An Overall Evaluation on Benefits of Competitive Influence Diffusion

Jianxiong Guo, Yapu Zhang, and Weili Wu, *Senior Member, IEEE*

Abstract—Influence maximization (IM) is a representative and classic problem that has been studied extensively before. The most important application derived from the IM problem is viral marketing. Take us as a promoter, we want to get benefits from the influence diffusion in a given social network, where each influenced (activated) user is associated with a benefit. However, there is often competing information initiated by our rivals that diffuses in the same social network at the same time. Consider such a scenario, a user is influenced by both our information and our rivals' information. Here, the benefit from this user should be weakened to a certain degree. How to quantify the degree of weakening? Based on that, we propose an overall evaluation on benefits of influence (OEBI) problem. We prove the objective function of the OEBI problem is not monotone, not submodular, and not supermodular. Fortunately, we can decompose this objective function into the difference of two submodular functions and adopt a modular-modular procedure to approximate it with a data-dependent approximation guarantee. Because of the difficulty to compute the exact objective value, we design a group of unbiased estimators by exploiting the idea of reverse influence sampling, which can improve time efficiency significantly without losing its approximation ratio. Finally, numerical experiments on real datasets verified the effectiveness of our approaches regardless of performance and efficiency.

Index Terms—Overall evaluations, influence maximization, submodularity, modular-modular proceduce, sampling techniques, social networks, approximation algorithm

1 Introduction

THE online social media, such as Twitter, Facebook, Wechat, and LinkedIn, have been booming in the recent decade and have become a dominating method to contact others and make friends [1]. People are more inclined to share their comments about some hot issues at every moment in these platforms. By the end of December 2019, there were more than 3.725 billion users active in these social media. The relationships among the users on these social platforms can be denoted by social networks. A large number of messages can be shared rapidly over the networks. Subsequently, influence maximization (IM) [2] was formulated to focus on the problem of selecting a small subset of users (seed set) for an information cascade to maximize the expected follow-up adoptions (influence spread). It is a natural generalization for viral marketing. The IM problem was based on the two influence diffusion models, independent cascade model (IC-model) and linear threshold model (LT-model), and they can be summarized into the trigger model. Besides, Kempe *et al.* [2] proved the expected influence spread is monotone and submodular, thereby a $(1-1/e)$-approximation can be obtained by the greedy algorithm implemented by the Monte-Carlo (MC) simulations.

Since this seminal work, it derives a series of optimization problems, such as profit maximization (PM) [3], [4], [5], competitive IM [6], [7], [8], [9], and influence blocking [10], [11]. Consider us as a promoter to initiate an information cascade, we aim to get benefits from the influence spread started from our selected seed set in a social network. If a user is activated during the influence diffusion, we can get a benefit associated with her. Suppose there exists cost needed to pay when selecting a seed set, the profit is defined by the total benefits of influence spread minus the cost of this seed set, where the PM problem aims to maximize the expected profit. However, this is only an idealized state, where there is no competitor diffusing its cascade simultaneously. Generally, more than one type of information can flood the same network. In the competitive IM problem, there are multiple information cascades diffusing their respective influence independently, where it assumes a user can only be activated by one cascade successfully. It aims to select a seed set to maximize our own expected influence spread or to minimize the influence spread from other competing cascades (influence blocking).

Combining the PM and competitive IM problem together, it formulates a competitive PM problem that maximizes our own expected profit when there are multiple information cascades diffusing on the same network. However, this model has a crucial drawback because it assumes that each user can only be activated by one cascade. Actually, for a user in a social network, she may be influenced by multiple cascades from different promoters. If a user is

- *Jianxiong Guo and Weili Wu are with the Department of Computer Science, Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas, Richardson, TX 75080 USA. E-mail: {jianxiong.guo, weiliwu}@utdallas.edu.*
- *Yapu Zhang is with the School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: zhangyapu16@mails.ucas.ac.cn.*

Manuscript received 3 July 2020; revised 12 May 2021; accepted 24 May 2021. Date of publication 27 May 2021; date of current version 14 Mar. 2023. This work was partly supported by National Science Foundation under Grants 1747818 and 1907472. (Corresponding author: Jianxiong Guo.) Digital Object Identifier no. 10.1109/TBDATA.2021.3084468

activated by our cascade but activated by rivals' cascades contemporarily, the benefit we can get from her will be weakened, even be negative. This is very different from the existing competitive models in [6], [7], [8], [9]. Let us consider the following example.

Example 1. Take us as an Apple carrier, we want to popularize a new iPhone across a given network by initiating an influence diffusion. If a user is influenced by us, we can get a benefit from her according to her appraisal of our product. When there is a rival existing such as Samsung, it will promote its new phone by diffusing the influence as well. If a user is influenced by both Samsung and us, its appraisal of our product is very likely to be reduced after comparing our product with Samsung. The benefit we can get from her will be reduced even to be negative.

Based on this realistic scenario, we propose an overall evaluation on benefits of influence (OEBI) problem, where we define how to quantify and maximize the benefits of our targeted influence diffusion disturbed by the rival's influence diffusion. We show that the OEBI problem is NP-hard and its objective function is not monotone, not submodular, and not supermodular. Because there is no direct approach to approximating it with a theoretical bound, we decompose this objective function into the difference of two monotone and submodular functions. Then, we adopt a modular-modular procedure [12] that replaces the first submodular function with one of its lower bound and the second submodular function with one of its upper bound. Thus, a data-dependent approximation ratio can be obtained by this procedure. Moreover, it is #P-hard to compute the exact objective value under the IC-model [13] and LT-model [14]. Even though we can estimate our objective value by use of MC simulations, the terrible time inefficiency is unavoidable, which restricts its scalability to larger networks. Based on the idea of reverse influence sampling (RIS) [15], we design a group of unbiased estimators to estimate the value of our objective function. If the number of samplings is large enough, its estimation error is neglectable. Next, we take this estimator as the input of modular-modular procedure, which reduces the running time greatly while maintaining the approximation guarantee. Finally, we conduct several experiments to evaluate the superiority of our proposed method to other heuristic algorithms, where they support the effectiveness of our method strongly.

Organization. Section 2 surveys the-state-of-art works. Section 3 is dedicated to introduce diffusion model, background, and define the OEBI problem formally. The monotonicity, submodularity, and computability are presented in Section 4. Section 5 is the main contributions, including algorithm design, sampling techniques, and approximation guarantee. Numerical experiments and performance analysis are presented in Section 7, and Section 8 is the conclusion for this paper.

2 Related Work

Influence Maximization. Kempe *et al.* [2] came up with the IC-model and LT-model, formulated the IM problem as a monotone submodular maximization problem, and gave a greedy algorithm that achieves $(1-1/e-\varepsilon)$-approximation implemented by MC simulations. Chen *et al.* proved it is #P-hard to compute the expected influence spread given a seed set under the IC-model [13] and LT-model [14]. Besides, they devised two efficient heuristic algorithms to solve the IM problem and evaluate their scalability. Contemporarily, a series of heuristic algorithms emerged, such as cost-effective lazy forward strategy [16] and degree discount heuristics [17]. Brogs *et al.* [15] made a breakthrough. They proposed the concept of RIS to estimate the expected influence spread, which is scalable in practice and has a theoretical bound at the same time. Then, a series of researchers designed more efficient algorithms that achieve $(1-1/e-\varepsilon)$-approximation based on the RIS. Tang *et al.* [18], [19] proposed TIM/TIM+ algorithms first and then develop a more efficient IMM based on the martingale analysis. Besides, it was improved further by SSA/DSSA [20] and OPIM-C [21].

Competitive IM and Profit Maximization. Bharathi *et al.* [6] studied the competitive IM first and generalized it as a game of influence diffusion with multiple competing cascades. Lu *et al.* [22] created a comparative IC-model that includes all settings of influence propagation from competition to complementarity. Wu *et al.* [8] proposed two heuristic algorithms to achieve influence blocking maximization under the competitive IC-model. Arazkhani *et al.* [9] designed a community based algorithm to minimize the bad effect of misinformation under the multi-compaign IC-model. Tong *et al.* [23] proposed an independent multi-cascade model and studied a multi-cascade IM problem under this model systematically, where they designed efficient algorithms and obtained a data-dependent approximation guarantee. In the classic PM problem [3], [24], they usually considered the cost of a seed set is modular with respect to the seed node in this seed set, which implies the profit function is still submodular but not monotone. It can be generalized as the unconstrained submodular maximization problem that has a constant approximate guarantee [25]. Tong *et al.* [26] considered the coupon allocation in the PM problem, and designed efficient randomized algorithms to achieve $(1/2-\varepsilon)$-approximation with high probability. Guo *et al.* [27] proposed a constrained budgeted coupon problem and provided a continuous double greedy algorithm with a valid approximation.

Non-Submodular Maximization. However, many realistic problems derived from the IM do not satisfy the submodularity. For a monotone non-submodular function, we can use the supermodular degree [28] and curvature [29] to analyze the approximation of greedy algorithm to maximize it. Then, Lu *et al.* [22] devised a sandwich approximation framework, which can obtain a data-dependent approximation ratio by maximizing its submodular upper and submodular lower bounds, then return the solution that can maximize the original objective function as the final result. However, our objective function of the OEBI problem is not monotone. For a non-monotone non-submodular function, it can be decomposed into the difference of two submodular functions [30], which can be approximated effectively by the submodular-supermodular procedure [30] and modular-modular procedure [12].

Even though there are many existing researches about competitive IM and influence blocking problem shown as

TABLE 1
The Frequently Used Notations Summarization

Notation	Description
$G = (V, E)$	an instance of the social network
$\Omega = (G, P)$	an instance of IC-model
$\sigma_\Omega(S)$	Expected influence spread of S under Ω
g	A realization sampled from a given model
$I_g(S)$	The set of nodes that are reachable from S in the realization g
C_p/C_r	Positive cascade / Rival cascade
S_p/S_r	Positive seed set / Rival seed set
$p(u)$	Benefit of a C_p-active not C_r-active user
$q(u)$	Benefit of a C_p-active and C_r-active user
$l(u)$	$l(u) = p(u) - q(u)$
$f(S_p)$	Expected overall benefit from S_p
$w(S_p)/z(S_p)$	$f(S_p) = w(S_p) - z(S_p)$
$m^b_X(Y)$	A modular upper bound of submodular function b that is tight at set X
$h^b_{X,\alpha}(Y)$	A modular lower bound of submodular function b that is tight at set X
$\hat{w}(S_p)/\hat{z}(S_p)$	Unbiased estimator of $w(S_p)$ and $z(S_p)$

[8], [9], [10], [11], [23], their models and basic assumptions are very different from us. They all tried to initiate a positive cascade to compete with the rival's influence diffusion. Assumed that each user can be activated by at most one cascade, they aimed to maximize the expected influence spread of positive cascade. In our OEBI problem, the formulation of competitiveness and definition of benefit are more practical where each user can be activated by our cascade and rival's cascade. We aim to evaluate how the rival's cascade weakens our influence spread comprehensively, and under such circumstances, how to maximize the benefits of our expected influence spread. The objective function is non-monotone and non-submodular, which is totally different from the monotone submodular maximization problem in the above works. Besides, to overcome the intrinsically high computational complexity, we design an efficient randomized algorithm to solve our OEBI problem with a satisfactory approximation guarantee based on the RIS and modular-modular procedure.

3 Problem Formulation

In this section, we introduce the diffusion model first and then formulate the OEBI problem. The frequently used notations in this paper are shown in Table 1.

3.1 Diffusion Model and Realization

Let $G = (V, E)$ be a directed graph that represents a social network where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of n users, $E = \{e_1, e_2, \ldots, e_m\}$ is the set of m directed edges. For each directed edge $(u, v) \in E$, it models their friendship where u (resp. v) is an incoming neighbor (resp. outgoing neighbor) of v (resp. u). Moreover, the set of incoming neigbhbors (resp. outgoing neighbors) of node $u \in V$ is denoted by $N^-(v)$ (resp. $N^+(v)$).

Given a seed set $S \subseteq V$, the influence diffusion model is a discrete-time stochastic process started from the seed nodes in S. In the beginning, all nodes in the seed set S are active, but the other nodes are inactive. At time step t_i, we denote by S_i the current active node set. Thereby we have $S_0 := S$ at t_0. Under the IC-model [2], there is a diffusion probability $p_{uv} \in (0, 1]$ associated with each edge $(u, v) \in E$. At time step t_i for $i \geq 1$, we have $S_i := S_{i-1}$ first; then, each new activated node $u \in (S_{t-1} \backslash S_{t-2})$ in the last time step has one chance to activate its each inactive outgoing neighbor v with the probability p_{uv}. We add v into S_i if u activates v successfully. The influence diffusion stops when no node can be activated further. The problems we will discuss in the subsequent sections are defaulted on the IC-model, but they can be extended to other diffusion models easily.

Here, a specific IC-model based on graph G can be defined as $\Omega = (G, P)$ where $P = \{p_{e_1}, p_{e_2}, \ldots, p_{e_m}\}$ is the set of m edge probabilities. Given a specific IC-model Ω, we define $g \sim \Omega$ as a realization sampled from Ω, which is an instance of influence diffusion on this probabilistic graph. Under the IC-model, a realization is residual graph built by removing each edge $(u, v) \in E$ with probability $1 - p_{uv}$. Thereby we have $\Pr[g] = \prod_{e \in E(g)} p_e \prod_{e \in E(G) \backslash E(g)} (1 - p_e)$ and there is 2^m possible realizations in total.

Given a seed set $S \subseteq V$ and a realization g, we denote by $I_g(S)$ the set of nodes that can be reachable from at least one node in this seed set and realization. Thus, the expected number of active nodes over all possible realizations (expected influence spread) can be expressed as

$$\sigma_\Omega(S) = \mathbb{E}_{g \sim \Omega}\big[|I_g(S)|\big] = \sum_{g \in \mathcal{G}(\Omega)} \Pr[g] \cdot |I_g(S)|, \tag{1}$$

where $\mathcal{G}(\Omega)$ is the collection of all possible realizations sampled from Ω. The IM problem is to select a seed set $S \subseteq V$ where $|S| \leq k$ such that the expected influence spread $\sigma(S)$ can be maximized. Given a set function $h : 2^V \to \mathbb{R}$ and any two sets $S, T \subseteq V$, it is monotone if $h(S) \leq h(T)$ when $S \subseteq T \subseteq V$, submodular if $h(S \cup \{u\}) - h(S) \geq h(T \cup \{u\}) - h(T)$ when $S \subseteq T \subseteq V$ and $u \notin T$, and supermodular if $h(S \cup \{u\}) - h(S) \leq h(T \cup \{u\}) - h(T)$ when $S \subseteq T \subseteq V$ and $u \notin T$. Based on that, we have the expected influence spread $\sigma(\cdot)$ is monotone non-decreasing and submodular under the IC-model [2].

3.2 Problem Definition

Consider a company, it wants to promote its new product by starting a cascade diffusing over the social network. Obviously, the expected influence spread is the benefit it can obtain. However, this is only in an ideal world because it does not consider whether there is another representing a competing product started by a rival company that diffuses over the social network at the same time. Thus, we can no longer evaluate this company's benefit only by the expected influence spread due to the rival's disturbance.

Given a social network $G = (V, E)$, there are multiple cascades diffusing on this network simultaneously. A user is referred as C-active if she is activated by cascade C. Consider such a scenario, we define a positive cascade C_p which represents the influence diffusion for the new product we

want to promote over the network. There exists a rival cascade C_r represents the influence diffusion for a competing product started by some rival company. Now, due to the existence of this competing cascade, our benefit from the influence spread of cascade C_p will be disturbed and impaired to some extent. Given a rival seed set S_r, we need to find a positive seed set S_p and start this positive cascade such that it can avoid the negative effects of the rival cascade starting from S_r as much as possible.

Next, we discuss how to quantify the disturbance caused by the rival cascade to our benefit. Given a social network $G = (V, E)$, we consider a positive cascade C_p diffuses under the IC-model $\Omega^p = (G, PP)$ and a rival cascade C_r diffuses under the IC-model $\Omega^r = (G, PR)$, where PP (resp. PR) is an edge probability distribution of Ω^p (resp. Ω^r). These two cascades diffuse over the network G respectively and independently. Then, we suppose each node $u \in V$ is associated with a benefit weight $p(u) \in \mathbb{R}_+$, which implies the benefit can be obtained from the fact that u is C_p-active but not C_r-active. In other words, it is the earning from activating user u by our positive cascade but not activating it by the rival cascade. Moreover, we suppose each node $u \in V$ is associated with a disturbed benefit weight $q(u) \in \mathbb{R}$ with $q(u) \leq p(u)$, which implies the earning can be obtained from the fact that u is C_p-active and C_r-active. Here, the disturbed benefit weight describes the degree of disturbance caused by the rival cascade. For a user $u \in V$, her degree of disturbance caused by the rival cascade rests with its disturbed benefit weight $q(u)$. If $q(u) \in [0, p(u)]$, it means that the rival cascade will not cause a negative effect on this node u even though it cuts down the benefit can be obtained from activating this node by positive cascade. If $q(u) \in (-\infty, 0)$, it means that the rival cascade will cause a negative effect on this node. Thus, this q controls the degree of disturbance caused by the rival cascade.

Given a rival seed set $S_r \subseteq V$, the expected overall benefit from our positive seed set S_p can be defined as

$$f(S_p) = \mathbb{E}_{g\sim\Omega^p}\mathbb{E}_{g'\sim\Omega^r}[f_{g,g'}(S_p)] \tag{2}$$

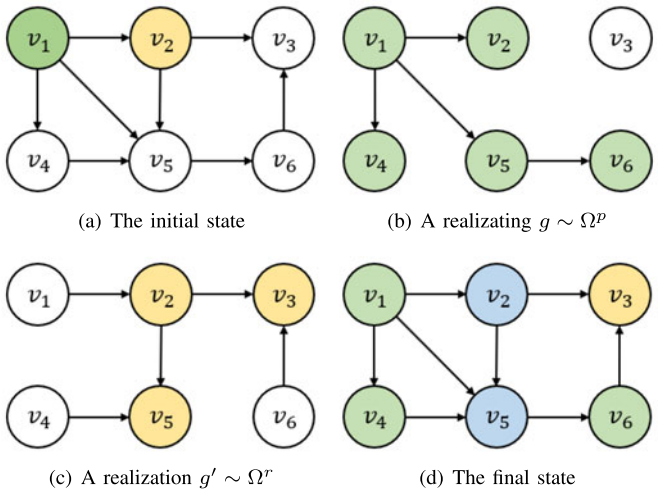
$$= \sum_{g\in\mathcal{G}(\Omega^p)} \Pr[g] \sum_{g'\in\mathcal{G}(\Omega^r)} \Pr[g'] \cdot f_{g,g'}(S_p), \tag{3}$$

where $f(S_p)$ is the expectation over the realizations sampled from the IC-model Ω^p and Ω^n. Given the two realizations $g \sim \Omega^p$ and $g' \sim \Omega^r$, the overall benefit of influence diffusion can be defined as

$$f_{g,g'}(S_p) = \sum_{u\in I_g(S_p)\setminus I_{g'}(S_r)} p(u) + \sum_{u\in I_g(S_p)\cap I_{g'}(S_r)} q(u), \tag{4}$$

where the first term is the benefit from nodes activated only by C_p and the second term is the disturbed benefit from nodes activated by both C_p and C_r.

Let us look at an example shown in Fig. 1. Shown as Fig. 1a, the positive seed set is $S_p = \{v_1\}$ and the rival seed set $S_r = \{v_2\}$ in the beginning. Then, the influence spread started from S_p is shown as Fig. 1b, which is a realization sampled from its IC-model Ω^p. Similarly, the influence spread started from S_r is shown as Fig. 1c, which is a realization sampled from its IC-model Ω^r. From here, we can see that they diffuse respectively and independently. Finally, node v_2 and v_5 are activated by both the positive and rival cascades, thereby we have $I_g(S_p) \cap I_{g'}(S_r) = \{v_2, v_5\}$ shown as Fig. 1d. Therefore, we have the overall benefit under this realization is $f_{g,g'}(S_p) = p(v_1) + p(v_4) + p(v_6) + q(v_2) + q(v_5)$. The overall evaluation on benefits of influence (OEBI) problem can be defined as follows.



(a) The initial state (b) A realizating $g \sim \Omega^p$

(c) A realization $g' \sim \Omega^r$ (d) The final state

Fig. 1. This is an example to demonstrate the diffusion precess caused by a positive cascade and a negative cascade, where the green nodes, yellow nodes, and blue nodes are activated by the positive cascade, rival cascade, and both positve and rival cascades.

Problem 1 (OEBI). *Given a social network $G = (V, E)$, a rival seed set S_r, diffusion model Ω^p and Ω^r, and a budget k, the OEBI problem aims at finding a positive set set $S_p \subseteq V$ with $|S_p| \leq k$ such that its expected overall benefit $f(S_p)$ can be maximized. That is*

$$S_p^* \in \arg\max_{|S_p|\leq k} f(S_p), \tag{5}$$

where the expected overall benefit $f(S_p)$ has been defined in Equs. (2) and (4).

4 Further Discussions About OEBI

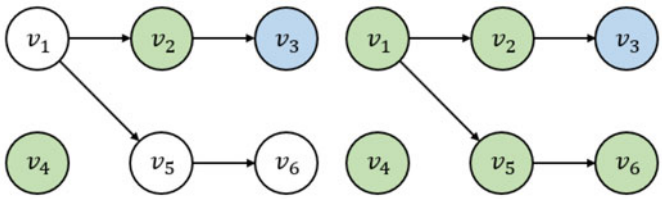
In this section, we analyze the properties of OEBI problem and introduce how to decompose its objective function.

4.1 The Properties

Given the rival seed set $S_r = \emptyset$, the OEBI problem can be reduced to the classical IM problem if we assume $p(u) = 1$ for each $u \in V$. Thus, the OEBI problem is NP-hard through inheriting the NP-hardness of IM problem [2] under the IC-model. Moreover, it is #P-hard to compute the expected overall benefit because of the #P-hardness to compute the expected influence spread under the IC-model [13]. Next, we will analyze the monotonicity, submodularity, and supermodularity of the expected overall benefit function $f(S_p)$ with respect to S_p step by step.

Theorem 1. *The objective function of the OEBI problem $f(S_p)$ is not monotone with respect to S_p.*

Proof. We consider the simplest case where the graph G has only one node. Here, we have $V = \{v\}$ ands $E = \emptyset$. Given



(a) $S_p = \{v_2, v_4\}$ (b) $S_p = \{v_1, v_4\}$

Fig. 2. This is an example to demonstrate the submodularity and supermodularity in Theorem 2.

a rival seed set $S_r = \{v\}$, the expected overall benefit $f(\{v\}) = q(u)$ and $f(\emptyset) = 0$. Subsequently, we have $f(\{v\}) - f(\emptyset) \geq 0$ if $q(u) \geq 0$; and $f(\{v\}) - f(\emptyset) \leq 0$ if $q(u) \leq 0$. Thus, the monotonicity of $f(S_p)$ depends on the definition of dusturbed earning weights. □

Theorem 2. *The objective function of the OEBI problem $f(S_p)$ is not submodular with respect to S_p and not supermodular with respect to S_p.*

Proof. Take a counterexample to prove it, we assume $p = p(u)$ and $q = q(u)$ for each node $u \in V$ with $q \in (-\infty, -p)$. Shown as Fig. 2, we can see that $f(\{v_2, v_4\}) = 2p - q$ and $f(\{v_1, v_4\}) = 5p - q$. First, we have $f(\{v_2, v_4\}) - f(v_4) = p + q < f(\{v_1, v_2, v_4\}) - f(v_1, v_4) = 0$, thereby $f(S_p)$ is not submodular with respect to S_p. Then, we have $f(\{v_4, v_5\}) - f(v_4) = 2p > f(\{v_1, v_4, v_5\}) - f(v_1, v_4) = 0$, thereby $f(S_p)$ is not supermodular with respect to S_p. □

4.2 Decomposition of Our Objective Function

From the above subsection, the expected overall benefit is non-monotone, non-submodular, and non-supermodular. Therefore, it is hard to get an effective solution with an approximation ratio. Narasimhan *et al.* [30] proposed a DS decomposition, which pointed out any set function can be decomposed into the difference of two submodular set functions. Even that, whether such two submodular set functions can be found in polynomial time is still unknown. Look at Equ. (4), the overall benefit $f_{g,g'}(S_p)$ under the $g \sim \Omega^p$ and $g' \sim \Omega^r$ can be re-arranged as

$$f_{g,g'}(S_p) = \sum_{u \in I_g(S_p)} p(u) - \sum_{u \in I_g(S_p) \cap I_{g'}(S_r)} (p(u) - q(u)). \tag{6}$$

Thus, we can decompose the expected overall benefit as $f(S_p) = w(S_p) - z(S_p)$, where $w(S_p)$ and $z(S_p)$ are defined as follows, that is

$$w(S_p) = \mathbb{E}_{g \sim \Omega^p} \left[\sum\nolimits_{u \in I_g(S_p)} p(u) \right] \tag{7}$$

$$z(S_p) = \mathbb{E}_{g \sim \Omega^p} \mathbb{E}_{g' \sim \Omega^r} \left[\sum\nolimits_{u \in I_g(S_p) \cap I_{g'}(S_r)} l(u) \right], \tag{8}$$

where we denote $l(u) = p(u) - q(u)$. Similarly, we denote $w_g(S_p) = \sum_{u \in I_g(S_p)} p(u)$ under the $g \sim \Omega^p$ and $z_{g,g'}(S_p) = \sum_{u \in I_g(S_p) \cap I_{g'}(S_r)} l(u)$ under the $g \sim \Omega^p$ and $g' \sim \Omega^r$.

Theorem 3. *The function $w(S_p)$ is monotone non-decreasing and submodular with respect to S_p.*

Proof. The function $w(S_p)$ is the objective function of weighted IM problem. It can be reduced to weighted maximum set cover problem, which is monotone non-decreasing and submodular since $p(u) \geq 0$ for any $u \in V$. □

Theorem 4. *The function $z(S_p)$ is monotone non-decreasing and submodular with respect to S_p.*

Proof. Given a rival seed set S_r, realization $g \sim \Omega^p$, and $g' \sim \Omega^r$, we consider the monotonicity and submodularity based on $z_{g,g'}(S_p)$. First, it is apparent that $z_{g,g'}(S_p)$ is monotone non-decreasing with respect to S_p. Then, there are two positive seed set S_p^1 and S_p^2 with $S_p^1 \subseteq S_p^2$. For any node in $I_{g'}(S_r)$, if it is reachable from node v but is not reachable from S_p^2, it must not be reachable from S_p^1 since $S_p^1 \subseteq S_p^2$. Thereby we have $z_{g,g'}(S_p^1 \cup \{v\}) - z_{g,g'}(S_p^1) \geq z_{g,g'}(S_p^2 \cup \{v\}) - z_{g,g'}(S_p^2)$ because of $l(u) \geq 0$ for any $u \in V$, which implies that $z_{g,g'}(S_p)$ is submodular with respect to S_p. Besides, $y(S_p)$ is a linear combination of $z_{g,g'}(S_p)$, thus $z(S_p)$ is monotone non-decreasing and submodular. □

Therefore, the expected overall benefit $f(S_p)$ has been decomposed into the difference of two monotone submodular functions $w(S_p)$ and $z(S_p)$ definitely.

5 Algorithm DEGISN and Speedup

From the last section, our objective function is not monotone, not submodular, and not supermodular. Fortunately, it can be decomposed into the difference of two monotone submodular functions. Iyer *et al.* [12] proposed a modular-modular procedure to minimize the difference between two submodular functions approximately. We have known that $f(S_p) = w(S_p) - z(S_p)$. We can find a modular lower bound of $w(\cdot)$ and a modular upper bound of $z(\cdot)$ that are tight at current set S_p. Thus, the difference between the lower bound of $w(\cdot)$ and the upper bound of $z(\cdot)$ is a lower bound of the objective function $f(\cdot)$. The main idea of modular-modular procedure is to maximize this lower bound in each iteration, which can be done in polynomial time. It can guarantee to increase our objective value in each iteration.

First, we need to define the modular upper bound and modular lower bound for a given submodular function.

5.1 Modular-Modular Procedure

Given a submodular function $b(\cdot)$, it has two modular upper bounds based on a given set $X \subseteq V$, that is

$$m_{X,1}^b(Y) = b(X) - \sum_{j \in X \setminus Y} b(j|X \backslash j) + \sum_{j \in Y \setminus X} b(j|\emptyset) \tag{9}$$

$$m_{X,2}^b(Y) = b(X) - \sum_{j \in X \setminus Y} b(j|V \backslash j) + \sum_{j \in Y \setminus X} b(j|X), \tag{10}$$

where $b(S|T) = b(S \cup T) - b(T)$, $m_{X,1}^b(Y) \geq b(Y)$, and $m_{X,2}^b(Y) \geq b(Y)$. They are tight at set X, so we have $m_{X,1}^b(X) = m_{X,2}^b(X) = f(X)$.

Given a set $X \subseteq V$, we define a permutation α of V as $\alpha = \{\alpha(1), \alpha(2), \ldots, \alpha(n)\}$ where α's chain contains X. Denote by $S_i^\alpha = \{\alpha(1), \alpha(2), \ldots, \alpha(i)\}$, we have $S_{|X|}^\alpha = X$, in other

words, we put all the elements in X prior to the elements in $V\backslash X$. Then, we define

$$h^b_{X,\alpha}(\alpha(i)) = b(S^\alpha_i) - b(S^\alpha_{i-1}), \tag{11}$$

where $h^b_{X,\alpha}(Y) = \sum_{v\in Y} h^b_{X,\alpha}(v)$ and $h^b_{X,\alpha}(Y) \leq b(Y)$ for any $Y \subseteq V$. Here, $h^b_{X,\alpha}(Y)$ is a lower bound of $b(Y)$. It is tight at set X, wo we have $h^b_{X,\alpha}(X) = b(X)$.

According to the definition of Equs. (7) and (8), we adopt the modular-modular proceduce to approximate it, which is formulated in Algorithm 1.

Algorithm 1. Modular-Modular

Input: A set function $f : 2^V \to \mathbb{R}$
1: Initialize: $X^t \leftarrow \emptyset$, $t \leftarrow 0$
2: **while** $X^{t+1} \neq X^t$ **do**
3: Selects a permutation α^t that contains X^t where the element in X^t are ranked ahead
4: $X^{t+1} \leftarrow \arg\max_{|Y|\leq k}\left\{h^w_{X^t,\alpha^t}(Y) - m^z_{X^t}(Y)\right\}$
5: $t \leftarrow t+1$
6: **end while**
7: **return** X^t

Theorem 5. *The objective function $f(X^t)$ is monotone non-decreasing with respect to t. If the $h^w_{X^t,\alpha^t}(Y) - m^z_{X^t}(Y)$ in line 4 of Algorithm 1 reaches a local maximum under the $O(n)$ different permutations α^t and both upper bounds, then the $f(Y)$ is a local maximum.*

Proof. Regardless of what the upper bound we use, in any iteration t, we have

$$\begin{aligned} f(X^{t+1}) &= w(X^{t+1}) - z(X^{t+1}) && (12)\\ &\geq h^w_{X^t,\alpha^t}(X^{t+1}) - m^z_{X^t}(X^{t+1}) && (13)\\ &\geq h^w_{X^t,\alpha^t}(X^t) - m^z_{X^t}(X^t) && (14)\\ &= w(X^t) - z(X^t) && (15)\\ &= f(X^t), && (16)\end{aligned}$$

where In Equ. (13) is based on the definitions of the upper bound and lower bound, In Equ. (14) is because the X^{t+1} maximizes the value of $h^w_{X^t,\alpha^t}(\cdot) - m^z_{X^t}(\cdot)$, and Equ. (15) is due to the tightness at set X^t.

Suppose the Algorithm 1 converges at $X^{t+1} = X^t$, we consider the $O(n)$ different permutations α^t which are placed with different elements at position $\alpha^t(|X^t|)$ and $\alpha^t(|X^{t+1}|)$. First, we have $h^w_{X,\alpha}(S^\alpha_i) = w(S^\alpha_i)$, $m^z_{X^t,1}(X^t\backslash j) = z(X^t) - z(j|X^t\backslash j) = z(X^t\backslash j)$, and $m^z_{X^t,2}(X^t \cup j) = z(X^t) + z(j|X^t) = z(X^t \cup j)$. At the convergence, we have $h^w_{X^t,\alpha^t}(X^t) - m^z_{X^t}(X^t) \geq h^w_{X^t,\alpha^t}(Y) - m^z_{X^t}(Y)$ for any $Y \subseteq V$ under the $O(n)$ different permutations α^t and both upper bounds. Given a α^t with $\alpha^t(|X^t|) = i$ and $\alpha^t(|X^t|+1) = j$, we have

$$\begin{aligned} f(X^t) &= w(X^t) - z(X^t) && (17)\\ &= h^w_{X^t,\alpha^t}(X^t) - m^z_{X^t,1}(X^t) && (18)\\ &\geq h^w_{X^t,\alpha^t}(X^t\backslash i) - m^z_{X^t,1}(X^t\backslash i) && (19)\\ &= f(X^t\backslash i), && (20)\end{aligned}$$

and

$$\begin{aligned} f(X^t) &= w(X^t) - z(X^t) && (21)\\ &= h^w_{X^t,\alpha^t}(X^t) - m^z_{X^t,2}(X^t) && (22)\\ &\geq h^w_{X^t,\alpha^t}(X^t \cup j) - m^z_{X^t,2}(X^t \cup j) && (23)\\ &= f(X^t \cup j). && (24)\end{aligned}$$

Therefore, $f(X^t)$ is a local maximum at the convergence. □

Algorithm 2. ModularMax

Input: A permutation α^t and a set X^t
1: Initialize: a map $unitValue = \{\}$
2: Initialize: a set $X^{t+1} \leftarrow \emptyset$
3: $zero \leftarrow h^w_{X^t,\alpha^t}(\emptyset) - m^z_{X^t}(\emptyset)$
4: **for** each $u \in V$ **do**
5: $unitValue[u] \leftarrow h^w_{X^t,\alpha^t}(\{u\}) - m^z_{X^t}(\{u\}) - zero$
6: **end for**
7: **for** $i = 1$ to k **do**
8: Select $u^* \in \max_{u\in V\backslash X^{t+1}} unitValue[u]$
9: **if** $unitValue[u^*] < 0$ **then**
10: Break
11: **end if**
12: $X^{t+1} \leftarrow X^{t+1} \cup \{u^*\}$
13: **end for**
14: **return** X^{t+1}

In each iteration of this algorithm, we need to maximize a modular function shown as in line 4 of Algorithm 1, which can be implemented easily. For example, we can compute the objective value for each node $u \in V$ and then select all those which has a non-negative objective value. In the iteration t, given a permutation α^t and a set X^t, the algorithm that selects a set Y where $|Y| \leq k$ to maximize the modular function $h^w_{X^t,\alpha^t}(Y) - m^z_{X^t}(Y)$ is shown in Algorithm 2. The update rule in Algorithm 2 is according to $h(u|S) = h(u|T) = h(u|\emptyset)$ for any set $S, T \subseteq V$ if $h(\cdot)$ is a modular function.

As for how to select a permutation α^t in each iteration X^t, the optimal solution is to select a permutation α^* such that $\alpha^t_* \in \arg\max_{\alpha^t}\max_{|Y|\leq k}\{h^w_{X^t,\alpha^t}(Y) - m^z_{X^t}(Y)\}$, however it is very difficult to execute. There are $n!$ permutations in total. Thus, a heuristic choice is to order the permutation α^t according to the magnitude of objective value for each node $u \in V$. We will compare the impact of different permutations on algorithm performance in later experiments.

According to Equs. (9) and (10), we have two upper bounds for a submodular function. Thereby the upper bound of the optimal value of our expected overall benefit $f(S^*_p)$ can be defined as follows:

$$\pi(X) = \max_{|Y|\leq k}\{\min\{m^w_{X,1}(Y), m^w_{X,2}(Y)\} - h^z_{X,\alpha}(Y)\}, \tag{25}$$

where $\min\{m^w_{X,1}(Y), m^w_{X,2}(Y)\}$ is aimed to make this upper bound tighter. It can be solved similar to the process of Algorithm 2. Then, for any set X, we have $\pi(X) \geq \max_{|Y|\leq k} f(Y)$. Denote by S°_p the seed set returned by Algorithm 1, we have $\pi(S^\circ_p) \geq f(S^*_p)$, then we are able to estimate the approximation ratio by $f(S^\circ_p)/\pi(S^\circ_p)$.

5.2 Sampling Techniques

Given a seed set S_p, we adopt the technique of reverse influence sampling (RIS) to estimate $f(S_p)$ due to its #P-hardness. Consider the IM problem under the IC-model $\Omega = (G, P)$, we introduce the concept of reverse reachable set (RR-set) first. A random RR-set R can be generated by three steps: (1) Selecting a node $u \in V$ uniformly; (2) Sampling a realization $g \sim \Omega$; and (3) Collecting those nodes that can reach u in realization g and putting them into R. A RR-set rooted at node u is a collection of nodes that are likely to influence u. A larger expected influence spread a seed set S has, the higher the probability that S intersects with a random RR-set is. Given a seed set S and a random RR-set R, we have $\sigma_\Omega(S) = n \cdot \Pr[R \cap S \neq \emptyset]$.

Back to our OEBI problem, the expected overall benefit can be denoted by $f(S_p) = w(S_p) - z(S_p)$. Thus, given a seed set S_p, we require to estimate $w(S_p)$ and $z(S_p)$ respectively. Here, we denote by $p(V) = \sum_{v \in V} p(v)$ and $l(V) = \sum_{v \in V} l(v)$ respectively for convenience. For the $w(S_p)$, a random RR-set R_w can be generated by

1) Selecting a node $u \in V$ with probability $p(u)/p(V)$.
2) Sampling a realization $g \sim \Omega^p$.
3) Collecting those nodes that can reach u in realization g and putting them into R_w.

Given a seed set S_p and a random RR-set R_w, we have $w(S) = p(V) \cdot \Pr[R_w \cap S_p \neq \emptyset]$. For the $z(S_p)$, a random RR-set R_z can be generated by

1) Selecting a node $u \in V$ with probability $l(u)/l(V)$.
2) Sampling a realization $g \sim \Omega^p$ and a realization $g' \sim \Omega^r$ independently.
3) Collecting those nodes that can reach u in realization g and putting them into $R_{z,1}$; Collectiong those nodeds that can reach u in realizationg g' and putting them into $R_{z,2}$. Then, we have $R_z = (R_{z,1}, R_{z,2})$.

Lemma 1. *Given a seed set S_p, a rival seed set S_r, and a random RR-set $R_z = (R_{z,1}, R_{z,2})$, we have*

$$z(S_p) = l(V) \cdot \Pr\big[S_p \cap R_{z,1} \neq \emptyset \wedge S_r \cap R_{z,2} \neq \emptyset\big]. \tag{26}$$

Proof. We denote by $R_{z,1}(g, u)$ the RR-set rooted at node u under the realization $g \sim \Omega^p$. According to Equ. (8), we have $z(S_p) = \mathbb{E}_{g\sim\Omega^p}\mathbb{E}_{g'\sim\Omega^r}[\sum_{u\in I_g(S_p)\cap I_{g'}(S_r)} l(u)] = \sum_{u\in V} \Pr_{g\sim\Omega^p,\ g'\sim\Omega^r}[S_p \cap R_{z,1}(g,u) \neq \emptyset \wedge S_r \cap R_{z,2}(g',u) \neq \emptyset] \cdot l(u) = l(V) \cdot \sum_{u\in V} \Pr_{g\sim\Omega^p,g'\sim\Omega^r}[S_p \cap R_{z,1}(g,u) \neq \emptyset \wedge S_r \cap R_{z,2}(g',u) \neq \emptyset] \cdot (l(u)/l(V)) = l(V) \cdot \Pr_{g\sim\Omega^p,g'\sim\Omega^r,u}[S_p \cap R_z(g,g',u) \neq \emptyset \wedge S_r \cap R_z(g,g',u) \neq \emptyset]$. Equ. (26) can be established equivalently. □

As mentioned above, we have to generate two collections of RR sets, $\mathcal{R}_w = \{R_w^1, R_w^2, \ldots, R_w^\lambda\}$ to estimate $w(S_p)$ and $\mathcal{R}_z = \{R_z^1, R_z^2, \ldots, R_z^\mu\}$ to estimate $z(S_p)$. Then, we define the following two estimations

$$F_{\mathcal{R}_w}(S_p) = \frac{1}{\lambda} \cdot \sum_{i=1}^{\lambda} \mathbb{I}[S_p \cap R_w^i \neq \emptyset] \tag{27}$$

$$F_{\mathcal{R}_z}(S_p) = \frac{1}{\mu} \cdot \sum_{i=1}^{\mu} \mathbb{I}[S_p \cap R_{z,1}^i \neq \emptyset \wedge S_r \cap R_{z,2}^i \neq \emptyset]. \tag{28}$$

They are the fractions of RR-sets covered by S_p where $\mathbb{I}[\cdot]$ is an indicator such that $\mathbb{I}[S_p \cap R_w^i \neq \emptyset] = 1$ if $S_p \cap R_w^i \neq \emptyset$, or else $\mathbb{I}[S_p \cap R_w^i \neq \emptyset] = 0$. Then, we can defined the following two unbiased estimators

$$\hat{w}(S_p) = p(V) \cdot F_{\mathcal{R}_w}(S_p); \hat{z}(S_p) = l(V) \cdot F_{\mathcal{R}_z}(S_p), \tag{29}$$

where $\hat{w}(S_p)$ is an unbiased estimator of $w(S_p)$ and $\hat{z}(S_p)$ is an unbiased estimator of $z(S_p)$. Thus, we have $\hat{f}(S_p) = \hat{w}(S_p) - \hat{z}(S_p)$. Next, we need to bound the gap between ground-truth values and our estimators.

Lemma 2 (Chernoff-Hoeffding). *Let $X_1, X_2, \ldots, X_\theta$ be a series of random variables sampled from a distribution X with expectation $\mathbb{E}[X]$ independently and identically in the set $\{0, 1\}$. Given an error $\varepsilon > 0$, we have*

$$\Pr\Big[\sum_{i=1}^{\theta} X_i - \theta \cdot \mathbb{E}[X] \geq +\varepsilon\Big] \leq \exp\left(-\frac{2\varepsilon^2}{\theta}\right) \tag{30}$$

$$\Pr\Big[\sum_{i=1}^{\theta} X_i - \theta \cdot \mathbb{E}[X] \leq -\varepsilon\Big] \leq \exp\left(-\frac{2\varepsilon^2}{\theta}\right). \tag{31}$$

According to the Lemma 2, we can get the relationship between $p(V) \cdot F_{\mathcal{R}_w}(S_p)$ and its real value $w(S_p)$.

Lemma 3. *Given a collection of RR-sets $\mathcal{R}_w$ with $|\mathcal{R}_w| = \lambda$ and any $\delta \in (0, 4)$, we have*

$$\Pr\left[w(S_p) \geq \hat{w}(S_p) - p(V)\sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)}\right] \geq 1 - \frac{\delta}{4} \tag{32}$$

$$\Pr\left[w(S_p) \leq \hat{w}(S_p) + p(V)\sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)}\right] \geq 1 - \frac{\delta}{4}. \tag{33}$$

Proof. To show In Equ. (32), it is equivalent to prove $\Pr[w(S_p) < \hat{w}(S_p) - p(V) \cdot \sqrt{(1/(2\lambda))\ln(4/\delta)}] \leq \delta/4$. Thus, we have

$$\begin{aligned}
&\Pr\left[w(S_p) < p(V) \cdot F_{\mathcal{R}_w}(S_p) - p(V) \cdot \sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)}\right] \\
&= \Pr\left[\lambda \cdot F_{\mathcal{R}_w}(S_p) - \frac{\lambda \cdot w(S_p)}{p(V)} > \sqrt{\frac{\lambda}{2}\ln\left(\frac{4}{\delta}\right)}\right] \\
&\leq \exp\left(-\frac{2 \cdot \frac{\lambda}{2}\ln\left(\frac{4}{\delta}\right)}{\lambda}\right) \\
&= \delta/4,
\end{aligned} \tag{34}$$

where In Equ. (34) based on Lemma 2.

Similarly, to show In Equ. (33), it is equivalent to prove $\Pr[w(S_p) > \hat{w}(S_p) + p(V) \cdot \sqrt{(1/(2\lambda))\ln(4/\delta)}] \leq \delta/4$. Thus, we have

$$\begin{aligned}&\Pr\left[w(S_p) < p(V)\cdot F_{\mathcal{R}_w}(S_p) + p(V)\cdot\sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)}\right]\\&= \Pr\left[\lambda\cdot F_{\mathcal{R}_w}(S_p) - \frac{\lambda\cdot w(S_p)}{p(V)} < -\sqrt{\frac{\lambda}{2}\ln\left(\frac{4}{\delta}\right)}\right] \quad (35)\\&\le \exp\left(-\frac{2\cdot\frac{\lambda}{2}\ln\left(\frac{4}{\delta}\right)}{\lambda}\right)\\&= \delta/4,\end{aligned}$$

where In Equ. (35) based on Lemma 2. □

Lemma 4. *Given a collection of RR-sets $\mathcal{R}_z$ with $|\mathcal{R}_z| = \mu$ and any $\delta \in (0,4)$, we have*

$$\Pr\left[z(S_p) \ge \hat{z}(S_p) - l(V)\sqrt{\frac{1}{2\mu}\ln\left(\frac{4}{\delta}\right)}\right] \ge 1 - \frac{\delta}{4} \quad (36)$$

$$\Pr\left[z(S_p) \le \hat{z}(S_p) + l(V)\sqrt{\frac{1}{2\mu}\ln\left(\frac{4}{\delta}\right)}\right] \ge 1 - \frac{\delta}{4}. \quad (37)$$

Proof. It can be derived similar to the proof process of Lemma 3, which is based on Lemma 2. □

Given an unbiased estimator $\hat{w}(S_p)$, an upper bound and a lower bound of $w(S_p)$ can be defined with at least $1-\delta/4$ probability. That is

$$w_u(S_p) = \hat{w}(S_p) + p(V)\cdot\sqrt{\left(\frac{1}{2\lambda}\right)\ln\left(\frac{4}{\delta}\right)} \quad (38)$$

$$w_l(S_p) = \hat{w}(S_p) - p(V)\cdot\sqrt{\left(\frac{1}{2\lambda}\right)\ln\left(\frac{4}{\delta}\right)}. \quad (39)$$

Given a collection of RR-sets $\mathcal{R}_z$ with $|\mathcal{R}_z| = \mu$, any $\delta \in (0,4)$, and an unbiased estimator $\hat{z}(S_p)$, an upper bouand and a lower bound of $z(S_p)$ can be defined at least $1-\delta/4$ probability in the same way. That is

$$z_u(S_p) = \hat{z}(S_p) + l(V)\cdot\sqrt{\left(\frac{1}{2\mu}\right)\ln\left(\frac{4}{\delta}\right)} \quad (40)$$

$$z_l(S_p) = \hat{z}(S_p) - l(V)\cdot\sqrt{\left(\frac{1}{2\mu}\right)\ln\left(\frac{4}{\delta}\right)}. \quad (41)$$

Based on Equs. (38), (39), (40), and (41), we can derive a lower bound for our objective value $f(S_p)$ naturally.

Lemma 5. *Given any seed set $S_p \subseteq V$, we can take $w_u(S_p) - z_l(S_p)$ as an upper bound of $f(S_p)$ with at least $1-\delta/2$ probability and $w_l(S_p) - z_u(S_p)$ as a lower bound of $f(S_p)$ with at least $1-\delta/2$ probability.*

Proof. To estimate the $f(S_p)$, we have

$$\begin{aligned}&\Pr[f(S_p) \le w_u(S_p) - z_l(S_p)]\\&\quad\ge \Pr[(w(S_p) \le w_u(S_p)) \wedge (z(S_p) \ge z_l(S_p))]\\&\quad= (1-\delta/4)\cdot(1-\delta/4)\\&\quad\ge 1-\delta/2.\end{aligned}$$

Similarly, we have

$$\begin{aligned}&\Pr[f(S_p) \ge w_l(S_p) - z_u(S_p)]\\&\quad\ge \Pr[(w(S_p) \ge w_l(S_p)) \wedge (z(S_p) \le z_u(S_p))]\\&\quad= (1-\delta/4)\cdot(1-\delta/4)\\&\quad\ge 1-\delta/2.\end{aligned}$$

Therefore, we have $w_l(S_p) - z_u(S_p) \le f(S_p) \le w_u(S_p) - z_l(S_p)$ with a high probability. □

Next, we are going to discuss how to compute the upper bound of our objective value $\pi(S_p^\circ)$ according to the solution S_p° returned by Algorithm 1 with our sampling techniques (computing all functions by our unbiased estimators). The unbiased estimator of $\pi(S_p)$ is denoted by $\hat{\pi}(S_p)$, and the value of $\hat{\pi}(S_p)$ can be obtained by $\hat{f}(\cdot) = \hat{w}(\cdot) - \hat{z}(\cdot)$. It is implemented by the same way as using Equ. (25), except that $\hat{f}(\cdot)$ is used instead of $f(\cdot)$ to compute the upper and lower bounds. Here, $\hat{w}(S_p)$ and $\hat{z}(S_p)$ are monotone and submodular with respect to S_p as well since they can be reduced to the classic set coverage problem. According to their submodularity, we have $\hat{\pi}(X) \ge \max_{|Y|\le k}\hat{f}(Y)$ for any set X,. From the Lemma 5, the objective value $f(S_p)$ is upper bounded by $w_u(S_p) - z_l(S_p)$ with a high probability. Thereby we have the following conclusions.

Lemma 6. *Given the solution S_p° returned by Algorithm 1 with our sampling techniques, for any seed set $|S_p| \le k$ and any $\delta \in (0,4)$, we have*

$$\begin{aligned}f(S_p) \le\ &\hat{\pi}(S_p^\circ)\\&+ p(V)\cdot\sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)} + l(V)\cdot\sqrt{\frac{1}{2\mu}\ln\left(\frac{4}{\delta}\right)},\end{aligned} \quad (42)$$

holds with at least $1 - 2/\delta$ probability.

Proof. According to Lemma 5, we have $\Pr[f(S_p) \le w_u(S_p) - z_l(S_p)] \ge 1-\delta/2$. Thus, we have $f(S_p) \le w_u(S_p) - z_l(S_p) = \hat{w}(S_p) - \hat{z}(S_p) + p(V)\cdot\sqrt{(1/(2\lambda))\ln(4/\delta)} + l(V)\cdot\sqrt{(1/(2\mu))\ln(4/\delta)} = \hat{f}(S_p) + p(V)\cdot\sqrt{(1/(2\lambda))\ln(4/\delta)} + l(V)\cdot\sqrt{(1/(2\mu))\ln(4/\delta)} \le \hat{\pi}(S_p^\circ) + p(V)\cdot\sqrt{(1/(2\lambda))\ln(4/\delta)} + l(V)\cdot\sqrt{(1/(2\mu))\ln(4/\delta)}$ due to the fact that $\hat{\pi}(S_p^\circ) \ge \max_{|S_p|\le k}\hat{f}(S_p)$. It holds with at least $1-\delta/2$ probability. □

Theorem 6. *The approximation guarantee achieved by the solution S_p° returned by Algorithm 1 with our sampling techniques satisfies as follows: $f(S_p^\circ)/\max_{|S_p|\le k} f(S_p) \ge$*

$$\frac{w_l(S_p^\circ) - z_u(S_p^\circ)}{\hat{\pi}(S_p^\circ) + p(V)\sqrt{\frac{1}{2\lambda}\ln\left(\frac{4}{\delta}\right)} + l(V)\sqrt{\frac{1}{2\mu}\ln\left(\frac{4}{\delta}\right)}}, \quad (43)$$

holds with at least $1-\delta$ probability.

TABLE 2
The Datasets Statistics ($K = 10^3$)

Dataset	n	m	Type	Avg.Degree
Netscie	0.40 K	1.01 K	undirect	5.00
Wikivot	1.00 K	3.15 K	directed	6.20
Bitcoin	4.00 K	25.1 K	directed	12.5

Proof. Based on Lemma 5, we have $f(S_p^\circ) \geq w_l(S_p^\circ) - z_u(S_p^\circ)$ holds with at least $1 - \delta/2$ probability. Then based on the Lemma 6, we have $\max_{|S_p| \leq k} f(S_p) \leq \hat{\pi}(S_p^\circ) + p(V) \cdot \sqrt{(1/(2\lambda)) \ln(4/\delta)} + l(V) \cdot \sqrt{(1/(2\mu)) \ln(4/\delta)}$ holds with at least $1 - \delta/2$ probability. Thereby the approximation (43) is established with at least $1 - \delta$ probability. □

6 Numerical Experiments

In this section, we carry out several experiments on different datasets to validate the performance of our proposed algorithms. It aims to test the efficiency of modular-modular procedure, shown as Algorithm 1, and its effectiveness compared to other heuristic algorithms. All of our experiments are programmed by Python, and run on Windows machine with a 3.40 GHz, 4 core Intel CPU and 16 GB RAM. There are three datasets used in our experiments: (1) NetScience [31]: a co-authorship network, co-authorship among scientists to publish papers about network science; (2) Wiki [31]: a who-votes-on-whom network, which comes from the collection Wikipedia voting; (3) Bitcoin [32]: a who-trusts-whom network of people who trade using Bitcoin on a platform called Bitcoin Alpha. The statistical information about these three datasets is represented in Table 2. For an undirected graph, each undirected edge is replaced with two reversed directed edges.

6.1 Experimental Settings

The diffusion process is based on the IC-model by default. Under the IC-model, we set the diffusion probability $p_{uv} = 1/|N^-(v)|$ for each $(u, v) \in E$ as the inverse of v's in-degree, which has been given by many existing researches about the IM problem. For each node $u \in V$, there is a benefit weight and a disturbed benefit weight associated with it. We sample the benefit weight $p(u)$ from [0,1] uniformly and sample the corresponding disturbed benefit weight $q(u)$ from $[-1, p(u)]$ uniformly for each $u \in V$.

Consider the modular-modular procedure, we have to define a modular lower bound for the function $w(\cdot)$ and a modular upper bound for the function $z(\cdot)$. Here, we denote "Modmod-1" to imply that we use the first upper bound $m^z_{X,1}(Y)$ defined in Equ. (9) and "Modmod-2" to imply that we use the second upper bound $m^z_{X,2}(Y)$ defined in Equ. (10). Then, we need to compare our modular-modular procedure with other heuristic algorithms, especially for Greedy algorithm. Greedy algorithm is shown in Algorithm 3, which selects the node with the maximum marginal expected overall benefit in each iteration until there is no positive marginal gain can be obtained. Other heuristic algorithms are shown as follows.

1) Random: it selects k nodes uniformly from the node set.
2) MaxDegree: it selects k nodes with the largest out-degree.
3) InfMax: it is similar to the greedy algorithm, but substitutes the overall benefit $f(\cdot)$ with benefit $w(\cdot)$.

Their objective vaules are all estimated on the same collection of RR-sets, where the number of random RR-set R_w and R_z is denoted by $\theta = \lambda = \mu$. Here, we set the parameter $\delta = 0.1$, which means that the approximation ratios can be satisfied with at least 0.9 probability.

Algorithm 3. Greedy

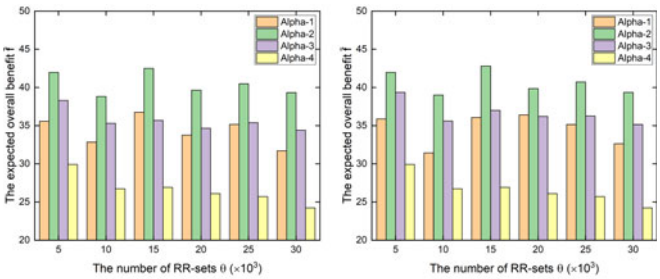
Input: A set function $f : 2^V \to \mathbb{R}$
1: Initialize: $S_p \leftarrow \emptyset$
2: **for** $i = 1$ to k **do**
3: Select u^* such that $u^* \in \arg\max_{u \in V \setminus S_p} f(u|S_p)$
4: **if** $f(u^*|S_p) < 0$ **then**
5: Break
6: **end if**
7: $S_p \leftarrow S_p \cup \{u^*\}$
8: **end for**
9: **return** S_p

To get a lower bound, the optimal permutation selections is very hard, thus we give several heuristic strategies to get that efficiently. For the permutation α^t that contains X^t in each iteration, there are four heuristic selection strategies to get it, which are shown as follows. (1) Alpha-1: rearranging X^t and $V \setminus X^t$ randomly and respectively, and then concatenating them together as a α^t; (2) Alpha-2: sorting X^t and $V \setminus X^t$ respectively from largest to smallest according to the expected overall benefit $f(u)$ for each $u \in V$, and then concatenating them together as a α^t; (3) Alpha-3: sorting X^t and $V \setminus X^t$ respectively from largest to smallest according to the expected benefit $w(u)$ for each $u \in V$, and then concatenating them together as a α^t; and (4) Alpha-4: sorting X^t and $V \setminus X^t$ respectively from smallest to largest according to the $z(u)$ for each $u \in V$, and then concatenating them together as a α^t. Then, we will test which strategy is the best.

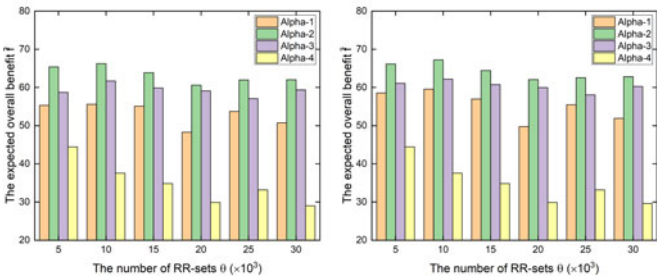
6.2 Experimental Results

1) Permutation Selection. Fig. 3 shows the performance comparison of modular-modular procedure under the aforementioned four permutation selections. Shown as Fig. 3, the solution achieved under the Alpha-2 that permutates according to the expected overall benefit has the best performance. Thus, in the follow-up experiments, we default that modular-modular procedures we will use are implemented under the Alpha-2. The performance under the Alpha-3 is slightly worse than that under the Alpha-2. The performance under the Alpha-4 is extremely worse, which implies this heuristic selection is invalid. Moreover, the performance under the Alpha-1 with random permutation selection is unstable, where the expected benefit is sometimes large sometimes small.

2) Performance of Different Algorithms. Figs. 4, 5, and 6 show the performance comparison with other heuristic algorithms under the different datasets. In these figures, we test the algorithms under the different number of RR-sets. Obviously, the estimations will be more and more accurate as the number of RR-sets increases, but the gap looks

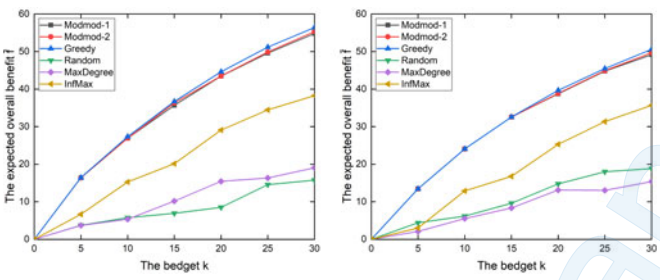


(a) Netscie, modmod-1 (b) Netscie, modmod-2

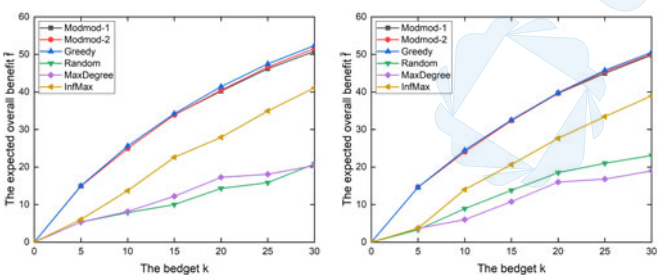


(c) Wikivot, modmod-1 (d) Wikivot, modmod-2

Fig. 3. The performance comparison of four permutation selections under the different datasets and upperbounds.

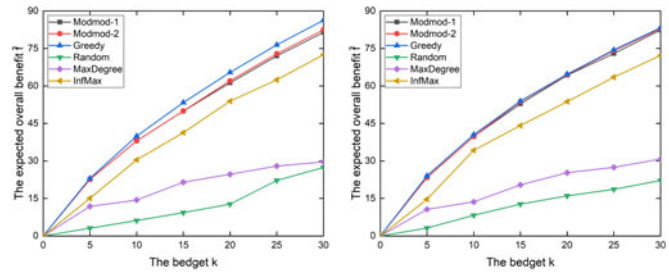


(a) Netscie, $\theta = 5K$ (b) Netscie, $\theta = 10K$

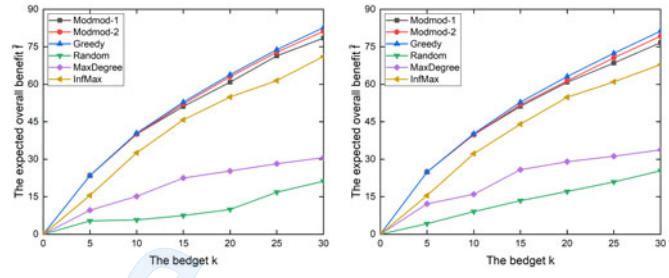


(c) Netscie, $\theta = 15K$ (d) Netscie, $\theta = 20K$

Fig. 4. The performance comparison with other heuristic algorithms under the Netscie dataset.

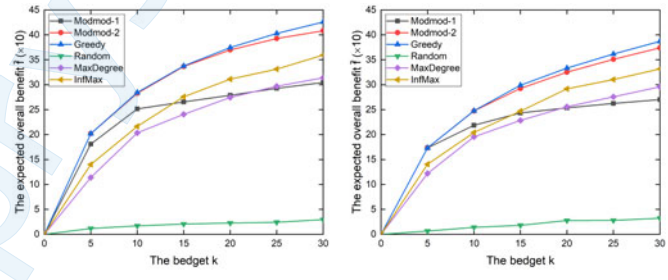


(a) Wikivot, $\theta = 5K$ (b) Wikivot, $\theta = 10K$

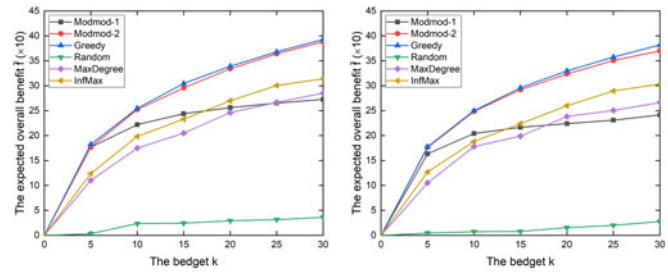


(c) Wikivot, $\theta = 15K$ (d) Wikivot, $\theta = 20K$

Fig. 5. The performance comparison with other heuristic algorithms under the Wikivot dataset.



(a) Bitcoin, $\theta = 5K$ (b) Bitcoin, $\theta = 10K$



(c) Bitcoin, $\theta = 15K$ (d) Bitcoin, $\theta = 20K$

Fig. 6. The performance comparison with other heuristic algorithms under the Bitcoin dataset.

inconspicuous from these figures. Then, we make the following observations. First, the expected overall benefit increases as the budget increases at least on a budget less than 30. Then, the performances achieved by Greedy and Modmod-2 algorithms are very close under the all datasets. The performances achieved by Modmod-1 algorithm are unstable under the different datasets, which have good results under the Netscie and Wikivot datasets but a bad result under the Bitcoin dataset. It implies that the selection of upper bounds is a critical factor that affects the results of the modular-modular procedure. In general, the performance of Modmod-2 is better than that of Modmod-1. A possible reason is that the second upper bound is tighter than the first upper bound, especially for using in larger datasets.

6.3 Approximation and Running Time

1) Approximation. The approximation ratios of our modular-modular procedure when $k = 20$ are shown in Table 3. From the Table 3, we can see that the approximation ratio improves as the number of RR-sets increases since the estimation errors in In Equ. (42) can be reduced gradually. Besides, the approximation has been improved more obviously with the increase of the number of RR-sets under the

TABLE 3
Approximation Ratios of Modular-Modular Proceduce When $k = 20$

	Netscie		Wikivot		Bitcoin	
θ	md-1	md-2	md-1	md-2	md-1	md-2
2 K	0.50	0.50	0.44	0.44	0.31	0.40
5 K	0.51	0.51	0.44	0.44	0.31	0.41
10K	0.50	0.50	0.47	0.47	0.31	0.42
15K	0.50	0.51	0.50	0.50	0.32	0.42
20K	0.52	0.53	0.51	0.51	0.32	0.45

TABLE 4
Running Time (Seconds) of Modular-Modular Proceduce When $k = 20$

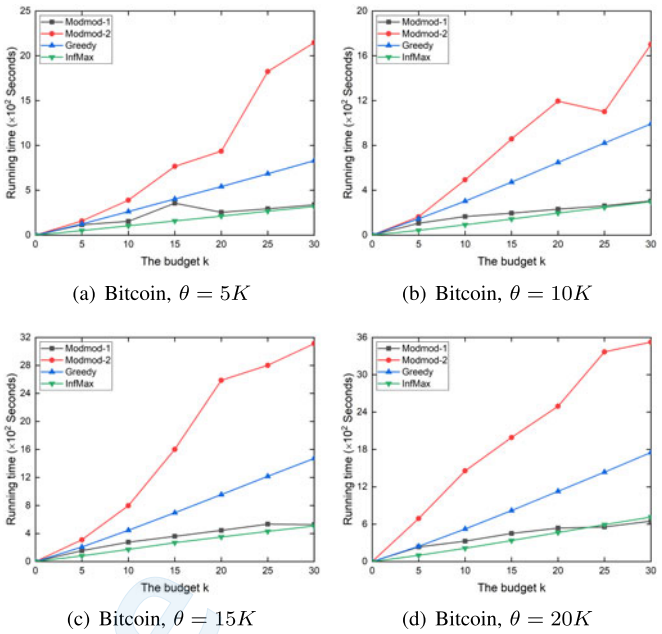
	Netscie		Wikivot		Bitcoin	
θ	md-1	md-2	md-1	md-2	md-1	md-2
2 K	03	05	10	052	092	0384
5 K	09	28	24	083	255	0935
10K	17	53	44	154	232	1195
15K	23	64	65	410	445	2587
20K	27	57	82	285	537	2493

Bitcoin dataset because a larger dataset requires a larger number of RR-sets to get an accurate estimation.

2) Running Time. The running times of our modular-modular procedure when $k = 20$ are shown in Table 4. From the Table 4, the running time increases as the number of RR-sets increases generally because the estimation of objective value is more time-consuming, which causes the modular maximization process shown as Algorithm 2 is more time-consuming. However, this is not strict to say that since the number of iterations varies under different circumstances, where Modmod-2 needs to update X^t more times than Modmod-1 actually. Fig. 7 shows the running time comparison with other heuristic algorithms under the Bitcoin dataset. Shown as Fig. 7, the running time of Modmod-2 is the highest among all these algorithms, but the running time of Modmod-1 lies between InfMax and Greedy. This is since Modmod-2 needs to be iterated more times to achieve convergence, which explains the reason why its performance is better than the performance of Modmod-1.

6.4 Further Discussion

According to the above analysis, we have known that the performances (expected overall benefits) obtained by Greedy and Modmod-2 algorithms are very similar, even Greedy algorithm sometimes performs better. In addition, Greedy algorithm also performs better in running time. Does this mean that our modular-modular procedure is meaningless? The answer is "No". For a non-monotone, non-submodular, and non-supermodular maximization problem, it is extremely difficult to solve it with a theoretical guarantee. Greedy is only a heuristic strategy that has no approximation guarantee, thus we cannot determine whether its solution is good or bad. Given our modular-modular procedure, we can obtain a worst approximation ratio that is around 0.5. If Greedy performs better than Modmod-2 algorithm, which at least shows that the approximation ratio of Greedy is greater than that of Modmod-2. This is equivalent to finding an apprxomation ratio for Greedy algorithm. Because of the high time complexity of modular-modular procedure, its theoretical value is greater than its practical application value. Therefore, how to reduce time complexity is a problem worth considering in the future. Also, this work can be used as a general framework for this kind of problems in social computing.



(a) Bitcoin, $\theta = 5K$ (b) Bitcoin, $\theta = 10K$

(c) Bitcoin, $\theta = 15K$ (d) Bitcoin, $\theta = 20K$

Fig. 7. The running time comparison with other heuristic algorithms under the Bitcoin dataset.

7 Conclusion

In this paper, we consider the disturbance of rival's influence on our benefits we can get from the influence diffusion in social networks and propose an OEBI problem formally, which is a generalization for a number of realistic scenarios. Then, we quantify this disturbance, define our objective function, and show its properties about monotonicity and submodularity. To solve it, we decompose it into the difference of two monotone and submodular functions, and apply modular-modular procedure to get a solution according to their lower bound and upper bound. Then, we design a series of efficient unbiased estimators to approximate it with a data-dependent approximation guarantee but reduce running time significantly. The approximations and running times are verified and analyzed by numerical simulations.

Our modular-modular procedure with sampling techniques can be considered as a general framework to address non-monotone and non-submodular maximization problem. However, its performance in running time is still not satisfactory, which is worth considering again in the future.

References

[1] Z. Zhang, R. Sun, X. Wang, and C. Zhao, "A situational analytic method for user behavior pattern in multimedia social networks," *IEEE Trans. Big Data*, vol. 5, no. 4, pp. 520–528, Dec. 2019.

[2] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2003, pp. 137–146.
[3] W. Lu and L. V. Lakshmanan, "Profit maximization over social networks," in *Proc. IEEE 12th Int. Conf. Data Mining*, 2012, pp. 479–488.
[4] Y. Dong, Z. Ding, F. Chiclana, and E. Herrera-Viedma , "Dynamics of public opinions in an online and offline social network," *IEEE Trans. Big Data*, vol. 7, no. 4, pp. 610–618, Oct. 2021.
[5] J. Guo, T. Chen, and W. Wu, "Continuous activity maximization in online social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2775–2786, Fourth Quarter 2020.
[6] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks," in *Proc. Int. Workshop Web Internet Econ.*, 2007, pp. 306–311.
[7] J. Guo and W. Wu, "A novel scene of viral marketing for complementary products," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 4, pp. 797–808, Aug. 2019.
[8] P. Wu and L. Pan, "Scalable influence blocking maximization in social networks under competitive independent cascade models," *Comput. Netw.*, vol. 123, pp. 38–50, 2017.
[9] N. Arazkhani, M. R. Meybodi, and A. Rezvanian, "An efficient algorithm for influence blocking maximization based on community detection," in *Proc. 5th Int. Conf. Web Res.*, 2019, pp. 258–263.
[10] G. A. Tong *et al.*, "An efficient randomized algorithm for rumor blocking in online social networks," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.
[11] J. Guo, T. Chen, and W. Wu, "A multi-feature diffusion model: Rumor blocking in social networks," *IEEE/ACM Trans. Netw.*, vol. 29, no. 1, pp. 386–397, Feb. 2021.
[12] R. Iyer and J. Bilmes, "Algorithms for approximate minimization of the difference between submodular functions, with applications," in *Proc. 28th Conf. Uncertainty Artif. Intell.*, 2012, pp. 407–417.
[13] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 1029–1038.
[14] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 88–97.
[15] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2014, pp. 946–957.
[16] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2007, pp. 420–429.
[17] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 199–208.
[18] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 75–86.
[19] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2015, pp. 1539–1554.
[20] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. Int. Conf. Manage. Data*, 2016, pp. 695–710.
[21] J. Tang, X. Tang, X. Xiao, and J. Yuan, "Online processing algorithms for influence maximization," in *Proc. Int. Conf. Manage. Data*, 2018, pp. 991–1005.
[22] W. Lu, W. Chen, and L. V. Lakshmanan, "From competition to complementarity: Comparative influence diffusion and maximization," *Proc. VLDB Endowment*, vol. 9, no. 2, pp. 60–71, 2015.
[23] G. Tong, R. Wang, and Z. Dong, "On multi-cascade influence maximization: Model, hardness and algorithmic framework," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1600–1613, 2021.
[24] J. Tang, X. Tang, and J. Yuan, "Profit maximization for viral marketing in online social networks," in *Proc. IEEE 24th Int. Conf. Netw. Protocols*, 2016, pp. 1–10.
[25] N. Buchbinder, M. Feldman, J. Seffi, and R. Schwartz, "A tight linear time (1/2)-approximation for unconstrained submodular maximization," *SIAM J. Comput.*, vol. 44, no. 5, pp. 1384–1402, 2015.
[26] G. Tong, W. Wu, and D.-Z. Du, "Coupon advertising in online social systems: Algorithms and sampling techniques," 2018, *arXiv: 1802.06946*.
[27] J. Guo, T. Chen, and W. Wu, "Budgeted coupon advertisement problem: Algorithm and robust analysis," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1966–1976, Third Quarter 2020.
[28] U. Feige and R. Izsak, "Welfare maximization and the supermodular degree," in *Proc. 4th Conf. Innovations Theor. Comput. Sci.*, 2013, pp. 247–256.
[29] Z. Wang, B. Moran, X. Wang, and Q. Pan, "Approximation for maximizing monotone non-decreasing set functions with a greedy method," *J. Combinatorial Optim.*, vol. 31, no. 1, pp. 29–43, 2016.
[30] M. Narasimhan and J. Bilmes, "A submodular-supermodular procedure with applications to discriminative structure learning," in *Proc. 21st Conf. Uncertainty Artif. Intell.*, 2005, pp. 404–412.
[31] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 4292–4293. [Online]. Available: http://networkrepository.com
[32] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," Jun. 2014. [Online]. Available: http://snap.stanford.edu/data

Jianxiong Guo received the BS degree in energy engineering and automation from the South China University of Technology, China, in 2015, and the MS degree in chemical engineering from the University of Pittsburgh, Pittsburgh, Pennsylvania, in 2016. He is currently working toward the PhD degree in the Department of Computer Science, University of Texas at Dallas, Richardson, Texas. His research interests include social networks, data mining, IoT application, blockchain, and combinatorial optimization.

Yapu Zhang received the BS degree in mathematics and applied mathematics from Northwest University, Xi'an, China, in 2016. She is currently working toward the PhD degree in the School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China. Her research interests include social networks and approximation algorithms.

Weili Wu (Senior Member, IEEE) received the MS and PhD degrees from the Department of Computer Science, University of Minnesota, Minneapolis, Minnesota, in 1998 and 2002, respectively. She is currently a full professor with the Department of Computer Science, The University of Texas at Dallas, Richardson, Texas. Her research mainly deals in the general research area of data communication and data management. Her research interests include design and analysis of algorithms for optimization problems that occur in wireless networking environments and various database systems.