applied sciences

MDPI

Article

Competitive Influence Maximization within Time and Budget Constraints in Online Social Networks: An Algorithmic Approach

Canh V. Pham [1], Hieu V. Duong [2], Huan X. Hoang [1,*] and My T. Thai [3]

[1] Vietnam National University, University of Engineering and Technology, Hanoi 100803, Vietnam; cvpham.vnu@gmail.com
[2] People's Security Academy, Hanoi 100803, Vietnam; dvhieubg95@gmail.com
[3] Department of Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611, USA; mythai@cise.ufl.edu
* Correspondence: huanhx@vnu.edu.vn

Received: 19 April 2019 ; Accepted: 28 May 2019; Published: 1 June 2019

Abstract: Competitive Influence Maximization (CIM) problem, which seeks a seed set nodes of a player or a company to propagate their product's information while at the same time their competitors are conducting similar strategies, has been paid much attention recently due to its application in viral marketing. However, existing works neglect the fact that the limited budget and time constraints can play an important role in competitive influence strategy of each company. In addition, based on the the assumption that one of the competitors dominates in the competitive influence process, the majority of prior studies indicate that the competitive influence function (objective function) is monotone and submodular.This led to the fact that CIM can be approximated within a factor of $1 - 1/e - \epsilon$ by a Greedy algorithm combined with Monte Carlo simulation method. Unfortunately, in a more realistic scenario where there is fair competition among competitors, the objective function is no longer submodular. In this paper, we study a general case of CIM problem, named Budgeted Competitive Influence Maximization (BCIM) problem, which considers CIM with budget and time constraints under condition of fair competition. We found that the objective function is neither submodular nor suppermodular. Therefore, it cannot admit Greedy algorithm with approximation ratio of $1 - 1/e$. We propose Sandwich Approximation based on Polling-Based Approximation (SPBA), an approximation algorithm based on Sandwich framework and polling-based method. Our experiments on real social network datasets showed the effectiveness and scalability of our algorithm that outperformed other state-of-the-art methods. Specifically, our algorithm is scalable with million-scale networks in only 1.5 min.

Keywords: social networks; competitive influence maximization; optimization; approximation algorithm

1. Introduction

Online social networks (OSNs) have recently been a very effective method for diffusing information, propagating opinions or ideas. Many companies have leveraged word-of-mouth effect in OSNs to promote their products. The key problem of viral marketing is Influence Maximization (IM), which aims to select a set of k users (called *seed set*) in a social network with maximum influence spread. Kempe et al. [1] first formulated IM problem in two diffusion models, named Linear Threshold (LT) and Independent Cascade (IC), which simulate the propagation of influence through social networks. IM has been widely studied due to its important role in viral marketing [2–10]. However, all of the above-mentioned studies only focus on studying influence propagation of single player or company in social networks. In the context of viral marketing, there are often many competitors simultaneously

implementing the same strategy of marketing spread on OSNs. This phenomenon requires the task of maximizing a product's influences under competitive circumstances, called Competitive Influence Maximization (CIM) problem.

Bharathi et al. [11] first proposed CIM problem, which seeks a seed set to maximize the propagation of their product's information while their competitors employ the same strategy. Since then, related works have tried to investigate CIM in many different contexts. Some authors show that the objective function is *monotone* and *submodular*. A set function f over a ground U is set to be submodular if

$$f(A \cup \{u\}) - f(A) \geq f(B \cup \{u\}) - f(B) \quad (1)$$

for any $A \subseteq B \subseteq U, u \in U \setminus B$. Based on that, they applied the classic hill-climbing algorithm, which provides a napproximation ratio of $(1 - 1/e)$ to solve CIM problem [12–17]. For example, Lu et al. [12] studied the problem in context of fair competitive influence from the host perspective. Chen et al. [13] proposed an independent cascade model with negative opinions (IC-N) model by extending IC model and showed a greedy algorithm with the approximation ratio of $1 - 1/e$. Recently, some works have addressed the problem in other directions, including proposing heuristic algorithm [15] and studying some variants of CIM [16,18,19].

Although previous works try to solve the CIM problem in many circumstances, the feasibility of the existing works is limited for following reasons. Firstly, they often assume that one of the competitors takes advantage of the competitive influence process. In this case, the objective function is monotone and submodular. As a result, the existing algorithms provide an approximation ratio of $1 - 1/e$ based on using Greedy framework algorithm, which sequentially selects the node that has the largest marginal benefit [20]. However, users also have different views when they receive the same information. As a result, the influence function is no longer submodular and there is no approximation algorithm for this case. Secondly, the prior works do not take into account time constraint and cost to select a seed user for CIM. In a more realistic scenario, the effectiveness of competitive influence process depends very much on these two factors. Thirdly, although many CIM algorithms have been proposed, there are no scalable and efficient algorithms for CIM in large social networks (million-scale). For the problems related to information diffusion, the complexity of calculating the objective function is enormous due to the randomness of the probabilistic diffusion model [8,9]. To address these challenges, some works use Mote-Carlo method to estimate the objective function [1,13,15,17]. However, the method requires high complexity, and it takes several hours even on very small networks. To the best of our knowledge, there is no randomized algorithm for CIM that can meet approximation guarantee with low complexity.

In this paper, we study a general problem of CIM, *Budgeted Competitive Influence Maximization* (BCIM), which takes into account both arbitrary cost and time constraints for CIM problem. To model this problem, we first introduce Time constraint Competitive Linear Threshold (TCLT) to capture competitive influence progress within time constraint by extending Competitive Linear Threshold [21,22]. Under TCLT model, the main challenges of BCIM lie in following aspects. Firstly, BCIM problem is NP-hard and it is #P-Hard [23] to calculate the objective function. Moreover, we point out that the objective function is neither submodular nor supermodular. Thus, it makes BCIM difficult to be solved using greedy-based algorithms, as well as methods for influence maximization. To address the above challenges, in this article, we present SPBA, an efficient randomized algorithm based on polling method and Sandwich Approximation framework [16]. Our main contributions are summarized as follows:

- We formulate Time constraint Competitive Linear Threshold (TCLT) model by extending Competitive Linear Threshold model in [21,22] to simulate competitive influence within time constraint τ. Given two competitors A and B who need to advertise their productions on OSNs, assume that we know nodes that are activated by B (B-seed set). Given the limited budget L, heterogeneous cost of each node to active by A (i.e., each node has a cost to add it into A-seed

set), and the time constraint τ, we study BCIM problem, which aims to seek A-seed set nodes within limited budget L and time constraint τ to maximize nodes influenced by A under TCLT model. We then show that BCIM is NP-hard and the objective function is neither *submodular* nor *supermodular*.

- We propose SPBA, an efficient randomized algorithm based on Sandwich approximation and polling method. We first design *upper* bound and *lower* bound submodular functions of the objective function and develop a polling-based approximation algorithm to find the solution of bound functions that guarantees approximation ratio of $(1 - 1/\sqrt{e} - \epsilon)$ with high probability. Based on that, the Sandwich framework approximation in [16] is applied to give a data-dependent approximation factor.
- We conducted extensive experiments on various real social networks. The experiments suggest that SPBA provides significantly higher quality solutions than existing methods including baseline algorithms and influence maximization algorithms. Furthermore, we also demonstrate that our algorithm can scale to million-scale networks within about 1.5 min.

Organization. The rest of paper is organized as follows. The related work is presented in Section 2 and the preliminaries for Competitive Linear Threshold model and Competitive Influence Maximization are introduced in Section 3. We introduce our propagation model, problem definition and its properties in Section 4. Section 5 presents our proposed algorithms. The experiments are shown in Section 6. Finally, we give some tasks for future work and conclusion in Section 7.

2. Related Work

Since CIM is one of variants of IM, we review the literature related to this work from two areas, namely influence maximization and competitive influence maximization.

2.1. Influence Maximization

The IM problem is a crucial problem in information diffusion research due to its potential commercial value. Basically, IM focuses on finding a set of k seed users on a social network to maximize the number of influenced nodes. Kempe et al. [1] first proposed two information diffusion models, Linear Threshold (LT) and Independent Cascade (IC). On these models, they formulated IM problem as a combinatorial optimization problem and designed a natural greedy algorithm with approximation ratio is $1 - 1/e$. IM has been received much attention from the following aspects: proposing efficiency algorithms [2–9,24] and studying its variants [7,10,25–28].

Kempe et al. [1] first purposed Greedy algorithm based on Monte-Carlo simulation with $(1 - 1/e - \epsilon)$ approximation guarantee. To improve the running time of Greedy algorithm, Leskovec et al. [3] proposed the cost-effective lazy forward (CELF) algorithm, which is up to 700 times faster than Greedy algorithm. This algorithm is improved in [29]. Several works propose heuristic algorithms to find solutions in large networks [8,9,30,31]. Although those heuristics are often faster in practice, they fail to retain the $(1 - 1/e - \epsilon)$ approximation guarantee and often give lower results than greedy algorithm. Chen et al. [9] proposed a heuristic algorithm based on the maximum influence arborescence (MIA) structure. In the LT model, Chen et al. [8] proposed using local directed acyclic graphs (LDAG) to approximate the influence of nodes. Recently, Borgs et al. [2] made a theoretical breakthrough for finding solution to IM by proposing Reverse Influence Sampling (RIS) algorithm. RIS algorithm returns a $(1 - 1/e - \epsilon)$ approximate solution with probability at least $1 - n^{-l}$. The main idea of the RIS algorithm is to generate Reachable Reverse (RR) sets to estimate the objective function and use greedy algorithm for a collection that includes a large enough RR set to the find solution. This motivates many state-of-the-art methods for IM including TIM/TIM++ [5], IMM [6], and SSA/D-SSA [4].

Recently, IM's variants have also received much attention due to their potential commercial value [25,27,28]. Lin et al. [28] studied k-Boosting problem, which aims at finding the set of k users

to boost so that the "boosted" influence spread is maximized. The authors of [25] investigated distance-aware influence maximization, which takes into account the effect of distance on influence process for IM problem. They showed that the objective function is monotone and submodular and proposed RIS-based and MIA-based algorithms. Influence Maximization with awareness of the topic are also studied in [27]. In this work, each user is associated with a profile that consists of the users preferences on different topics in their model and the problem asks to select seed set with budget and topic queries so that the competitive influence function is maximized.

2.2. Competitive Influence Maximization

In the context of viral marketing, it is often the case that many companies propagate their product's information simultaneously, leading to the fact that the Competitive Influence Maximization (CIM) problem has been studied in recent years. Bharathi et al. [11] first proposed CIM problem by proposing a new propagation model, which is an extension of IC model. Later, some authors proposed variations of the IC model for CIM problem. Chen et al. [13] investigated CIM under the context of combating the dissemination of negative opinions under IC-N model. This model is based on the observation that rumors and misinformation are often more attractive than official information. Lui et al. [14] considered CIM problem under a new diffusion–containment model and presented a $(1 - 1/e)$-approximation algorithm. Carnes et al. [17] proposed distance based and wave propagation models in competitive influence process social networks. They showed that the objective function is submodular and devised a greedy algorithm with approximation ratio of $(1 - 1/e)$. Some other works propose an expanding LT model approach for CIM problem [12,21,22,24]. For instance, Borodin et al. [24] proposed Competitive Linear Threshold models and provided some property results of these models. Lu et al. [12] considered the problem of fair competitive viral marketing from the host's perspective. They proposed K-LT model and showed that the influence function is monotone and submodular. Generally, Chen et al. [21] summarized two competitive influence models, which are extended from the IC model and the LT model, named Competitive Independent Cascade (CIC) and Competitive Linear Threshold (CLT) models. They also provided the properties of such models and categorized them by tie-breaking rules including fixed probability tie-breaking (TB-FP) rule and proportional probability tie-breaking (TB-PP) rule. TB-FP means that, when a node v is influenced by both competitors, it will be influenced by one of them with a fixed probability. TB-FP means that v becomes influenced by one competitor with a proportional probability. TB-FP reflects the dominance of a competitor and most purposed algorithms are based on this feature to give an approximation of $1 - 1/e$. However, there is no approximation algorithm for TP-FP case.

In other directions, Bozrgi et al. [15] proposed a community-based algorithm for CIM under DC model. Some variants of CIM have been studied. Yan et al. [19] found the seed set with minimum cost set for threshold competitive influence problem. Lu et al. [16], Yan et al. [19] proposed competition and complementary approaches for CIM problem by extending IC model. The authors of [18] formulated Dominated Competitive Influence Maximization (DCIM) problem, which aims to maximize the difference in value between the influence of desired information and its competitors under a new competitive independent cascade model with meeting events.

Different from most of the existing works, in this paper, we study a general problem of CIM, namely BCIM, which considers the CIM within budget and time constraints under condition of fair competition. We show that the objective function is not submodular and calculate that the objective function is #P-Hard. To overcome this challenge, we propose a randomized algorithm based on Sandwich approximation and polling-based method.

3. Preliminaries

To clearly introduce the problem of BCIM, we first introduce some preliminaries. Table 1 summarizes the frequently used notations.

Table 1. Notations.

Notations	Descriptions
n, m	the number of nodes and the number of edges
$N_-(v), N_+(v)$	the sets of incoming, and outgoing neighbor nodes of v
S_A, S_B	seed sets of A and B, respectively
n_0	$n - \lvert S_B \rvert$
$\mathbb{I}(\cdot), \mathbb{L}(\cdot), \mathbb{U}(\cdot)$	The expected number of A-active nodes, its lower bound and its upper bound, respectively
$\hat{\mathbb{U}}_c(S_A), \hat{\mathbb{U}}_t(S_A)$	Estimations of $\mathbb{U}(S_A)$ over set $\mathcal{R}_c$ and $\mathcal{R}_t$, respectively
S_A^*, S_L^*, S_U^*	Optimal solution for BCIM, optimal solution for maximizing $\mathbb{L}(\cdot)$, and $\mathbb{U}(\cdot)$
OPT, OPT_l, OPT_u	$\mathbb{I}(S^*), \mathbb{L}(S_L^*), \mathbb{U}(S_U^*)$
$\Upsilon(\epsilon, \delta)$	$1 + (1+\epsilon)(2 + \frac{2}{3}\epsilon) \ln \frac{2}{\delta} \frac{1}{\epsilon^2}$
$\mathsf{Cov}_{\mathcal{R}}(S)$	number of LRR (or URR) sets R_j be covered by S
k_{max}	$\max\{k : \exists A \subseteq V, c(A) \leq L\}$
α, β	$(1 - \frac{1}{\sqrt{e}})\sqrt{\ln(\frac{2}{\delta})}$, $\sqrt{\left(1 - \frac{1}{\sqrt{e}}\right)\left(\ln \frac{2}{\delta} + \ln \binom{n}{k_{max}}\right)}$
$N(\epsilon, \delta)$	$2n \left(\alpha + \beta\right)^2 \frac{1}{\epsilon^2 \mathsf{OPT}_l}$

*3.1. Competitive Linear Threshold (*CLT*) Model*

In this model, a social network is abstracted by a directed graph $G = (V, E)$, where V is the set of nodes (or vertices) representing users and E is the set of edges representing links among users. There are two competitors A and B who want to promote their products in a social network G. Each edge $(u, v) \in E$ has two weights $w_A(u, v)$ and $w_B(u, v)$ representing the influence of A and B on edge (u, v), respectively. The weights satisfy conditions

$$\sum_{u \in N_-(v)} w_A(u, v) \leq 1 \ , \ \sum_{u \in N_-(v)} w_B(u, v) \leq 1, \forall v \in V$$

Each node can choose one of three status: *A-active*, *B-active*, and *inactive*, which represent the nodes that have been successfully activated by A, activated by B, and have not been activated by either A or B. Each node v picks two independent thresholds $\theta_A(v), \theta_B(v)$ *uniformly* from $[0, 1]$, called *A-threshold* and *B-threshold*. The propagation process happens in discrete steps $t = 0, 1, \ldots$. S_A and S_B are the seed sets of competitors A and B ($S_A \cap S_B = \emptyset$). A_t and B_t are the set of *A-active* and *B-active* nodes at step t, respectively. The process of propagation operates as follows:

- At step $t = 0$, $A_0 = S_A, B_0 = S_B$.
- At step $t \geq 1$, it first sets $A_t = A_{t-1}$ and $B_t = B_{t-1}$. Each node $v \notin A_{t-1} \cup B_{t-1}$ becomes A-active if

$$\begin{cases} \sum_{u \in N_-(v) \cap A_{t-1}} w_A(u, v) \geq \theta_A(v) \\ \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u, v) < \theta_B(v) \end{cases} \tag{2}$$

  Node v becomes B-active if

$$\begin{cases} \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u, v) \geq \theta_B(v) \\ \sum_{u \in N_-(v) \cap A_{t-1}} w_A(u, v) < \theta_A(v) \end{cases} \tag{3}$$

  in the case when node u that has the total influence weight of two competitors are greater than corresponding thresholds. Chen et al. [21] summarized tie-breaking rules can be used to determine whether v is A-active or B-active.

  - Fixed probability tie-breaking rule (TB-FP): TB-FP means that with a fixed probability p, u becomes A-active with probability p and becomes B-active with probability $1-p$. The special cases of this rule include TB-FP(A)-competitor A's dominance, TB-FP(B)-competitor B's dominance.
  - Proportional Probability tie-breaking rule (TB-PP): $A_t(v) = N_-(v) \cap A_{t-1} \setminus A_{t-2}$ is *A-active successful attempt set* of u and $B_t(v) = N_-(u) \cap B_{t-1} \setminus B_{t-2}$ is *B-active successful attempt set* of u. Node v becomes A-active with probability $\frac{|A_t(u)|}{|A_t(u)| + |B_t(u)|}$, and u is B-activated with probability $\frac{|B_t(u)|}{|A_t(u)| + |B_t(u)|}$.
- Once a node becomes activated (A-active or B-active), its status remains in next steps. The propagation process ends when no more nodes can be activated.

TB-FP is used in [11,13,22,32,33] to reflect the dominance of one competitor. This is motivated by the phenomenon of negativity bias, which is well studied in social psychology, and matches the common sense that rumors or misinformation are usually hard to fight with in social networks. In contrast, TB-PP reflects fair competition among competitors. This rule is used for IC-N model (a variant of IC model) [13], while no study uses this rule for a variant of LT model.

3.2. Competitive Influence Maximization

Definition 1. *Given a directed graph $G = (V, E)$ representing a social network under an information diffusion model $\mathcal{M}$, there a two competitors A and B. Given B-seed set $S_B \subset V$ and a positive number k, find A-seed set $S_A \subseteq V \setminus S_B$ with $|A| \leq k$ so that the number of A-active nodes is maximized*

4. Models and Problem Definition

4.1. Time Constraint Competitive Linear Threshold (TCLT) Model

In this section, we introduce our model incorporating CLT model with limited spread step τ, namely Time Constraint Competitive Linear Threshold Model (TCLT). In addition, we propose a new tie-breaking rule in our model that can truly reflect the competitive context in viral marketing by our explanation.

In this model, we reuse all notations and symbols in CLT model. Given a constraint of propagation hop $\tau \geq 1$, the propagation process happens in discrete steps $t = 0, 1, \ldots, \tau$ as follows:

- At step $t = 0$, $A_0 = S_A, B_0 = S_B$.
- At step $t \geq 1$, first set $A_t = A_{t-1}$ and $B_t = B_{t-1}$. Each node $v \notin A_{t-1} \cup B_{t-1}$ becomes A-active if

$$\begin{cases} \sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v) \geq \theta_A(v) \\ \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v) < \theta_B(v) \end{cases} \tag{4}$$

  Node v becomes B-active if

$$\begin{cases} \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v) \geq \theta_B(v) \\ \sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v) < \theta_A(v) \end{cases} \tag{5}$$

- If in step t, a node v has

$$\begin{cases} \sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v) \geq \theta_A(v) \\ \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v) \geq \theta_B(v) \end{cases} \tag{6}$$

We propose *weight proportional probability tie-breaking rule* (TB-WPP) to determine its state. Accordingly, v is A-activated with probability.

$$p_A(v|A_{t-1}, B_{t-1}) = \frac{\sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v)}{\sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v) + \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v)}$$

and v is B-activated with probability

$$p_B(v|A_{t-1}, B_{t-1}) = \frac{\sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v)}{\sum_{u \in N_-(v) \cap A_{t-1}} w_A(u,v) + \sum_{u \in N_-(v) \cap B_{t-1}} w_B(u,v)}$$

- Once a node becomes activated (A-active or B-active), it keeps this status in the next steps. The propagation process ends after τ hops of propagation or no more nodes can be activated.

Different from TB-PP, in TB-WPP rule, we consider the total influence weight of the in-neighbors to decide state of node v. Our TB-WPP rule reflects more closely the competition process. Consider the example in Figure 1 to clarify this observation. Graph G contains four nodes $\{a, b, c, u\}$ and three edges $\{(a,u), (b,u), (c,u)\}$. There is a pair (w_A, w_B) on each edge, $S_A = \{a, b\}, S_B = \{c\}$. At step $t = 1$, if we use TB-PP, node v will change its state from inactive to A-active or B-active with probabilities $\frac{2}{3}$ and $\frac{1}{3}$, respectively. In other words, the probability v becomes A-active is higher. If we use TB-WPP, node v would change its state to inactive to A-active or B-active with probabilities $\frac{3}{11}$ and $\frac{8}{11}$, i.e. probability v becomes B-active is higher. In this case, the influence weight of c (0.8) for u is greater than that of two nodes a, b (total influence weight is 0.3). Considering the total weigh in TB-WPP rule is more suitable for the fact that users create different influences on each other depending on the relationship between them. Therefore, it is reasonable to consider TB-WPP about the competitive influence spread.

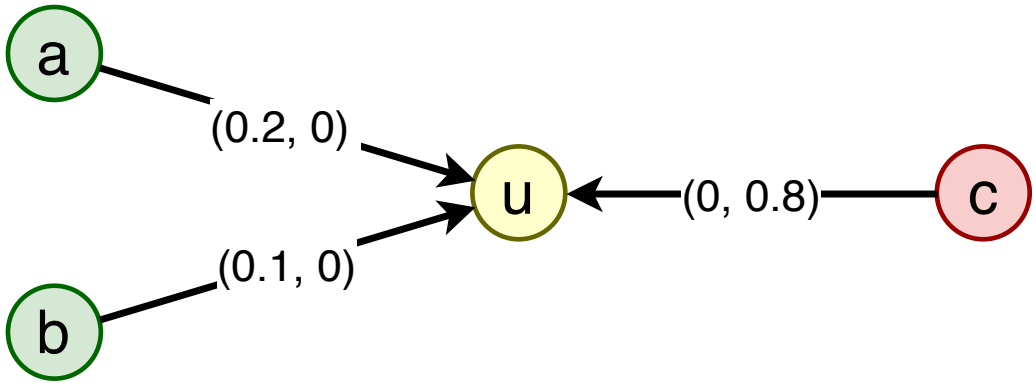

Figure 1. Illustrating for TB-WPP and TB-PP rules.

4.2. Budgeted Competitive Influence Maximization Problem

In this paper, we assume that we have known knowledge that seed set of competitor B is $S_B \subset V$ and each node u is associated with an arbitrary cost $c(u) \geq 0$ to add in S_A. We define *Budgeted Competitive Influence Maximization* (BCIM) as follows:

Definition 2. BCIM *problem. Given a directed graph* $G = (V, E)$ *representing a social network under* TCLT *model, B-seed set* $S_B \subset V$*, a budget* $L > 0$*, and time constraint* τ*, find A-seed set* $S_A \subseteq V \setminus S_B$ *with total cost* $\sum_{u \in A} c(u) \leq L$ *to maximize* $\mathbb{I}(S_A)$*.*

Theorem 1. BCIM *problem is NP-hard and calculating the objective function* $\mathbb{I}(\cdot)$ *is #P-Hard.*

Proof. We see that, when $S_B = \emptyset$ and $\tau = n$, the TCLT model becomes well-known LT model [1] and BCIM becomes IM problem [1]. In other words, IM is a special case of BCIM so BCIM is NP-hard problem and calculating the influence $\mathbb{I}(S_A)$ is #P-hard. □

Although the objective function in IM problem is *monotone* and *submodular* function, unfortunately, the objective function in BCIM is neither *submodular* nor *supermodular*. Therefore, we cannot use the nature greedy for optimizing submodular and supermodular function to get an approximation guarantee.

Theorem 2. *The function $\mathbb{I}(\cdot)$ is neither submodular nor supermodular under* TCLT *model*

Proof. We prove that by counter example (see Figure 2). Consider an instance of BCIM problem $G = (V, E)$ with $V = \{a, b, c, d, e, f\}$, $E = \{(a, b), (b, c), (c, e), (c, d), (c, f)\}$, and $\tau = 2$. The A-weight and B-weight on each edge is equal to 1, and we set $S_B = \{f\}$. In this example, we have $\mathbb{I}(\emptyset) = 0$, $\mathbb{I}(\{a\}) = 2$, $\mathbb{I}(\{a, c\}) = 5$, and $\mathbb{I}(\{a, b, c\}) = 5$. Therefore, $\mathbb{I}(\{a, c\}) - \mathbb{I}(\{a\}) = 3 > \mathbb{I}(\{a\}) - \mathbb{I}(\emptyset) = 2$. That is, $\mathbb{I}(\cdot)$ is not submodular. On other hand, we have $\mathbb{I}(\{a, c\}) - \mathbb{I}(\{a\}) = 3 > \mathbb{I}(\{a, b, c\}) - \mathbb{I}(\{a, c\}) = 0$. Therefore, $\mathbb{I}(\cdot)$ is also not supermodular. □

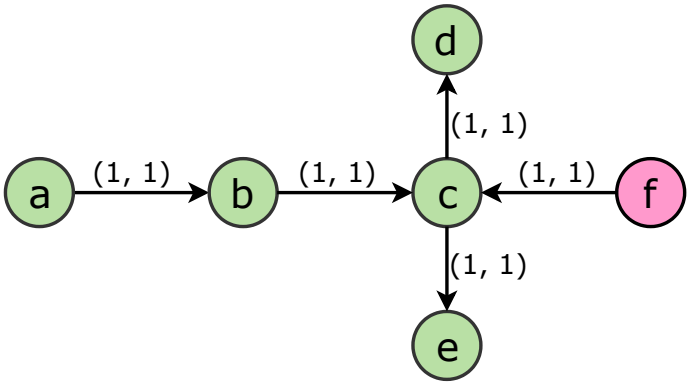

Figure 2. A counter example.

*4.3. Competitive Live-Edge (*CLE*) Model*

We follow the method in [22] to construct a *live-edge* model and prove this model is equivalent to TCLT model. This property is used for estimating the objective function as well as the designing of our algorithm in the next sections.

From original graph $G = (V, E)$ under TCLT model, we construct a *sample* graph (or *realization*) g from G as follows. For each $v \in V$, we randomly select one in-edge (u, v) with probability $w_A(u, v)$, and do not select any in-edge with probability $1 - \sum_{u \in V} w_A(u, v)$. The selected edge is called *A-live edge*. On the other hand, we also randomly select one in-edge (u, v) (called *B-live edge*) with probability $w_B(u, v)$, and do not select any in-edge with probability $1 - \sum_{u \in V} w_B(u, v)$. Let g_A and g_B be the sub-graph including only A-live edges and B-live edges, respectively. Finally, we return g as union of g_A and g_B.

In graph g, we denote A'_t and B'_t as sets of A-active nodes and B-active nodes on g at step t, respectively. we denote $d_A(A_t, u)$ ($d_B(B_t, u)$) was the minimum distance from A_t (B_t) on g_A (g_B) to node u. The distribution of A-active and B-active nodes in g happens in discrete steps t as follows:

- At step $t = 0$, $A'_t = S_A$ and $B'_t = S_B$.
- At step $t \geq 1$, first set $A_t = A_{t-1}$ and $B_t = B_{t-1}$. A node $v \notin A'_{t-1} \cup B'_{t-1}$ becomes *A-active* if v is reachable from A'_{t-1} in one step in g_A (i.e., $d_A(A'_{t-1}, v) = 1$) but not reachable from B'_{t-1} in one step in g_B (i.e., $d_B(B'_{t-1}, v) > 1$), then v is in A'_t. Symmetrically, if v is reachable from B'_{t-1} in one step in g_B but not reachable from A'_{t-1} in one step in g_A, then v is in B'_t.
- If at step $t \geq 1$, v is reachable from A'_{t-1} in one step in g_A and reachable from B'_{t-1} in one step in g_B, v is A-activated with probability

$$p_A(v|A'_{t-1}, B'_{t-1}) = \frac{\sum_{u \in N_-(v) \cap A'_{t-1}} w_A(u, v)}{\sum_{u \in N_-(v) \cap A'_{t-1}} w_A(u, v) + \sum_{u \in N_-(v) \cap B'_{t-1}} w_B(u, v)} \quad (7)$$

and v is B-activated with probability

$$p_B(v|A'_{t-1}, B'_{t-1}) = \frac{\sum_{u \in N_-(v) \cap B'_{t-1}} w_B(u,v)}{\sum_{u \in N_-(v) \cap A'_{t-1}} w_A(u,v) + \sum_{u \in N_-(v) \cap B'_{t-1}} w_B(u,v)} \tag{8}$$

- The process of propagation ends after hop $t = \tau$ or no more nodes can be activated.

We demonstrate the equivalence of two models through the following theorem.

Theorem 3. *For a given A-seed set S_A and B-seed set S_B, the distribution over A-active sets and B-active node sets at hop t for any $t = 1, 2.., \tau$ on* TCLT *model and* CLE *are equivalent.*

The proof of Theorem 3 is presented in Appendix A. We denote X_G as the set of sample graphs generated from G and $\Pr[g|G]$ as the probability of generating sample graph g in G. We have:

$$\Pr[g|G] = \Pr[g_A|G] \cdot \Pr[g_B|G] = \prod_{v \in V} p_A(v, G, g) \cdot \prod_{v \in V} p_B(v, G, g) \tag{9}$$

where

$$p_A(v, G, g) = \begin{cases} w_A(u,v), \text{ If } \exists u : (u,v) \in E(g_A) \\ 1 - \sum_{u:(u,v) \in E} w_A(u,v), \text{otherwise} \end{cases}$$

$$p_B(v, G, g) = \begin{cases} w_B(u,v), \text{ If } \exists u : (u,v) \in E(g_B) \\ 1 - \sum_{u:(u,v) \in E} w_B(u,v), \text{otherwise} \end{cases}$$

$E(g_A)$ and $E(g_B)$ are the set edges of g_A and g_B, respectively. We denote $\mathbb{I}_B^\tau(S_A)$ as the expected number of A-active nodes after τ hops with given B-seed set S_B. For convenience, we simplify $\mathbb{I}_B^\tau(S_A)$ as $\mathbb{I}(S_A)$. Based on the result of Theorem 3, we have:

$$\mathbb{I}(S_A) = \sum_{v \in V \setminus S_B} \sum_{g \in X_G} \Pr[g|G] \gamma_g^v(S) \tag{10}$$

where $\gamma_g^v(S_A)$ is a random variable under sample graph g, defined as follows:

$$\gamma_g^v(S_A) = \begin{cases} 1, \text{ If } v \text{ is } A\text{-active when run CLE model in } g \\ 0, \text{ Otherwise} \end{cases} \tag{11}$$

Node v is called source node. Let v be randomly selected in $V \setminus S_B$ and g be a random graph generated from G. Lemma 1 shows that we can use $\gamma_g^v(\cdot)$ to estimate objective function.

Lemma 1. *For any $S_A \subset V \setminus S_B$, we have $\mathbb{I}(S_A) = n_0 \cdot \mathbb{E}[\gamma(S_A)]$, where $\gamma(S_A)$ is the expectation of $\gamma_g^v(A)$ over all random sources and sample graphs.*

Proof. Since the source node is randomly selected, the probability that v is selected is equal to $\frac{1}{n_0}$ for $n_0 = |V \setminus S_B|$. We have:

$$\begin{aligned} \mathbb{I}(S_A) &= \sum_{g \in X_G} \Pr[g|G] \sum_{v \in V \setminus S_B} \gamma_g^v(S_A) = n_0 \sum_{g \in X_G} \Pr[g] \sum_{v \in V \setminus S_B} \gamma_g^v(S_A) \frac{1}{n_0} \\ &= n_0 \cdot \sum_{g \in X_G} \Pr[g|G] \sum_{v \in V \setminus S_B} \gamma_g^v(S) \Pr[v \text{ is } \textit{source} \text{ node}] \\ &= n_0 \cdot \mathbb{E}[\gamma(S_A)] \end{aligned} \tag{12}$$

The transition from the second to third equality follows from the definition of $\gamma(S_A)$. This complete the proof. □

Based on Lemma 1, we design upper and lower submodular functions, which are cores of our proposed algorithms in next sections.

5. Our Proposed Algorithm for BCIM Problem

In this section, we present SPBA, our approximation algorithm for BCIM problem. Since the $\mathbb{I}(\cdot)$ is not submodular, we use the *Sandwich Approximation* (SA) method [16] to design approximation for the problem.

Outline. Our algorithm contains two key components: (1) We devise the *lower bound* and *upper bound* submodular function of $\mathbb{I}$, namely $\mathbb{L}(\cdot)$ and $\mathbb{U}(\cdot)$, respectively. We then design Polling-Based Algorithm (PBA) a $(1 - 1/\sqrt{e} - \epsilon)$ approximation algorithm to find solution to maximize $\mathbb{L}$ and $\mathbb{U}$ based on polling-based method [4–7]. (2) We apply SA with upper and lower bound function to provide a solution with approximation guarantee. It first finds a solution to the BCIM problem with any strategy. It then finds an approximate solution to the lower bound and the upper bound by PBA algorithm. Finally, it returns the solution that has the best result for the BCIM problem. The framework of SPBA is presented in Figure 3.

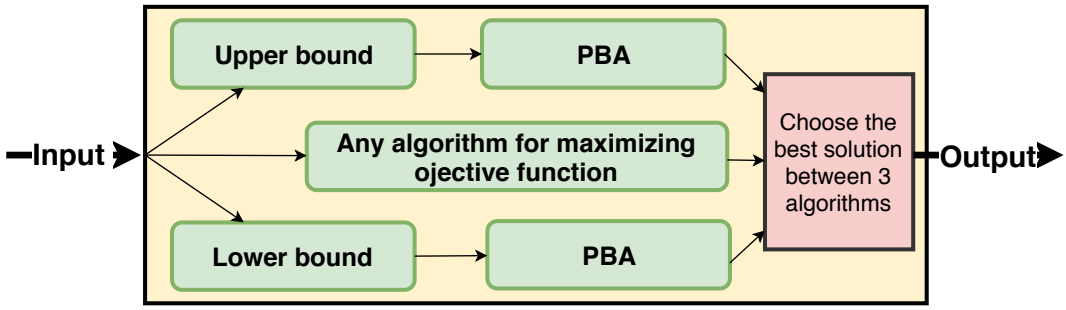

Figure 3. Framework of SPBA algorithm.

5.1. Lower and Upper Bound Functions

We leverage the equivalence between the TCLT and CLE model and result of Lemma 1 to design lower and upper bound of objective functions.

5.1.1. Upper Bound Function

For a random source node v and a sample graph g with given B-seed set S_B, the idea of this method is that we only choose set of nodes satisfying: (1) the distance of influence path from them to v is smaller than τ; and (2) the influence path from them to v is not blocked by S_B. We consider set $C_U(g, v)$ defined as follows:

$$C_U(g, v) = \{u | d_A(u, v) \leq \tau, d_A(u, v) < d_B(u, S_B)\} \tag{13}$$

Figure 4 shows an example of $C(g, v)$. In this example, the sample graph g contains nine nodes and eight edges, the source nodes is v and $S_B = \{c, h\}$ and $\tau = 4$. g_A contains edges: $(a, v), (b, a), (c, b), (d, h)$. g_B contains edges: $(f, v), (e, f), (f, e), (h, f)$. We have $C_U(g, v) = \{b, a, v\}$. Node d lies on the simple path that ends at v, but it cannot influence v since its influence is blocked by c. We define *Upper bound Reachable Reversal* (URR) set as follows:

Definition 3 (URR set)**.** *Given graph $G = (V, E, w_A, w_B)$, a random* URR *set R_j is generated from G by: (1) picking a random source node $v \in V$; and (2) generating a sample graph g from G by running* CLE *model, and returning $R_j \leftarrow C_U(g, v)$.*

For any $S_A \subseteq V \setminus S_B$, denote a random variable:

$$X_j = \begin{cases} 1, \text{ If } R_j \cap S_A \neq \emptyset \\ 0, \text{ Otherwise} \end{cases} \tag{14}$$

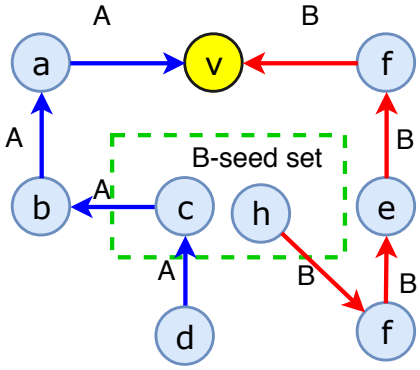

Figure 4. Description for $C_U(g, v)$.

We denote $R_j(g, v)$ is a URR set with source node v and sample graph g, and $X_j(g, v)$ is value of X_j corresponding to $R_j(g, u)$. The following lemma shows the upper bound characters of X_j.

Lemma 2. *For any set $S_A \subseteq V \setminus S_B$, a random source node v and random sample graph g, we have $X_j(g, v) \geq \gamma_g^v(S_A)$.*

Proof. We consider two following cases

Case 1: $S_A \cap R_j(g, v) = \emptyset$, each node $u \in S_A$ cannot reach v in g_A after τ steps. By running CLE model, S_A cannot activate v. Hence, $X_j(g, v) = \gamma_g^v(S_A) = 0$.

Case 2: $S_A \cap R_j(g, v) \neq \emptyset$. Assume that $u \in S \cap R_j(g, v)$, u can reach v in g_A after τ steps, thus $\mathbb{E}[\gamma_g^v(S_A)] \leq 1 = X_j(g, v)$. Therefore, $\gamma_g^v(S_A) \leq X_j(g, v)$. □

Define $\mathbb{U}(S_A) = n_0 \cdot \mathbb{E}[X_j]$ and $\mathcal{R}$ is a set of URR. We estimate $\mathbb{U}(S_A)$ as

$$\hat{\mathbb{U}}(S_A) = \frac{n_0}{|\mathcal{R}|} \sum_{R_j \in \mathcal{R}} X_j \tag{15}$$

Lemma 3 shows the properties of $\mathbb{U}$ function.

Lemma 3. *Given seed set $S_B \subset V$, for any set of nodes $S_A \subseteq V \setminus S_B$, we have: $\mathbb{U}(S_A) \geq \mathbb{I}(S)$ and $\mathbb{U}(\cdot)$ is a monotone and submodular function.*

Proof. Using Lemma 2, we have

$$\begin{aligned} \mathbb{I}(A) &= n_0 \cdot \textstyle\sum_{g \in X_G} \Pr[g|G] \sum_{v \in V \setminus S_B} \gamma_g^v(A) \\ &\leq n_0 \cdot \textstyle\sum_{g \in X_G} \Pr[g|G] \sum_{v \in V \setminus S_B} X_j(g, v) = \mathbb{U}(A) \end{aligned} \tag{16}$$

Since $\mathbb{U}(S) = n_0 \cdot \mathbb{E}[X_j] = n_0 \cdot \sum_{X_j} \Pr[X_j] \mathbb{E}(S \cap R_j \neq \emptyset)$ is a form of *weight coverage* function, in which every R_j is an element, the universal is set of all URR sets, and each node $u \in V$ corresponding to a subsets that contains LRR R_j is covered by u. The probability $n_0 \Pr[g|G]$ is the weight of element $R_j^g(u)$. Since the weighted coverage function is monotone and submodular, it follows that $\mathbb{U}(A)$ has the same properties □

Lemma 3 suggests that we can use $\mathbb{U}$ as an upper bound submodular function of $\mathbb{I}$. We further devise an algorithm, which is summarized in Algorithm 1, to generate an URR set. We first randomly

selected a source node $v \in V \setminus S_B$ with uniform distribution (Line 1). After that, it attempts to select an in-neighbor u of v on g_A according to the CLE model (Line 4). Then, it moves from v to u and repeats the process. The algorithm stops within τ steps (Line 3), when no edge is selected (Line 10), or the selected node v belongs to S_B or belongs to R_j (Line 7).

Algorithm 1: Generate LRR set.

Input: Graph $G = (V, E, w_A, w_B)$, B-seed set $S_B \subseteq V, \tau$
Output: a LRR set R_j

1. Select randomly a node $v \in V \setminus S_B$
2. $g_B \leftarrow \emptyset$; $d_B \leftarrow 0$
3. $R_j \leftarrow \emptyset$
4. **repeat**
5. Add v to R_j
6. Select an A-edge (u, v) by CLE model
7. **if** (u, v) *is selected* **then**
8. If $u \in S_B$ then break
9. $v \leftarrow u$, $d_A = d_A + 1$
10. **if** CheckBF$(u, g_B, d_A - 1).Check = False$ **then**
11. Add v in R_j;
12. $g_B \leftarrow$ CheckBP$(u, g_B, d_A - 1).g_B$; `// Update graph` g_B
13. **else**
14. **return** R_j
15. **end**
16. **else**
17. break
18. **end**
19. **until** $d_A > \tau$;
20. **return** R_j;

5.1.2. Lower Bound Submodular Function

We next devise a lower submodular function of objective function. The idea of this method is that we only choose set of nodes that make v becomes A-active with probability 1 in estimating of objective function. We consider set $C_L(g, v)$ defined as follows:

$$C_L(g, v) = \{u | d_A(u, v) < \tau, d_A(u, w) < d_B(S_B, w), \forall w \in \mathsf{P}_A(u, v)\} \tag{17}$$

where $\mathsf{P}_A(u, v)$ is the simple path from u to v in g_A. The left inequality in Equation (17) ensures that v can reach from u on g_A. The right inequality ensures the influence from u to v is not blocked by the influence from S_B.

Consider the example in Figure 5. We have the sample graph g that contains nine nodes and eight edges, the source nodes is v and $S_B = \{c, h\}$ and $\tau = 5$. g_A contains edges: $(a, v), (b, a), (c, b), (d, b)$. g_B contains edges: $(f, v), (e, f), (h, e), (i, e)$. We have $C_L(g, v) = \{b, a, v\}$. According CLE model, we can easily prove that $\gamma_g^v(u) = 1, \forall u \in C_L(g, v)$. Based on that, we define *Lower bound Reachable Reversal* (LRR) set as follows:

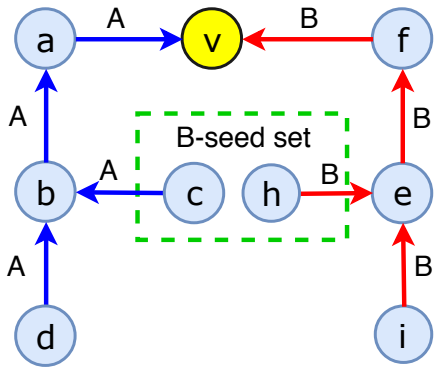

Figure 5. Description for $C_L(g, v)$.

Definition 4. LRR *set. Given graph* $G = (V, E, w_A, w_B)$*, a random* LRR *set* R_j *is generated from* G *by: (1) picking a random node* $v \in V$*; and (2) generating a sample graph* g *from* G *by* CLE *model and returning* $R_j \leftarrow C_L(g, v)$.

For any $S_A \subset V \setminus S_B$, denote a random variable:

$$Y_j = \begin{cases} 1, \text{ If } R_j \cap S_A \neq \emptyset \\ 0, \text{ Otherwise} \end{cases} \tag{18}$$

and define $\mathbb{L}(S) = n_0 \cdot \mathbb{E}[Y_j]$. Let $\mathcal{R}$ be a set of LRR. We estimate $\mathbb{L}(S)$ as

$$\hat{\mathbb{L}}(S_A) = \frac{n_0}{|\mathcal{R}|} \sum_{R_j \in \mathcal{R}} Y_j \tag{19}$$

Lemma 4 shows the properties of $\mathbb{L}$ function.

Lemma 4. *Given seed set* S_B*, for any set of nodes* $S_A \subseteq V \setminus S_B$*, we have:* $\mathbb{L}(S_A) \leq \mathbb{I}(S_A)$ *and* $\mathbb{L}(\cdot)$ *is a monotone and submodular function.*

The proof of Lemma 4 is omitted here because it is similar to that of Lemma 3. Based on Lemma 4, we use $\mathbb{L}$ as a lower-bound submodular function of $\mathbb{I}$. Algorithm 1 depicts the generation of LRR set. We first randomly select source node v, and then select an edge (u, v) on g_A according to the Competitive live-edge model. If edge (u, v) is selected, we update d_A as the distance from u to v on g_A and check distance condition $d_A(u, v) < d_B(u, S_B)$ by Algorithm 2. In this algorithm, we sequentially generate a simple path from u on g_B (called B-path) according to the CLE model until the length of path exceeds d_A, or no B-edge is selected. If $d_B \leq d_A$ (Lines 13–16, Algorithm 2), it returns *True* and Algorithm 1 returns current LRR set R_j (Line 14, Algorithm 1). In Algorithm 1, if node u is selected into R_j, it moves from v to u and repeats process until distance from current selected node to v on g_A exceeds τ (Line 18, Algorithm 1), or no A-edge is selected (Line 16, Algorithm 1). This process ensures that, if node u is added to the set R_j, the previous nodes on $\mathrm{P}_A(u, v)$ are not affected by S_B.

Algorithm 2: Check the distance from u to B on g_B—CheckBP(u, g_B, d_A).

Input: Graph $G = (V, E, w_A, w_B)$, B-seed set $S_B \subseteq V$, τ, g_B, d_A
Output: g_B and *Check*

Check $\leftarrow$ *False*; $d_B \leftarrow 0$
repeat
  if *exists an B-edge* (u, v) *in* g_B **then**
    $v \leftarrow u$; $d_B \leftarrow d_B + 1$
  else
    Select a B-edge (u, v) using Competitive live-edge model
    if *a edge* (u, v) *is selected* **then**
      $v \leftarrow u$; $d_B = d_B + 1$
      Add B-edge (u, v) into g_B
    else
      break ; `// If no B-edge is selected`
    end
    if $v \in B$ **then**
      Check $\leftarrow$ *True*
      break
    end
  end
until $d_B \geq d_A$;
return g_B and *Check*

5.2. Polling-Based Algorithm for Maximum Bound Functions

We now introduce an approximation algorithm for finding maximum lower and upper function in previous subsection in which all nodes have heterogeneous cost, namely PBA. Our algorithm is based on polling method, which was proposed for IM problem [2,4–6]. We describe our algorithm for maximizing the lower bound function, and it is similar to applying for maximizing the upper bound function. The details of our algorithm are depicted in Algorithm 3.

Algorithm 3: Polling-Based Approximation algorithm (PBA).

Input: Graph $G = (V, E, w_A, w_B)$, budget $L > 0$, and $\epsilon, \delta \in (0, 1)$
Output: A-seed set S_A

$\Lambda = N_{max} \cdot k_{max} \epsilon^2 / n$, $t \leftarrow 1$, $N_{max} \leftarrow N(\epsilon, \frac{\delta}{3}) \frac{\mathsf{OPT}_u}{k_{max}}$, $t_{max} = \left\lceil \log_2 \frac{N_{max}}{\Lambda} \right\rceil$
Generate Λ URR sets by Algorithm 4 and add them into $\mathcal{R}_1$
repeat
  $< S_A, \mathsf{Cov}_{\mathcal{R}_t}(S_A) > \leftarrow \mathsf{Greedy}(\mathcal{R}_t, L)$
  if $\mathsf{CheckQS}(\mathcal{R}_t, \mathsf{Cov}_{\mathcal{R}_t}(S_A), \delta, \epsilon) = True$ *or* $|\mathcal{R}_t| \geq N_{max}$ **then**
    return S_A
  else
    $t \leftarrow t + 1$
    $\mathcal{R}_t \leftarrow \mathsf{CheckQS}(\mathcal{R}_t, \mathsf{Cov}_{\mathcal{R}_t}(S_A), \delta, \epsilon)$
  end
until $|\mathcal{R}_t| \geq N_{max}$;
return S_A;

Algorithm 4: Generate URR set.

Input: Graph $G = (V, E, w_A, w_B)$, B-seed set $S_B \subseteq V$, τ
Output: a URR set R_j

1. Select randomly a node $v \in V \setminus S_B$
2. $R_j \leftarrow v$; $d_A \leftarrow 0$
3. **while** $d_A \leq \tau$ **do**
4. Select an A-edge (u, v) by using Competitive live-edge model
5. **if** *An edge* (u, v) *is selected* **then**
6. $v \leftarrow u$; $d_A \leftarrow d_A + 1$
7. If ($v \in R_j$ or $v \in S_B$) then break
8. Add v into R_j
9. **else**
10. break
11. **end**
12. **end**
13. **return** R_j;

5.2.1. Description of PBA

PBA first generates a collection $\mathcal{R}_1$ of Λ URR sets. The main phrase of PBA contains several iterators (at most t_{max}). In each iterator, the algorithm first finds the candidate solution S_A by using Greedy algorithm (Algorithm 5) to solve Budgeted Maximum Coverage (BMC) problem [20] (Line 6). It provides an approximation ratio of $(1 - \frac{1}{\sqrt{e}})$. We denote $\mathsf{Greedy}(\mathcal{R}, L)$ as Greedy algorithm with the input data consisting a collection URR sets $\mathcal{R}$ and budget $L > 0$. Then, S_A is checked for quality in Algorithm 6, which independently generates more $|\mathcal{R}_t|$ URR, adds them to $\mathcal{R}_c$, calculates

$$\mathsf{Cov}_{\mathcal{R}_c}(S_A) = \sum_{R_j \in \mathcal{R}_c} \min\{|S_A \cap R_j|, 1\} \tag{20}$$

Algorithm 5: Greedy algorithm for Budgeted Maximum Coverage problem—$\mathsf{Greedy}(\mathcal{R}, L)$.

Input: URR set $\mathcal{R}$, budget k
Output: Seed set S_A

1. $S \leftarrow \emptyset$
2. $U \leftarrow \bigcup_{R_j \in \mathcal{R}} R_j$
3. **repeat**
4. $v \leftarrow \arg\max_{u \in U \setminus S} \left(\mathsf{Cov}_{\mathcal{R}}(S \cup \{u\}) - \mathsf{Cov}_{\mathcal{R}(S)}\right) / c(u)$
5. **if** $c(S \cup \{u\}) \leq L$ **then**
6. $S \leftarrow S \cup \{u\}$
7. **end**
8. $U \leftarrow U \setminus u$
9. **until** $U = \emptyset$;
10. $v_{max} \leftarrow \arg\max_{u \in U | c(u) \leq L} \mathsf{Cov}_{\mathcal{R}}(u)$
11. $S_1 \leftarrow \arg\max_{S' \in \{S, v_{max}\}} \mathsf{Cov}_{\mathcal{R}}(S')$
12. **return** S_1;

and uses it to calculate parameters ϵ_1, ϵ_2. We next calculate the lower-bound of $\mathbb{I}(S_A)$: $f_l(S, \mathcal{R}_c, \epsilon_1)$, and upper-bound of optimal solution $f_u(\mathsf{OPT}_u, \mathcal{R}_t, \epsilon_2)$. If current solution S_A meets approximation guarantee condition that

$$\frac{f_l(S_A, \mathcal{R}_c, \epsilon_1)}{f_u(\mathsf{OPT}_u, \mathcal{R}_t, \epsilon_2)} \geq 1 - \frac{1}{\sqrt{e}} - \epsilon \tag{21}$$

the algorithm returns S_A. If not, it moves to the next iterator and stops when the number of URR sets is at least N_{max}.

Algorithm 6: Check quality of solution (CheckQS).

Input: $\mathcal{R}_t, \mathsf{Cov}_{\mathcal{R}_t}(S_A), \delta, \epsilon$
Output: True or (False and $\mathcal{R}_{t+1}$)

$\delta_1 = \frac{\delta}{3t_{max}}, \epsilon_1 = \epsilon,$
$cov \leftarrow \mathsf{Cov}_{\mathcal{R}_t}(S_A)$
$\mathcal{R}_c \leftarrow \mathcal{R}_t$
for $i = 1$ *to* $|\mathcal{R}_t|$ **do**
  Generate a URR set R_j by Algorithm 4 and add it to $\mathcal{R}_c$
  If $R_j \cap S_A \neq 0$, then $cov \leftarrow cov + 1$
end
$a \leftarrow \frac{cov-1}{\ln(2\delta_1^{-1})}$
If $a = \frac{2}{3}$ **return** False and $\mathcal{R}_c$
$\epsilon_1^1 \leftarrow \frac{1}{a-\frac{2}{3}}\left(\frac{4}{3} - \sqrt{2a + \frac{4}{9}}\right), \epsilon_1^2 \leftarrow \frac{1}{a-\frac{2}{3}}\left(\frac{4}{3} + \sqrt{2a + \frac{4}{9}}\right)$
$\epsilon_1 \leftarrow \min\{\epsilon' \in \{\epsilon_1^1, \epsilon_1^2\} | \epsilon' > 0\}$
$\epsilon_2 \leftarrow \sqrt{\frac{2(1+\epsilon_1)|\mathcal{R}_c| \ln \frac{3t_{max}}{\delta_1}}{|\mathcal{R}_t| \mathbb{U}_c(S)}}$
$f_l(S_A, \mathcal{R}_c, \epsilon_1) \leftarrow \frac{n_0}{(1+\epsilon_1)|\mathcal{R}_c|}\mathsf{Cov}_{\mathcal{R}_c}(S_A)$
$f_u(S_A, \mathcal{R}_t, \epsilon_2) \leftarrow \frac{n_0 \mathsf{Cov}_{\mathcal{R}_t}(S_A)}{|\mathcal{R}_t|(1-\sqrt{e})(1-\epsilon_2)}$
if $\frac{f_l(S_A)}{f_u(\mathsf{OPT}_u)} \geq 1 - \frac{1}{\sqrt{e}} - \epsilon$ **then**
  return True
else
  return False and $\mathcal{R}_c$
end
return False;

5.2.2. Theoretical Analysis

Now, we prove that PBA returns a $(1 - 1/\sqrt{e} - \epsilon)$-approximation solution with probability at least $1 - \delta$.

We observe that $X_j \in [0, 1]$. Let random variable $Z_i = \sum_{j=1}^{i}(X_j - \mathbb{E}[X_j]), \forall i \geq 1$. For a sequence random variables $Z_1, Z_2, \ldots$, we have $\mathbb{E}[Z_i | Z_1, \ldots, Z_{j-1}] = E[Z_{i-1}] + \mathbb{E}[X_i - \mu_X] = \mathbb{E}[Z_{i-1}]$. Hence, $Z_1, Z_2, \ldots$ is a form of martingale [34]. Therefore, we obtain the same results as in [6].

Lemma 5 ([6]). *For any $T > 0, \epsilon > 0$, μ is the mean of X_j, and an estimation of μ is $\hat{\mu} = \frac{\sum_{i=1}^{T} X_i}{T}$. We have:*

$$\Pr[\hat{\mu} \geq (1+\epsilon)\mu] \leq \exp\left(\frac{-T\mu\epsilon^2}{2+\frac{2}{3}\epsilon}\right) \tag{22}$$

$$\Pr[\hat{\mu} \leq (1-\epsilon)\mu] \leq \exp\left(\frac{-T\mu\epsilon^2}{2}\right) \tag{23}$$

Based on Lemma 5, Tang et al. [6] proposed IMM algorithm based on *Reverse Influence Set (RIS)* [24] process for solving IM problem. They showed that the number of random *Reachable Reversal (RR)* sets, which ensures RIS process returns an $(1-1/e)$-approximation with probability $1-\delta$, is

$$\theta_{max} = 2n\left(\left(1-\frac{1}{e}\right)\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\left(1-\frac{1}{e}\right)\left(\ln\frac{2}{\delta} + \ln\binom{n}{k}\right)}\right)^2 \frac{1}{\epsilon^2 k} \tag{24}$$

This threshold is also used to obtain stopping condition for IM algorithms [4,7]. However, it does not guarantee that the candidate solution S_A is a $(1-1/\sqrt{e})$-approximation under the heterogeneous selecting costs. In this case, we provide the number of URR sets to guarantee $(1-1/\sqrt{e})$-approximation ratio by the following theorem.

Theorem 4. *For $\epsilon > 0$, $\delta \in (0,1)$ is the parameter.If $|\mathcal{R}|$ is greater or equal to*

$$N(\epsilon,\delta) = 2n_0\left((1-\frac{1}{\sqrt{e}})\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\left(1-\frac{1}{\sqrt{e}}\right)\left(\ln\frac{2}{\delta} + \ln\binom{n}{k_{max}}\right)}\right)^2 \frac{1}{\epsilon^2 \mathsf{OPT}_u}$$

Algorithm 5 returns a $(1-1/\sqrt{e}-\epsilon)$-approximation solution with probability at least $1-\delta$.

Lemma 6. *Let event $B_t^1 = \left(|\mathcal{R}_t| \geq N_{max}\right) \wedge \left(\mathbb{U}(S) < (1-1/\sqrt{e}-\epsilon)\mathsf{OPT}_u\right)$ then, $\Pr[B_t^1] < \frac{\delta}{3}$.*

Proof. From Theorem 4, since $N_{max} \geq N\left(\epsilon, \frac{\delta}{3}\right)$, the bad event $\mathbb{U}(S) < (1-1/\sqrt{e}-\epsilon)\mathsf{OPT}_u$ happens with probability at most $\frac{\delta}{3}$ □

Lemma 7. *For each $1 \leq t \leq t_{max}$, let $f_l(S_A, \mathcal{R}_c, \epsilon_1) = \frac{n_0}{|\mathcal{R}_c|(1+\epsilon_1)}\mathsf{Cov}_{\mathcal{R}_c}(S_A)$. We have: $\Pr[f_l(S_A, \mathcal{R}_c, \epsilon_1) \leq \mathbb{U}(S_A)] \geq 1-\delta_1$*

Proof. We denote $\hat{\mathbb{U}}_c(S_A)$ as an estimation of $\mathbb{U}(S_A)$ over $\mathcal{R}_c$. In each iterator t, after ending the for loop (Line 7) in Algorithm 6, we have

$$\mathsf{Cov}_{\mathcal{R}_c}(S_A) = \Upsilon(\epsilon_1,\delta_1) = (1+\epsilon_1)\left(2+\frac{2}{3}\epsilon_1\right)\ln\frac{2}{\delta_1}\frac{1}{\epsilon_1^2}$$

It satisfies the stopping rule theorem in [35], therefore it guarantees that

$$\Pr[\hat{\mathbb{U}}_c(S_A) \leq (1+\epsilon_1)\mathbb{U}(S_A)] = \Pr[\hat{\mu}_c \leq (1+\epsilon_1)\mu] \geq 1-\delta_1$$

Hence, $\Pr[f_l(S_A) \leq \mathbb{U}(S_A)] \geq 1-\delta_1$ □

Lemma 8. *Assume that the bad event in Lemma 7 does not happen. Let $f_u(S_A, \mathcal{R}_t, \epsilon_2) = \frac{n_0\mathsf{Cov}_{\mathcal{R}_t}(S_A)}{|\mathcal{R}_t|(1-\sqrt{e})(1-\epsilon_2)}$. We have: $\Pr[f_u(S_A, \mathcal{R}_t, \epsilon_2) \geq \mathbb{U}(S_U^*)] \geq 1-\delta_1$.*

Proof. Since the bad event in Lemma 7 does not happen, we have $\mu(S_A)(1+\epsilon_1) \geq \hat{\mu}_c(S_A)$. Applying Lemma 5 for optimal set S_U^*, with random variable X_j, the mean $\mu(S_U^*) = \mathbb{U}(S_U^*)/n_0$, and $|\mathcal{R}_t|$ samples

$$\begin{aligned}\Pr[\hat{\mathbb{U}}_t(S_U^*) < (1-\epsilon_2)\mathbb{U}(S_U^*)] &= \Pr[\hat{\mu}_t(S_U^*) < (1-\epsilon_2)\mu(S_U^*)] \\ &\leq \exp\left(-\frac{|\mathcal{R}_t|\epsilon_2^2\mu(S_U^*)}{2}\right) \leq \exp\left(-\frac{|\mathcal{R}_t|\epsilon_2^2\mu(S_A)}{2}\right) \\ &= \exp\left(-\frac{|\mathcal{R}_t|\epsilon_2^2\hat{\mu}_c(S_A)/(1+\epsilon_1)}{2}\right) = \frac{\delta}{3t_{max}} = \delta_1\end{aligned}$$

Now, since Greedy algorithm returns a $(1-1/\sqrt{e})$-approximation solution, we have:

$$\begin{aligned}\hat{\mathbb{U}}_t(S_A) &= \frac{n_0}{|\mathcal{R}_t|}\mathsf{Cov}_{\mathcal{R}_t}(S_A) \geq \frac{n_0}{|\mathcal{R}_t|}(1-1/\sqrt{e})\mathsf{Cov}_{\mathcal{R}_t}(S_t^*) \\ &\geq \frac{n_0}{|\mathcal{R}_t|}(1-1/\sqrt{e})\mathsf{Cov}_{\mathcal{R}_t}(S_U^*) = (1-1/\sqrt{e})\hat{\mathbb{U}}_t(S_U^*) \\ &\geq (1-1/\sqrt{e})(1-\epsilon_2)\mathbb{U}(S_U^*)\end{aligned}$$

where S_t^* is an optimal solution of instance $(\mathcal{R}_t, L)$ of maximum coverage problem. Therefore, $f_u(S_A, \mathcal{R}_t, \epsilon_2) = \frac{\hat{\mathbb{U}}(S)}{(1-1/\sqrt{e})(1-\epsilon_2)} \geq \mathbb{U}(S_U^*)$ with probability at least $1-\delta_1$. □

Theorem 5. *Given* $0 \leq \epsilon, \delta \leq 1$, PBA *algorithm returns the set node S satisfying:*

$$\Pr[\mathbb{U}(S_A) \geq (1-1/\sqrt{e}-\epsilon)\mathbb{U}(S_U^*)] \geq 1-\delta \tag{25}$$

Proof. Assume that none of the bad events in Lemmas 6–8 happen in any iterator $t = 1, 2 \ldots, t_{max}$. We apply the union bounding the probability of bad events, and the probability of this assumption is at least

$$1 - \left(\frac{\delta}{3} + \delta_1 \cdot t_{max} + \delta_1 \cdot t_{max}\right) = 1-\delta \tag{26}$$

Under this assumption, we show that PBA algorithm returns a solution satisfying:

$$\mathbb{U}(S_A) \geq (1-1/\sqrt{e}-\epsilon)\mathbb{U}(S_U^*) \tag{27}$$

If the algorithm stops with condition $|\mathcal{R}_t| \geq N_{max}$, the solution S satisfies Equation (27) due to Lemma 6. Otherwise, PBA algorithm stops at some iterator $t, t = 1, 2, \ldots, t_{max}$, in which the CheckQS on Line 5 returns "True". Since the bad events do not happen and the condition on Line 12 of Algorithm 6 is true, we have

$$\frac{\mathbb{U}(S_A)}{\mathsf{OPT}_u} \geq \frac{f_l(S_A, \mathcal{R}_c, \epsilon_1)}{f_u(S_A, \mathcal{R}_t, \epsilon_2)} \geq 1 - \frac{1}{\sqrt{e}} - \epsilon \tag{28}$$

This completes the proof. □

5.2.3. Improved Guarantees with Tightened Bound

Lemma 8 provides an upper bound of OPT_u, in which we use the inequality $\mathsf{Cov}_{\mathcal{R}_t}(S_A) \geq (1-1/\sqrt{e})\mathsf{Cov}_{\mathcal{R}_t}(S_t^*)$. However, this upper bound is tight in the worst case [20], but loose for specific instances of budgeted maximum coverage problem. We propose another upper bound of $\mathsf{Cov}_{\mathcal{R}_t}(S_t^*)$ that is much tighter in practice, as explained in the following. In Greedy algorithm, we denote S_i at iterator i. The following lemma provides a tighter bound of $\mathsf{Cov}_{\mathcal{R}_t}(S_t^*)$.

Lemma 9. *Assume that the number iterators in this algorithm is k and u_i is the node added at ith iterator. Letting* $g(\mathcal{R}_t, S_k) = \frac{L \cdot \mathsf{Cov}_{\mathcal{R}_t}(S_k)}{c(u_k)} + \left(1 - \frac{L}{c(u_k)}\right) \mathsf{Cov}_{\mathcal{R}_t}(S_{k-1})$, *we have*

$$\mathsf{Cov}_{\mathcal{R}_t}(S_t^*) \leq g(\mathcal{R}_t, S_k) \leq \frac{\mathsf{Cov}_{\mathcal{R}_t}(S)}{1 - 1/\sqrt{e}} \tag{29}$$

Proof. From Lemma 1 in [20], we have

$$\mathsf{Cov}_{\mathcal{R}_t}(S_k) - \mathsf{Cov}_{\mathcal{R}_t}(S_{k-1}) \geq \frac{c(u_k)}{L}(\mathsf{Cov}_{\mathcal{R}_t}(S_t^*) - \mathsf{Cov}_{\mathcal{R}_t}(S_{k-1})) \tag{30}$$

Rearranging Equation (30) yields $\mathsf{Cov}_{\mathcal{R}_t}(S_t^*) \leq g(\mathcal{R}_t, S_k)$. On the other hand,

$$\begin{aligned} g(\mathcal{R}_t, S_k) - \mathsf{Cov}_{\mathcal{R}_t}(S_k) &= \left(1 - \tfrac{c(u_k)}{L}\right)(g(\mathcal{R}_t, S_k) - \mathsf{Cov}_{\mathcal{R}_t}(S_{k-1})) \\ &= \textstyle\prod_{i=1}^{k} \left(1 - \tfrac{c(u_i)}{L}\right) g(\mathcal{R}_t, S_k) \end{aligned} \tag{31}$$

It follows that

$$g(\mathcal{R}_t, S_k) = \frac{\mathsf{Cov}_{\mathcal{R}_t}(S_k)}{1 - \prod_{i=1}^{k} \left(1 - \frac{c(u_i)}{L}\right)} \leq \frac{\mathsf{Cov}_{\mathcal{R}_t}(S_k)}{1 - 1/\sqrt{e}} \tag{32}$$

□

By Lemma 9, we have the new upper bound OPT_u as follows:

$$f_u^a(S_A, \mathcal{R}_t, \epsilon_2) = \frac{n_0}{|\mathcal{R}_t|(1 - \epsilon_2)} g(\mathcal{R}_t, S_k) \tag{33}$$

5.3. Sandwich Approximation

We apply Sandwich Approximation framework in [16] to design our algorithm, namely SA-PBA. Let S_U and S_L be solutions selected by PBA algorithm for maximizing $\mathbb{L}$ and $\mathbb{U}$ within the total cost at most L, respectively. S'_A is a solution for original problem. We denote $\hat{\mathbb{I}}(S_A)$ as a (δ', ϵ')-approximation of $\mathbb{I}(S_A)$, i.e.,

$$\Pr[(1 - \epsilon')\mathbb{I}(S_A) \leq \hat{\mathbb{I}}(S_A) \leq (1 + \epsilon')\mathbb{I}(S_A)] \geq 1 - \delta' \tag{34}$$

The sandwich approximation algorithm operates as follows. First, we find a solution to the original problem with any strategy. Second, we find an approximate solution to the lower bound and the upper bound by PBA algorithm. Last, we return $S_{sa} = \arg\max_{S \in \{S_L, S', S_U\}} \hat{\mathbb{I}}(S)$ as the solution of SPBA algorithm. The details of our algorithm are shown in Algorithm 7.

Algorithm 7: Sandwich Approximation base on PBA algorithm (SPBA).

Input: Graph $G = (V, E, w_A, w_B)$, budget $L > 0$, and $\epsilon, \delta, \epsilon', \delta' \in (0, 1)$
Output: Seed set S_A

1. $S_U \leftarrow \mathsf{PBA}(\mathbb{L}, G, L, \epsilon, \delta)$
2. $S_L \leftarrow \mathsf{PBA}(\mathbb{U}, G, L, \epsilon, \delta)$
3. $S' \leftarrow$ a solution for maximizing $\mathbb{I}$ by any algorithm.
4. $S \leftarrow \arg\max_{S \in \{S_U, S_L, S'\}} \hat{\mathbb{I}}(S)$
5. **return** S;

The following theorem shows the approximation ratio of our algorithm.

Theorem 6. *Let S^* be the optimal solution, and S_{sa} be a solution returned by Algorithm 7. We have:*

$$\mathbb{I}(S_{sa}) \geq \max\left\{\frac{\mathbb{I}(S_U)}{\mathbb{U}(S_U)}, \frac{\mathbb{L}(S_L^*)}{\mathbb{I}(S_A^*)}\right\}\frac{(1-\epsilon')}{(1+\epsilon')}\left(1-\frac{1}{\sqrt{e}}-\epsilon\right)\mathsf{OPT} \tag{35}$$

with probability at least $1-2\delta-\delta'$.

Proof. Due to $\mathbb{U}(S_U^*) \geq \mathbb{U}(S_A^*) \geq \mathbb{I}(S_A^*)$, we have:

$$\begin{aligned}\hat{\mathbb{I}}(S_U) &= \frac{\hat{\mathbb{I}}(S_U)}{\mathbb{U}(S_U)}\mathbb{U}(S_U) \geq \frac{\mathbb{I}(S_U)}{\mathbb{U}(S_U)}\left(1-\frac{1}{\sqrt{e}}-\epsilon\right)\mathbb{U}(S_U^*)\\ &\geq \frac{\mathbb{I}(S_U)}{\mathbb{U}(S_U)}(1-\epsilon')\left(1-\frac{1}{\sqrt{e}}-\epsilon\right)\mathbb{I}(S_A^*)\end{aligned} \tag{36}$$

On the other hand,

$$\begin{aligned}\hat{\mathbb{I}}(S_L) &\geq (1-\epsilon')\mathbb{I}(S_L) \geq (1-\epsilon')\mathbb{L}(S_L)\\ &\geq \left(1-\frac{1}{\sqrt{e}}-\epsilon\right)\mathbb{L}(S_L^*)\\ &\geq \frac{\mathbb{L}(S_L^*)}{\mathbb{I}(S_A^*)}(1-\epsilon')\left(1-\frac{1}{\sqrt{e}}-\epsilon\right)\mathbb{I}(S_A^*)\end{aligned} \tag{37}$$

Since $(1-\epsilon')\mathbb{I}(S_{sa}) \leq \hat{\mathbb{I}}(S_{sa}) \leq (1+\epsilon')\mathbb{I}(S_{sa})$, we have:

$$\mathbb{I}(S_{sa}) \geq \frac{1}{1+\epsilon'}\hat{\mathbb{I}}(S_{sa}) \geq \max\left\{\frac{\mathbb{I}(S_U)}{\mathbb{U}(S_U)}, \frac{\mathbb{L}(S_L^*)}{\mathbb{I}(S_A^*)}\right\}\cdot\frac{(1-\epsilon')}{(1+\epsilon')}(1-\frac{1}{\sqrt{e}}-\epsilon)\cdot\mathsf{OPT}_u$$

Applying union bound of probabilities, the inequality in Equation (38) happens with probability at least $1-2\delta-\delta'$. □

6. Experiments

We experimentally evaluated and compared the performance of our algorithm to other algorithms, namely baseline algorithms and influence maximization methods, on two aspects: solution quality and the scalability from various network datasets.

6.1. Experimental Settings

6.1.1. Datasets

We performed our experiments on six real-world datasets: Gnutella, Enron, Epinions, Email-Eu, DBLP and Wiki.The basic statistics of these networks are summarized in Table 2.

Table 2. Datasets.

Dataset	Nodes	Edges	Type	Avg. Degree
Gnutella [36]	6301	20,777	Directed	3.29
Enron [37]	36,692	183,831	Undirected	5.01
Epinions [38]	75,879	508,837	Directed	6.7
Email-Eu [36]	265,214	420,045	Directed	1.58
DBLP [39]	317,080	1,049,866	Undirected	5.01
Wiki [40]	1,791,489	28,511,807	Directed	6.7

6.1.2. Algorithm Compared

We compared our algorithm with influence maximization BCT algorithm and several baseline algorithms, which are described as follows

- BCT: An influence maximization algorithm under the heterogeneous selecting cost. The reason we chose BCT to compare is that BCIM is a variant of IM and BCT considers of nodes with arbitrary costs.
- Degree: This algorithm selects nodes with the highest degree and we keep on adding the highest-degree nodes until total costs of the selection of nodes exceeds L.
- Random: This algorithm randomly selects nodes within budget L.

6.1.3. Parameters

In all the experiments, we kept $\epsilon = 0.1$ and $\delta = 1/n$ as general settings. We set $\epsilon' = \delta' = 0.01$ and used the stopping condition algorithm in [35] to estimate $\hat{\mathbb{I}}$. We assigned the weights of edges in TCLT model according to LT model in previous studies [4–8,13]. The weight of the edge (u, v) was calculated as follows,

$$w(u, v) = \frac{1}{|N_{-}(v)|} \tag{38}$$

Our implementation was written in C++ and compiled with GCC 4.7. All our experiments were carried out using a Linux machine with a 2 × Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30 GHz 8 × 16 GB DIMM ECC DDR4 @ 2400 MHz

6.2. Results

6.2.1. Comparison of Algorithms under General Case

In this experiment, we compared algorithms when $\tau = 5$, the budget L varied from 0 to 100 and the costs of node were uniformly distributed in $[1.0, 3.0]$. Figure 6 shows the performance of all algorithms on Gnutella, Enron, Epinions, Email-Eu, DBLP and Wiki networks. On average, we observed that our algorithm SPBA always had the best performance. SPBA was 10–30% better than the result of BCT. This is because BCT only considers the number of influenced nodes and ignores the competitive influence under time constraint τ. It also confirmed that IM's algorithms do not have good performance for BCIM problem. Random algorithm had the worst performance of all cases. Degree algorithm performed well on the Enron dataset but had bad performance on the other datasets. SPBA was up to 7.7 times better than Degree. The reason is that Degree only uses the topology properties of the social network but can not consider competitive diffusion process. In the opposite, our algorithm takes advantage of the upper and lower bounds of objective function to obtain the approximation ratio. This explains why our algorithm had good performance while the others had poor performance in many cases.

6.2.2. Comparison of Algorithms under Unit-Cost Setting

To show more clearly the performance of these algorithms, we conducted experiments on the unit-cost case (i.e., all node costs are equal to 1) on Gnutella, Enron, Epinions and Email-EU datasets. We set $\tau = 5$ and L varied from 1 to 100. Figure 7 displays the results of all algorithms. Once again, we found that our algorithm SPBA gave the best performance. SPBA was 1.06–1.76 times better than BCT and 1.2–17.2 times better than the result of Degree. These results are also consistent with what was observed in the previous case.

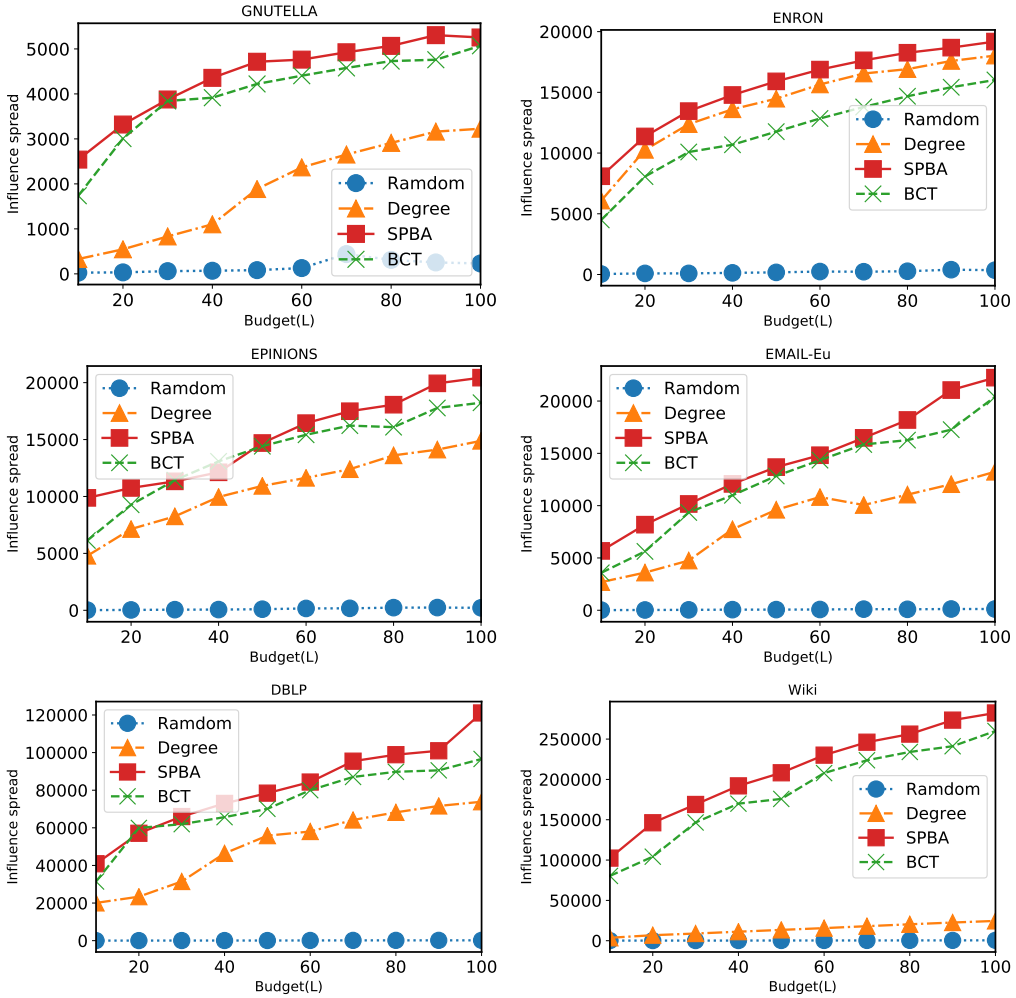

Figure 6. Comparison between different algorithms under general cost setting.

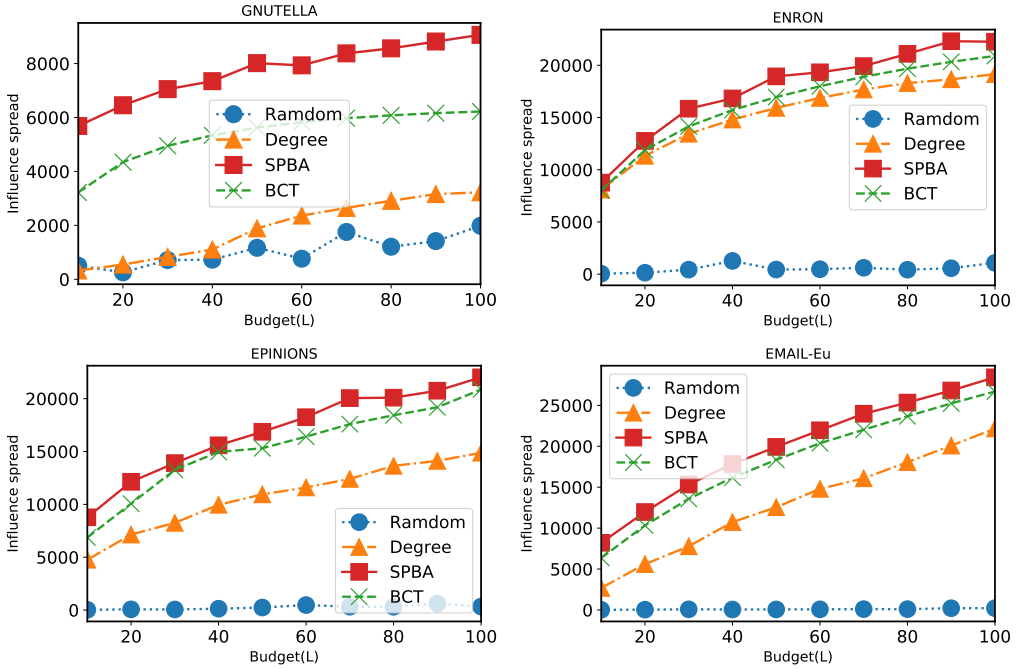

Figure 7. Comparison between different algorithms under unit-cost setting.

6.2.3. Comparison of Running Time

Figure 8 shows running time of algorithms on six datasets. SPBA had the longest running time on any datasets. This is because the running time of SPBA consists of total running time of $\mathrm{PBA}(\mathbb{L}, G, L, \epsilon, \delta)$, $\mathrm{PBA}(\mathbb{U}, G, L, \epsilon, \delta)$ and calculating $\hat{\mathbb{I}}(\cdot)$. Random and Degree algorithms are simple heuristic algorithms thus their costs are low. This resulted in their shortest running time. Although BCT is based on polling method, it ran faster than our algorithm. The reason for this result is due to the following reasons. Firstly, the sampling process of BCT and our algorithm are different. The sampling complexity of BCT is mainly dependent on the number of randomly selected node while the sampling process in our algorithm is more complicated because it must check the influence paths from S_B. Secondly, to obtain a data approximation, our algorithm must solve three problems with polling based method. It is worth noting that SPBA is scalable with million-scale networks. For Wiki network, which has 1.79 millions nodes and 28.5 millions edges, our algorithm finished in 90 s.

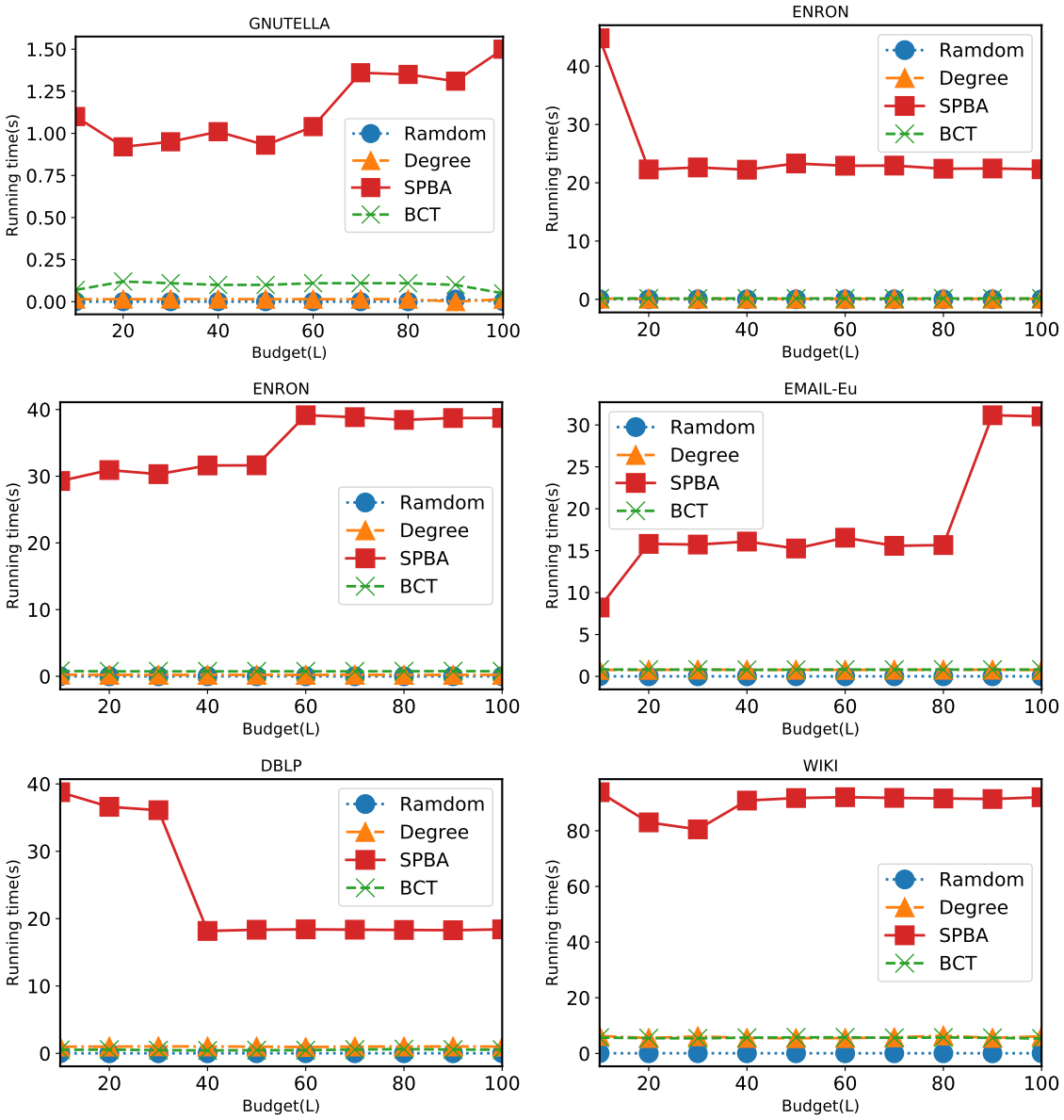

Figure 8. Running time of algorithms for BCIM problem.

6.2.4. Impact of τ

Considering the importance of early competitive influence in viral marketing, we were very interested in the role of time constraint in influence. We compared our solution with three other algorithms while varying τ from 3 to 5. Figure 9 shows results of algorithms when $L = 50$. SPBA was clearly still the best performer. Specifically, our SPBA was 1.01–1.23 times better than BCT and up to 2.5 times better than Degree.

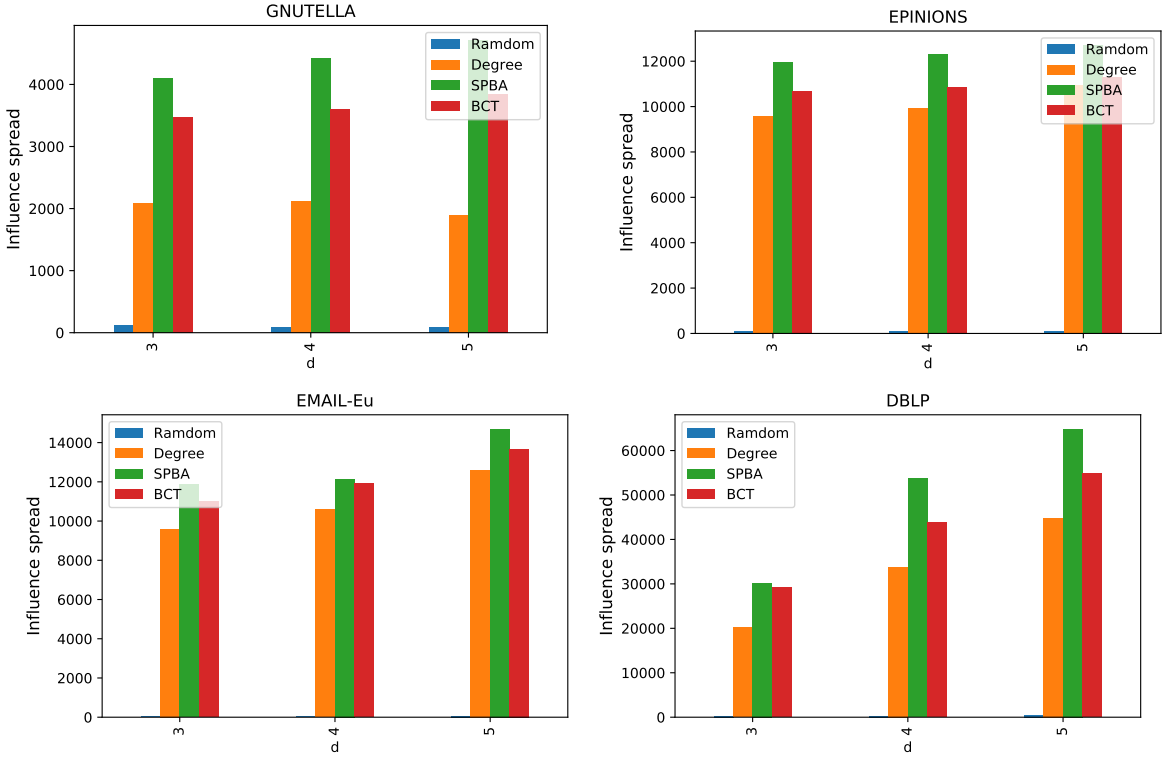

Figure 9. Comparison between different algorithms when τ varies.

7. Conclusions

In this paper, we investigate BCIM problem, which finds the seed set of a player to maximize their influence while their competitors are conducting similar strategies. We first propose TCLT model to capture the competitive influence of two competitors on a social network and formulate BCIM in this model. We provide the hardness results and properties of objective function. A randomized SPBA-based approximation is proposed for finding the solution of BCIM. Experiments on real world social networks were conducted. The results show that our proposed algorithm outperformed the other heuristics.

Author Contributions: Investigation, C.V.P. and H.V.D.; Methodology, C.V.P.; Supervision, H.X.H. and M.T.T.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Proof of Theorem 3. We use the inductive method according to two models when $t = 0, 1, 2..., \tau$. In the step $t = 0$, we have $A_0 = A'_0$ and $B_0 = B'_0$. Assume that, at step $t \geq 0$, the two models give the same distribution of A-active set and B-active set nodes, which are A_t and B_t. For TCLT model, we consider a node v that has not been activated by the end of step t, i.e., $v \notin A_t \cup B_t$. We divide the state of v in step $t + 1$ into two cases:

Cases 1: Total A-influence weights exceed threshold $\theta_A(v)$ while the total B-influence weights is smaller than $\theta_B(v)$ in step $t+1$. The probability that this case happens is:

$$P_{1,t+1}(v) = \frac{\left(\sum_{u\in A_t\setminus A_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B_t\setminus B_{t-1}} w_B(u,v)\right)}{\left(1-\sum_{u\in A_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B_{t-1}} w_B(u,v)\right)} \tag{A1}$$

Cases 2: Total A-influence weights exceed threshold $\theta_A(v)$ while the total negative influence weights also exceed than threshold $\theta_B(v)$. The probability that this case happens is:

$$P_{2,t+1}(v) = \frac{\left(\sum_{u\in A_t\setminus A_{t-1}} w_A(u,v)\right)\left(\sum_{u\in B_t\setminus B_{t-1}} w_B(u,v)\right)}{\left(1-\sum_{u\in A_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B_{t-1}} w_B(u,v)\right)} \tag{A2}$$

In this case, TB-WPP rule is used to determine state of v. According to this rule, the probability node v is A-activated at step $t+1$ is equal to

$$P_{1,t+1}(v) + p_A(v|A_{t-1}, B_{t-1}) \cdot P_{2,t}(v) \tag{A3}$$

On CLE model, we assume that A'_t and B'_t are the set of A-active set and B-active set nodes at step t. For $v \notin B'_t \cup A'_t$, the probability that v has an in-edge from A'_t and does not have an in-edge from B'_t (called $P'_{1,t}(v)$) is equal to

$$P'_{1,t}(v) = \frac{\left(\sum_{u\in A'_t\setminus A'_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B_t\setminus B'_{t-1}} w_B(u,v)\right)}{\left(1-\sum_{u\in A'_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B'_{t-1}} w_B(u,v)\right)} = P_{1,t}(v) \tag{A4}$$

We denote $P'_{2,t}(v)$ as the probability v has both in-edge from A'_t and in-edge from B'_t. We have

$$P'_{2,t}(v) = \frac{\left(\sum_{u\in A'_t\setminus A'_{t-1}} w_A(u,v)\right)\left(\sum_{u\in B'_t\setminus B'_{t-1}} w_B(u,v)\right)}{\left(1-\sum_{uA'_{t-1}} w_A(u,v)\right)\left(1-\sum_{u\in B'_{t-1}} w_B(u,v)\right)} = P_{2,t}(v) \tag{A5}$$

According to CLE model, the probability v is A-activated in this case is $P'_{2,t}(v) \cdot p_A(v|A'_{t-1}, B'_{t-1})$. Thus, the probability v is A-activated at step $t+1$ is

$$P'_{1,t}(v) + P'_{2,t}(v) \cdot p_A(v|A'_{t-1}, B'_{t-1}) = P_{1,t}(v) + P_{2,t}(v) \cdot p_A(v|A_{t-1}, B_{t-1}) \tag{A6}$$

Due to $A_0 = A'_0 = A$, by step-by-step induction, we reach the conclusion that the random CLE model producing the same distribution over A-active sets as the TCLT model at any hop $t = 0, 1, \ldots, \tau$. Similarly, we obtain two models produce the same distribution over B-active sets. □

Proof of Theorem 4. We first show that, for any parameters $\epsilon_1 > 0$ and $\theta_1 \in (0,1)$: If

$$|R| \geq \theta_1 = \frac{2n_0 \ln(1/\delta_1)}{\mathsf{OPT}_u \epsilon_1^2} \tag{A7}$$

then

$$\hat{\mathbb{U}}(S^*_U) \geq (1-\epsilon_1) \cdot \mathsf{OPT}_u \tag{A8}$$

with probability at least $1-\delta_1$. Indeed, applying Lemma 5, we obtain

$$\begin{aligned} \Pr[\hat{\mathbb{U}}(S^*_U) \leq (1-\epsilon_1)\mathbb{U}(S^*_U)] &= \Pr\left[\tfrac{n_0}{T}\sum_{j=1}^{T} X_j \leq (1-\epsilon_1)\mu \cdot n_0\right] \\ &= \Pr\left[\hat{\mu} \leq (1-\epsilon_1)\mu\right] \leq \exp\left(-\tfrac{\epsilon_1^2 \mu T}{2}\right) = \delta_1 \end{aligned} \tag{A9}$$

For the instance $(\mathcal{R}, L)$, we denote S as solution Greedy, and S_0^* as an optimal solution. Since Algorithm 5 returns a $(1-1/\sqrt{e})$-approximation solution, the following event happens with probability at least $1-\delta_1$:

$$\hat{\mathbb{U}}(S) \geq (1-1/\sqrt{e})\hat{\mathbb{U}}(S_0^*) \geq (1-1/\sqrt{e})\hat{\mathbb{U}}(S_U^*) \geq (1-1/\sqrt{e})(1-\epsilon_1)\hat{\mathbb{U}}(S_U^*) \tag{A10}$$

We next show that for $\epsilon_2 > 0, \epsilon_2 = \epsilon - (1-1/\sqrt{e})\epsilon_1$ and $\delta_2 \in (0,1)$. If Equation (A8) holds, and

$$T \geq \theta_2 = \frac{2(1-1/\sqrt{e})\ln(\binom{n_0}{k_{max}}/\delta_2)}{\mathsf{OPT}_l(\epsilon - \epsilon_1(1-1/\sqrt{e}))^2} \tag{A11}$$

the following inequality holds with probability at least $1-\delta_2$

$$\hat{\mathbb{U}}(S) \geq (1-1/\sqrt{e}-\epsilon)\cdot \mathsf{OPT}_l \tag{A12}$$

Since Equation (A8) holds, we have

$$\hat{\mathbb{U}}(S) \geq (1-1/\sqrt{e})(1-\epsilon_1)\mathbb{U}(S_U^*) = (1-1/\sqrt{e}-\epsilon)\mathsf{OPT}_u + \epsilon_2\mathsf{OPT}_u \tag{A13}$$

Applying Equation (A13) and combining with Lemma 5, we have:

$$\begin{aligned}
\Pr[\mathbb{U}(S) \leq (1-1/\sqrt{e}-\epsilon)\cdot \mathsf{OPT}_u] &\leq \Pr[\hat{\mathbb{U}}(S) - \mathbb{U}(S) \geq \epsilon_2 \cdot \mathsf{OPT}_u] \\
&= \Pr\left[\tfrac{n_0}{T}\textstyle\sum_{i=1}^{T} Y_i - n_0\mu \geq \epsilon_2 \cdot \mathsf{OPT}_u\right] \\
&= \Pr\left[\tfrac{1}{T}\textstyle\sum_{i=1}^{T} Y_i - \mu \geq \left(\frac{\mathsf{OPT}_u\epsilon_2}{n_0\mu}\right)\cdot\mu\right] \\
&= \Pr\left[\hat{\mu} - \mu \geq \left(\frac{\mathsf{OPT}_l\epsilon_2}{n_0\mu}\right)\cdot\mu\right] \\
&\leq \exp\left(-\frac{\left(\frac{\mathsf{OPT}_u\epsilon_2}{n_0\mu}\right)^2 T\mu}{2+3\cdot\left(\frac{\mathsf{OPT}_u\epsilon_2}{n_0\mu}\right)}\right) \\
&\leq \exp\left(-\frac{\epsilon_2^2\cdot\mathsf{OPT}_u^2}{2n_0^2\mu+\frac{2}{3}\epsilon_2 n_0\mathsf{OPT}_u}\cdot T\right) \\
&\leq \exp\left(-\frac{\epsilon_2^2\cdot\mathsf{OPT}_u^2}{2n_0(1-1/\sqrt{e}-\epsilon)\cdot\mathsf{OPT}_u+\frac{2}{3}\epsilon_2 n_0\mathsf{OPT}_u}\cdot T\right) \\
&\leq \exp\left(-\frac{(\epsilon-(1-1/\sqrt{e})\epsilon_1)^2\cdot\mathsf{OPT}_u}{2n_0(1-1/\sqrt{e})}\cdot\theta_2\right) = \delta_2/\binom{n_0}{k_{max}}
\end{aligned} \tag{A14}$$

Due to $k_{max} = \max\{k : \exists S \subseteq V, c(S) \leq L\}$, there are at most $\binom{n_0}{k_{max}}$ candidate solutions. By applying union bound, the inequality in Equation (A12) happens with probability at least $1-\delta_2$. From the above results, we found that, if $T \geq \max\{\theta_1, \theta_2\}$, Algorithm 5 returns a $(1-1/\sqrt{e})$-approximation solution with probability at least $1-(\delta_1+\delta_2)$. By setting $\theta_1 = \theta_2 = \theta/2, \epsilon_2 = \epsilon - (1-1/\sqrt{e})\epsilon_1$, and

$$\epsilon_1 = \frac{\epsilon\sqrt{\ln(2/\delta)}}{(1-1/\sqrt{e})\sqrt{\ln(2/\delta)} + \sqrt{(1-1/\sqrt{e})\ln\left(2\binom{n_0}{k_{max}}/\delta\right)}}$$

we obtain $\theta_1 = \theta_2 = N(\delta,\epsilon)$. Hence, if $T \geq N(\delta,\epsilon)$, Algorithm 5 returns a $(1-1/\sqrt{e}-\epsilon)$-approximation solution with probability at least $1-(\delta_1+\delta_2) = 1-\delta$. □

References

1. Kempe, D.; Kleinberg, J.M.; Tardos, É. Maximizing the spread of influence through a social network. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 24–27 August 2003; pp. 137–146. [CrossRef]
2. Borgs, C.; Brautbar, M.; Chayes, J.T.; Lucier, B. Maximizing Social Influence in Nearly Optimal Time. In Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, ON, USA, 5–7 January 2014; pp. 946–957. [CrossRef]
3. Leskovec, J.; Krause, A.; Guestrin, C.; Faloutsos, C.; VanBriesen, J.M.; Glance, N.S. Cost-effective outbreak detection in networks. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, CA, USA, 12–15 August 2007; pp. 420–429. [CrossRef]
4. Nguyen, H.T.; Thai, M.T.; Dinh, T.N. Stop-and-Stare: Optimal Sampling Algorithms for Viral Marketing in Billion-scale Networks. In Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, 26 June–1 July 2016; pp. 695–710. [CrossRef]
5. Tang, Y.; Xiao, X.; Shi, Y. Influence maximization: Near-optimal time complexity meets practical efficiency. In Proceedings of the International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, 22–27 June 2014; pp. 75–86. [CrossRef]
6. Tang, Y.; Shi, Y.; Xiao, X. Influence Maximization in Near-Linear Time: A Martingale Approach. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, 31 May–4 June 2015; pp. 1539–1554. [CrossRef]
7. Nguyen, H.T.; Thai, M.T.; Dinh, T.N. A Billion-Scale Approximation Algorithm for Maximizing Benefit in Viral Marketing. *IEEE/ACM Trans. Netw.* **2017**, *25*, 2419–2429. [CrossRef]
8. Chen, W.; Yuan, Y.; Zhang, L. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In Proceedings of the 10th IEEE International Conference on Data Mining, Sydney, Australia, 14–17 December 2010; pp. 88–97. [CrossRef]
9. Chen, W.; Wang, C.; Wang, Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 25–28 July 2010; pp. 1029–1038. [CrossRef]
10. Nguyen, H.; Zheng, R. On Budgeted Influence Maximization in Social Networks. *IEEE J. Sel. Areas Commun.* **2013**, *31*, 1084–1094. [CrossRef]
11. Bharathi, S.; Kempe, D.; Salek, M. Competitive Influence Maximization in Social Networks. In Proceedings of the Internet and Network Economics, Third International Workshop, WINE 2007, San Diego, CA, USA, 12–14 December 2007; pp. 306–311. [CrossRef]
12. Lu, W.; Bonchi, F.; Goyal, A.; Lakshmanan, L.V.S. The bang for the buck: Fair competitive viral marketing from the host perspective. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, 11–14 August 2013; pp. 928–936. [CrossRef]
13. Chen, W.; Collins, A.; Cummings, R.; Ke, T.; Liu, Z.; Rincón, D.; Sun, X.; Wang, Y.; Wei, W.; Yuan, Y. Influence Maximization in Social Networks When Negative Opinions May Emerge and Propagate. In Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, Mesa, AZ, USA, 28–30 April 2011; pp. 379–390. [CrossRef]
14. Liu, W.; Yue, K.; Wu, H.; Li, J.; Liu, D.; Tang, D. Containment of competitive influence spread in social networks. *Knowl.-Based Syst.* **2016**, *109*, 266–275. [CrossRef]
15. Bozorgi, A.; Samet, S.; Kwisthout, J.; Wareham, T. Community-based influence maximization in social networks under a competitive linear threshold model. *Knowl.-Based Syst.* **2017**, *134*, 149–158. [CrossRef]
16. Lu, W.; Chen, W.; Lakshmanan, L.V.S. From Competition to Complementarity: Comparative Influence Diffusion and Maximization. *PVLDB* **2015**, *9*, 60–71. [CrossRef]
17. Carnes, T.; Nagarajan, C.; Wild, S.; van Zuylen, A. Maximizing Influence in a Competitive Social Network: A Follower's Perspective. In Proceedings of the Ninth International Conference on Electronic Commerce, Minneapolis, MN, USA, 19–22 August 2007; pp. 351–360.
18. Wang, X.; Zhang, Y.; Zhang, W.; Lin, X. Dominated competitive influence maximization with time-critical and time-delayed diffusion in social networks. *J. Comput. Sci.* **2018**, *28*, 318–327. [CrossRef]

19. Yan, R.; Zhu, Y.; Li, D.; Ye, Z. Minimum cost seed set for threshold influence problem under competitive models. *World Wide Web* **2018**. [CrossRef]
20. Khuller, S.; Moss, A.; Naor, J. The Budgeted Maximum Coverage Problem. *Inf. Process. Lett.* **1999**, *70*, 39–45. [CrossRef]
21. Chen, W.; Lakshmanan, L.V.S.; Castillo, C. *Information and Influence Propagation in Social Networks*; Synthesis Lectures on Data Management, Morgan & Claypool Publishers: Williston, VT, USA, 2013.
22. He, X.; Song, G.; Chen, W.; Jiang, Q. Influence Blocking Maximization in Social Networks under the Competitive Linear Threshold Model. In Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, CA, USA, 26–28 April 2012; pp. 463–474. [CrossRef]
23. Valiant, L.G. The Complexity of Enumeration and Reliability Problems. *SIAM J. Comput.* **1979**, *8*, 410–421. [CrossRef]
24. Borodin, A.; Filmus, Y.; Oren, J. Threshold Models for Competitive Influence in Social Networks. In Proceedings of the Internet and Network Economics—6th International Workshop, WINE 2010, Stanford, CA, USA,13–17 December 2010; pp. 539–550. [CrossRef]
25. Wang, X.; Zhang, Y.; Zhang, W.; Lin, X. Efficient Distance-Aware Influence Maximization in Geo-Social Networks. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 599–612. [CrossRef]
26. Song, C.; Hsu, W.; Lee, M. Targeted Influence Maximization in Social Networks. In Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, 24–28 October 2016; pp. 1683–1692. [CrossRef]
27. Li, Y.; Zhang, D.; Tan, K. Real-time Targeted Influence Maximization for Online Advertisements. *PVLDB* **2015**, *8*, 1070–1081. [CrossRef]
28. Lin, Y.; Chen, W.; Lui, J.C.S. Boosting Information Spread: An Algorithmic Approach. In Proceedings of the 33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, 19–22 April 2017; pp. 883–894. [CrossRef]
29. Goyal, A.; Lu, W.; Lakshmanan, L.V.S. CELF++: Optimizing the greedy algorithm for influence maximization in social networks. In Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, 28 March–1 April 2011; pp. 47–48. [CrossRef]
30. Jung, K.; Heo, W.; Chen, W. IRIE: Scalable and Robust Influence Maximization in Social Networks. In Proceedings of the 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, 10–13 December 2012; pp. 918–923. [CrossRef]
31. Goyal, A.; Lu, W.; Lakshmanan, L.V.S. SIMPATH: An Efficient Algorithm for Influence Maximization under the Linear Threshold Model. In Proceedings of the 11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, 11–14 December 2011; pp. 211–220. [CrossRef]
32. Budak, C.; Agrawal, D.; El Abbadi, A. Limiting the spread of misinformation in social networks. In Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, 28 March–1 April 2011; pp. 665–674. [CrossRef]
33. Tong, G.A.; Wu, W.; Guo, L.; Li, D.; Liu, C.; Liu, B.; Du, D. An efficient randomized algorithm for rumor blocking in online social networks. In Proceedings of the 2017 IEEE Conference on Computer Communications, INFOCOM 2017, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9. [CrossRef]
34. Chung, F.R.K.; Lu, L. Survey: Concentration Inequalities and Martingale Inequalities: A Survey. *Int. Math.* **2006**, *3*, 79–127. [CrossRef]
35. Dagum, P.; Karp, R.M.; Luby, M.; Ross, S.M. An Optimal Algorithm for Monte Carlo Estimation. *SIAM J. Comput.* **2000**, *29*, 1484–1496. [CrossRef]
36. Leskovec, J.; Kleinberg, J.M.; Faloutsos, C. Graph evolution: Densification and shrinking diameters. *TKDD* **2007**, *1*, 2. [CrossRef]
37. Leskovec, J.; Lang, K.J.; Dasgupta, A.; Mahoney, M.W. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Int. Math.* **2009**, *6*, 29–123. [CrossRef]
38. Richardson, M.; Agrawal, R.; Domingos, P.M. Trust Management for the Semantic Web. In Proceedings of the Semantic Web—ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, 20–23 October 2003; pp. 351–368. [CrossRef]

39. Yang, J.; Leskovec, J. Defining and Evaluating Network Communities based on Ground-truth. *CoRR* **2012**, *42*, 181–213.
40. Yin, H.; Benson, A.R.; Leskovec, J.; Gleich, D.F. Local Higher-Order Graph Clustering. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; ACM: New York, NY, USA, 2017; pp. 555–564. [CrossRef]