in

5. Applying and analyzing various propagation models under GCIC

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 $k$, $\ell$ and $\epsilon$, TCIM returns a $(1-1/e-\epsilon)$ solution with probability at least $1-n^{-\epsilon}$. First, Algorithm 2 returns the estimated lower bound of $OPT$, denoted by $LB_r^*$. Then, we feed $LB_r^*$ to Algorithm 3 and get a refined lower bound $LB$. Finally, $\theta = \lambda/LB$, and Algorithm 1 returns a set $S_B$ of $k$ seeds for source $B$ based on $\theta = \lambda/LB$ 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: $LB_r = \text{EstimateLB}(G, \mathcal{D}_A, \ell')$
3: $LB = \text{RefineLB}(G, k, \mathcal{D}_A, LB_r^*, \epsilon, \ell')$
4: $\lambda = (8+2e)n (\ell' \ln n + \ln \left(\frac{n}{k}\right) + \ln 2)/\epsilon^2$
5: $\theta = \lambda/LB$
6: $S_B = \text{NodeSelection}(G, \mathcal{D}_A, k, \theta)$
7: $\text{return } S_B$

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 $1-n^{-\epsilon}$. 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 $\ell$ for Algorithms 1–3. By setting this, Algorithms 1–3 each fails with probability at most $n^{-\epsilon}/3$. Hence, by union bound, TCIM succeeds in returning a $(1-1/e-\epsilon)$ approximation solution with probability at least $1-n^{-\epsilon}$. 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 uninfueled 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

Fig. 2. Example of a random RAPG instance with 6 nodes and 6 edges. The set $S_{R,A}$ contains a single node 3.
the dominant source always attempts 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 $I_k$ (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 $E_o$. In COICM, source $B$ ends up influencing a node $u$ if and only if $d_u(S_A, E_o) \leq d_u(S_B, E_o)$. For the RAPG instance $R$ in Fig. 2 where node 3 is a seed of source $A$, we have $f_R(S_A) = 1$ if $S_A \cap \{0, 1, 2, 4, 5\} \neq \emptyset$ and $f_R(S_A) = 0$ otherwise.

**Analysis of TCIM algorithm.** Recall that while analyzing the running time of TCIM, we assume that if $|R| = \theta$, the expected time complexity for the initialization and update of the “marginal gain vector” $MG$ is $O(\theta \cdot E[|E_o|])$. Now we show that $c = O(1)$ for COICM. Suppose we are selecting nodes based on a set $R$ of $\theta$ RAPG instances. The initialization of $MG$ takes time $O(\theta \cdot E[|E_o|])$ for any RAPG instance $R = (V_o, E_o, S_o)$. We have $f_R(u) = 1$ for all $u \in V_o \setminus S_A$ 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 $R' = \{R \in R \setminus E_o \mid d_R(v_i) > 0\}$ in the greedy approach. For every RAPG instance $R \in R'$ and for all $u \in V \setminus (S_B \cup S_A)$, we would have $\Delta(u|S_B) = 0$ and $\Delta(u|S_A) = 1$ and hence we need to update $MG(u)$ correspondingly. For each RAPG instance $R$, it only appears in $R'$ in at most one iteration. Hence, the expected total time complexity for the initialization and update of the “marginal gain vector” is $O(\theta \cdot E[|E_o|])$. It then follows that the expected running time of TCIM is $O(\theta \cdot E[|E_o|])$.

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 $E_o \subseteq E$ of active edges. Let $d_u(E_o, S_A \cup S_B)$ be the shortest distance from $u$ to $S_A \cup S_B$ along edges in $E_o$ and let $d_u(E_o, S_B \cup S_A) = +\infty$ if there $u$ cannot reach any node in $S_A \cup S_B$. For set $S \subseteq V$, we define $h_b(S, d_u(E_o, S_A \cup S_B))$ as the number of nodes in $S$ at distance $d_u(E_o, S_A \cup S_B)$ from $u$ along edges in $E_o$. Given $S_A$, $S_B$ and a set of active edges $E_o$, the probability that node $u$ will be influenced by source $B$ is $\sigma(S_A, S_B) = \sum_{u \in V} h_b(S_A, d_u(E_o, S_A \cup S_B))$, where the expectation is taken over the randomness of $E_o$.

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

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

5.3. Wave propagation

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 $E_o$. Let $\mu(v, S_A, S_B, E_o)$ be the probability that source $B$ influences node $v$. We also let $d_u(S) \cup S, u)$ be the shortest distance from seed nodes to $u$ through edges in $E_o$. Let $N_u$ be the set of neighbors of $u$ whose shortest distance from seed nodes through edges in $E_o$ is $d_u(S \cup S, u) - 1$. Then, Carnes et al. [4] defined $\mu(v, S_A, S_B, E_o) = \sum_{u \in N_u} \mu(v, S_A, S_B, E_o) / |N_u|$. The expected number of nodes $S_B$ can influence given $S_A$ is $\sigma(S_A|S_B) = \sum_{v \in V} \mu(v, S_A, S_B, E_o)$. Hence, the expected total number of nodes being $O(\theta \cdot E[|E_o|])$, where $E_o$ is the set of active edges in $E$.

**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 $1/2$ and influence node $0$ with probability $1/2$. 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 $\theta$ random RAPG instances, the expected time complexity for the initialization and update of the “marginal gain vector” is $O(\theta \cdot E[|E_o|])$. In each iteration of the greedy approach, for each RAPG instance $R = (V_o, E_o, S_A, S_B)$ and each node $u \in V_o \setminus (S_A \cup S_B)$, it takes $O(\theta)$ to update the marginal gain vector. Since there are $\theta$ RAPG instances each having at most $n$ nodes and the greedy approach runs in $k$
iteration, it takes at most $O(kn \cdot \theta \cdot E[D_{R}])$ in total to initialize and update the marginal gain vector. Substituting $\epsilon = O(kn)$ into $O(c(\ell + k)(m + n) \log n / \epsilon^2)$, 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 $S_{A}$ is empty initially and the greedy selection approach runs in $k$ iterations. In the $i$th 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 $\alpha(S_{A} \cup \{v_{i}\}, S_{A}) - \alpha(S_{A}, S_{B})$, and puts it into $S_{A}$. Every estimation of the marginal gain is done by Monte-Carlo simulations. Hence, GreedyMC runs in $O(knm^{2})$ time. Tang et al. [2] provided the lower bound of $\theta$ 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 $1 - n^{-1}$ probability, if

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

Proof. Please refer to Appendix. \hfill \square

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 $\theta$ to the smallest value satisfying Inequality (4), the time complexity of GreedyMC is $O(k^{2}n^{2}m \log n \cdot \epsilon^{-2}/\text{OPT})$. Given that $\text{OPT} \leq n$, the time complexity of GreedyMC is at least $O(k^{2}n^{2}m \log n \cdot \epsilon^{-2})$. Under the assumption that $\ell$ 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 $\delta_{1}$ and $\delta_{2}$ so that $\delta_{1} + \delta_{2} \geq 0$ and $c = O(k^{2}n^{-1}n^{-1} \cdot \epsilon^{-2})$, 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 $c = O(k^{2}n)$, TCIM may still be a better choice than GreedyMC. For example, the time complexity of GreedyMC depends on the value of $\text{OPT}$ or a lower bound of $\text{OPT}$. However, in practice, we usually have no prior knowledge about $\text{OPT}$. Suppose we are not able to get a tight lower bound of $\text{OPT}$, 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 $\text{OPT}$.

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 [38]. (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 $p_{uv} = 1/d_{v}^{-2}$ where $d_{v}^{-2}$ 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
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 \( R \) be all RAPG instances generated in Algorithm 1 and let \( S_B \) be the returned seed set for source \( B \), we report \( n \cdot (\sum_{R \in R} I_f(S_B)) / |R| \) 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 \( \epsilon \). 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 \( \epsilon \) 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 \( LB_r \) and \( LB_s \) 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 returned by SingleDiscount. Moreover, for every \( k \), the lower bound \( LB_s \) improved by Algorithm 3 is significantly larger than the lower bound \( LB_r \) 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
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 $\theta = \lambda / LB$. When $k$ is small, $LB$ is also small as OPT is small. With the increase of $k$, if $LB$ increases faster than the decrease of $\theta$, $LB$ 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 $\epsilon$, which is consistent with its $O(c(\epsilon + k)(m + n)\log n/\epsilon^2)$ 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 $\epsilon$ 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 $LB$ 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.
When the influence propagation model is the set returned by TCIM given sup-pose the network is has a larger influence than any other seed set returned by TCIM given a wrong guess of the explicit seed set. For example, TCIM given is 629.85. From shows that method distribution average influence given different true seed set simulations to compute the influence given that the true seed set of competitor being given seed distribution . We also report the running time and memory consumption for each experiment. Here, the default parametric values of TCIM are .

Moreover, we “guess” that the seed set of our competitor is a set of nodes selected by the single source influence maximization algorithm . (a) NetHEPT. (b) Epinion.

Fig. 8. Results on large datasets: Running time vs. \(|S_\text{u}|\) under three propagation models. (\(k = 50, \epsilon = 0.1, \ell = 1\)).

Fig. 9. Results on large datasets: Memory consumption vs. \(k\) under propagation models. (\(|S_\text{u}| = 50, \epsilon = 0.1, \ell = 1\)).

Table 1 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 \(S_\text{g}, S_\text{d}\) and \(S_\text{c}\). Let \(\mathcal{D}_u\) be our estimated seed distribution. For each node \(u\), define \(\mathcal{D}_u(u)\) as the probability of \(u\) being in a set randomly chosen from the above mentioned three possible sets. We refer to the seed distribution \(\mathcal{D}_u\) 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 \(\mathcal{D}_u\), given explicit seed set \(S_\text{g}, S_\text{d}\) and \(S_\text{c}\). For each seed set \(S_\text{g}\) returned, we run 50,000 Monte-Carlo simulations to compute the influence given that the true seed set of competitor being \(S_\text{g}, S_\text{d}\) and \(S_\text{c}\). We also report the average influence given different true seed set \(S_\text{c}\). Denote the seed set returned by the TCIM framework given the mixed method distribution by \(S_\text{w}\). For example, given that the network is NetHEPT and the propagation model is COICM, Table 1 shows that \(\sigma(S_\text{c}|S_\text{w}) = 599.82, \sigma(S_\text{d}|S_\text{w}) = 632.23\) and \(\sigma(S_\text{g}|S_\text{w}) = 657.49\). And, on average, the expected influence of \(S_\text{w}\) is 629.85. From Table 1, we can see that the seed set \(S_\text{w}\) has higher average influence than seed sets returned by running TCIM given \(S_\text{g}, S_\text{d}\) and \(S_\text{c}\) as seeds of the competitor. Moreover, we observe from Table 1 that, for any true seed set \(S_\text{g}, S_\text{d}\), \(S_\text{w}\) 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 \(S_\text{g}\). Let \(S_\text{g}^\ast\) be the set returned by TCIM given \(S_\text{g}\) as a (wrong) guess of \(S_\text{g}\). In this case, we have \(\sigma(S_\text{g}|S_\text{g}^\ast) = 400.18 < \sigma(S_\text{g}|S_\text{g}) = 599.82\).

When the influence propagation model is the Distance-based model, the results are similar to those in Table 1. This indicates
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 CIM 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 CIM problems under the GCIC model. TCIM returns \((1 - \frac{1}{e} - \epsilon)\)-approximate solutions with probability of at least \(1 - n^{-\epsilon}\) and has an efficient expected time complexity \(O(c(k+n)(m+n) \log n/e^2)\), 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 the 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 6. To prove for all have approach achieving an approximation ratio of \( \frac{\mu}{2+\mu} \).

The last step follows by Inequality (6) as the sum of \( \mu \). Then, for any \( \beta \geq 0 \),

\[
\Pr[X > (1+\beta)\theta\mu] \leq \exp\left(-\frac{\delta^2}{2+\beta} \cdot \theta\mu\right),
\]

(5)

\[
\Pr[X < (1-\beta)\theta\mu] \leq \exp\left(-\frac{\epsilon^2}{2} \cdot \theta\mu\right).
\]

(6)

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

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

\[
\theta \geq (8+2\epsilon) n \cdot \frac{\ell \ln n + \ln \left(\binom{k}{2}\right)}{OPT \cdot \epsilon^2}.
\]

Then, with probability at least \( 1 - n^{-f} \),

\[
\left| \frac{n}{\theta} \cdot F_\mathcal{R}(S_b) - \sigma(S_b|\mathcal{D}_b) \right| < \frac{\epsilon}{2} OPT
\]

(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 \( \mu = \mathbb{E}[f_b(S_b)], F_\mathcal{R}(S_b) = \sum_{R \in \mathcal{R}} f_b(S_b) \) can be regarded as the sum of \( \theta \) i.i.d. variables with a mean \( \mu \). By Lemma 1, we have \( \mu = \sigma(S_b|\mathcal{D}_b)/n \leq OPT/n \). Thus, by Chernoff–Hoeffding bound,

\[
\Pr\left[ \frac{n}{\theta} \cdot F_\mathcal{R}(S_b) - \sigma(S_b|\mathcal{D}_b) \geq \frac{\epsilon OPT}{2} \right] = \Pr\left[ |F_\mathcal{R}(S_b) - \theta \cdot \mu| \geq \frac{\epsilon OPT}{2n \mu} \cdot \theta \mu \right] \\
\leq 2 \exp\left(-\frac{\epsilon^2 OPT^2}{8n^2 \mu + 2\epsilon OPT \cdot \theta} \cdot \theta \mu \right) \\
\leq 2 \exp\left(-\frac{\epsilon^2 OPT}{(8+2\epsilon) \cdot n} \cdot \theta \right) \leq n^{-f} \left(\frac{n}{k}\right).)
\]

The last step follows by Inequality (3). There are at most \( \binom{n}{2} \) node set \( S_b \subseteq V \) with \( k \) nodes. By union bound, with probability at least \( 1 - n^{-f} \), 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 \( \theta \) random RAPG instances where \( \theta \) 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 \( F_\mathcal{R}(S_b) \geq (1 - 1/e) F_\mathcal{R}(S_b^+) \).

Let \( S_b^\text{opt} \) be the optimum seed set for source \( b \), i.e., the set of nodes that maximizes the influence spread of \( b \). Then, we have \( F_\mathcal{R}(S_b^\text{opt}) \leq F_\mathcal{R}(S_b^+) \). By Lemma 6, with probability at least \( 1 - n^{-f} \), we have \( \sigma(S_b|\mathcal{D}_b) \geq F_\mathcal{R}(S_b) \cdot n/\theta - OPT \cdot \epsilon/2 \) holds for all \( S_b \subseteq V \) with \( k \) nodes. Thus, we can conclude

\[
\sigma(S_b|\mathcal{D}_b) \geq \frac{n}{\theta} \cdot F_\mathcal{R}(S_b) - \frac{\epsilon}{2} OPT \\
\geq \frac{n}{\theta} \cdot (1 - 1/e) F_\mathcal{R}(S_b^+) - \frac{\epsilon}{2} OPT \\
\geq \frac{n}{\theta} \cdot (1 - 1/e) F_\mathcal{R}(S_b^\text{opt}) - \frac{\epsilon}{2} OPT \\
\geq (1 - 1/e) \left( OPT - \frac{\epsilon}{2} OPT \right) - \frac{\epsilon}{2} OPT \geq (1 - 1/e - \epsilon)OPT,
\]

which completes the proof.

**Proof of Lemma 2.** Let \( S_b^+ \) be a set formed by \( k \) samples from \( V^+ \) 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 \( V_b^0 \). For any \( S_b^+ \), we have \( f_b(S_b^+) \geq 0 \) by the definition of the scoring system. Moreover, if \( S_b^+ \) overlaps with \( V_b^\varepsilon \), we would have \( f_b(S_b^+) = 1 \). Hence, \( p_1(R) \leq f_b(S_b^+) \) holds and \( n \cdot \mathbb{E}[p_1(R)] = n \cdot \mathbb{E}[f_b(S_b^+)] = \sigma(S_b^+|\mathcal{D}_b) \) follows from Lemma 1. Furthermore, we define a probability distribution \( \mathcal{E} \) over the edges in \( G \) where the probability mass of each edge \( e \) is to \( (1 - D(v)) \). Let \( \mathcal{E} \) 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 $p_2(R) = \alpha(R) = 1 - (1 - w(R)/m)^2$. Therefore, we can conclude that

$$E[\alpha(R)] = E[p_2(R)] = E[p_1(R)] = \sigma(S^+_A | D_A)/n,$$

which completes the proof.

**Proof of Lemma 3.** We denote $N_{RE}$ 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 $D_A(u) = 1$. And, we denote the random numbers we generate to decide whether edges are “active” as $N_{RE}$. Then, we have $N_{RE} = N_{RV} + N_{RE}$. 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 node $u$ is a seed of source $A$. Therefore, we have $N_{RE} - 1 \leq N_{RE}$ and $N_{RE} \leq 2N_{RE} + 1$.

We first focus on finding the upper bound of $N_{RE}$. Recall that $V'_R$ is defined as $V'_R = \{v | v \in V_R, f_R([u]) = 1\}$. 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 $N_{RE} \leq w(R)$. Note that $D_A(u) < 1$ for all $u \in V'_R$. And, we have the assumption that $\max_{u \in V_R} (D_A(u) | D_A(u) < 1) < C$ where $C$ is a non-negative constant less than 1. Hence, we have $N_{RE} \leq w(R) \leq w(R)/(1 - C)$.

Now, we can conclude that

$$\frac{n}{m}E[N_{RE}] \leq \frac{n}{m}E[2N_{RE} + 1] \leq 2nE\left[\frac{N_{RE}}{m^2}\right] + \frac{n}{m} \leq \frac{2n}{1 - C} \sum_k \left(Pr(R) \cdot \frac{w(R)}{m^2}\right) + \frac{n}{m} \leq \frac{2n}{1 - C} \sum_k (Pr(R) \cdot \alpha(R)) + \frac{n}{m} \leq \frac{2n}{1 - C} \cdot E[\alpha(R)] + \frac{n}{m} = \frac{2}{1 - C}LB_R + \frac{n}{m},$$

and therefore $LB_R \geq (E[N_{RE}] - 1) \cdot (1 - C)n/(2m)$.

If $E[N_{RE}] \geq 2$, we have $LB_R \geq E[N_{RE}] \cdot (1 - C)n/(4m)$ and $E[N_{RE}]/LB_R \leq 4m/(1 - C)n$ for the case where $E[N_{RE}] < 2$, we have $E[N_{RE}]/LB'_R < 2$ because $LB'_R \geq 1$. We can now conclude that $O(E[N_{RE}]/LB'_R) = O(1 + m/n)$, which completes the proof.

**Proof of Lemma 4.** As $LB_R = \max(F/(1 + \epsilon'), LB'_R)$ and $LB'_R \leq OPT$, it suffices to show $F/(1 + \epsilon') \leq OPT$ holds with probability at least $1 - n^{-2}$. By Line 13 in Algorithm 3, we know $F/(1 + \epsilon') \leq OPT$ if and only if $\sum_{R \in R'} f_R(S_R') \leq OPT \cdot \theta'/(1 + \epsilon')/n$. Let $\mu = E[R(S_R')]$, by Lemma 1, we have $\mu \mu = \sigma(S^+_A | D_A) \leq OPT$. Since $\sum_{R \in R'} f_R(S_R')$ is the summation of $|R'| = \theta'$ i.i.d. random variable with mean $\mu$, by $\mu \leq OPT$, $LB'_R \leq OPT$ and Chernoff bound,

$$\Pr\left[\sum_{R \in R'} f_R(S_R') \geq \frac{OPT(1 + \epsilon')}{n} \cdot \theta'\right] \leq \Pr\left[\sum_{R \in R'} f_R(S_R') - \mu \theta' \geq \frac{OPT \epsilon'}{n \mu} \cdot \theta'\right] \leq \exp\left(-\frac{\left(\frac{OPT \epsilon'}{n \mu} \cdot \theta'\right)^2}{2 \cdot \frac{OPT \epsilon'}{n \mu} \cdot \theta'}\right) = \exp\left(-\frac{OPT \epsilon'^2}{2n^2 \mu} \cdot \theta'/\theta'\right) \leq \exp\left(-\frac{OPT \epsilon'^2}{2(2 + \epsilon')n} \cdot \frac{\theta'}{LB'_R}\right) \leq \exp\left(-\frac{\theta'^2}{(2 + \epsilon')n} \cdot \frac{\theta'}{LB'_R}\right) \leq 1.$$ 

The last inequality holds since $\theta' = (2 + \epsilon')\ell n \in n^{-1/2}$ 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 $R'$ 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 $O(\ell (LB'_R \cdot E[N_{RE}]))$, because we generate $\ell' / LB'_R$ RAPG instances and the running time of computing $f_R(S_R')$ for an RAPG instance $R = (V_R, E_R, S_{R,A})$ is linear with $|V_R|$. From Lemma 3, $O(\ell (LB'_R \cdot E[N_{RE}] = O(\ell ((1 + \frac{\theta'}{LB'_R}) = O(\ell (m + n) \log n^{-1/2})$. To make sure that Algorithm 3 has the time complexity $O(c(\ell (m + n) \log n - \epsilon^2)$, the value of $\epsilon$ must satisfy $\epsilon^2 \geq \epsilon^2 / (\ell (\ell + k))$. In TIP/TIM + [2] that returns approximation solution for single source influence maximization problem under the IC model, Tang et al. set $\epsilon = 5\frac{\ell}{\ell'} \cdot \epsilon^2 / (\ell + k)$ for any $\epsilon \leq 1$. Because the GCIC model with $S_0 = \emptyset$ is actually the single source IC model, we also set $\epsilon = 5\frac{\ell}{\ell'} \cdot \epsilon^2 / (\ell + k)$ for any given $\epsilon \leq 1$. Note that $c \geq 1$.
it could be verified that $5\sqrt{e} \cdot \epsilon^2 / (k + \epsilon) \geq \epsilon / (c(\ell + k))\epsilon$ 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 $S_0$ be any node set that contains at most $k$ nodes in $V$ and let $\sigma'(S_0|S_0)$ be the estimation of $\sigma(S_0|S_0)$ computed by $r$ Monte-Carlo simulations. Then, $r\sigma'(S_0|S_0)$ can be regarded as the sum of $r$ i.i.d. random variable bounded in $[0, 1]$ with the mean value $\sigma(S_0|S_0)$. By Chernoff–Hoeffding bound, if $r$ satisfies Inequality (4), it could be verified that

$$\Pr\left[|\sigma'(S_0|S_0) - \sigma(S_0|S_0)| > \frac{\epsilon}{2k} \cdot \text{OPT}\right] \leq \frac{1}{k \cdot r^{1/2}}.$$ 

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

$$|\sigma'(S_0|S_0) - \sigma(S_0|S_0)| > \frac{\epsilon}{2k} \cdot \text{OPT}$$

holds for all sets $S_0$ considered by the greedy approach. Under the assumption that $\sigma'(S_0|S_0)$ for all sets $S_0$ 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


