Adaptive Influence Maximization: If Influential Node Unwilling to Be the Seed

JIANXIONG GUO and WEILI WU, The University of Texas at Dallas, USA

Influence maximization problem attempts to find a small subset of nodes that makes the expected influence spread maximized, which has been researched intensively before. They all assumed that each user in the seed set we select is activated successfully and then spread the influence. However, in the real scenario, not all users in the seed set are willing to be an influencer. Based on that, we consider each user associated with a probability with which we can activate her as a seed, and we can attempt to activate her many times. In this article, we study the adaptive influence maximization with multiple activations (Adaptive-IMMA) problem, where we select a node in each iteration, observe whether she accepts to be a seed, if yes, wait to observe the influence diffusion process; if no, we can attempt to activate her again with a higher cost or select another node as a seed. We model the multiple activations mathematically and define it on the domain of integer lattice. We propose a new concept, adaptive dr-submodularity, and show our Adaptive-IMMA is the problem that maximizing an adaptive monotone and dr-submodular function under the expected knapsack constraint. Adaptive dr-submodular maximization problem is never covered by any existing studies. Thus, we summarize its properties and study its approximability comprehensively, which is a non-trivial generalization of existing analysis about adaptive submodularity. Besides, to overcome the difficulty to estimate the expected influence spread, we combine our adaptive greedy policy with sampling techniques without losing the approximation ratio but reducing the time complexity. Finally, we conduct experiments on several real datasets to evaluate the effectiveness and efficiency of our proposed policies.

CCS Concepts: • **Networks** → **Network algorithms**; • **Theory of computation** → **Design and analysis of algorithms**

Additional Key Words and Phrases: Adaptive influence maximization, social networks, integer lattice, adaptive dr-submodularity, sampling techniques, approximation algorithm

ACM Reference format:
Jianxiong Guo and Weili Wu. 2021. Adaptive Influence Maximization: If Influential Node Unwilling to Be the Seed. *ACM Trans. Knowl. Discov. Data.* 15, 5, Article 84 (May 2021), 23 pages.
https://doi.org/10.1145/3447396

1 INTRODUCTION

Online social networks (OSNs) were blossoming prosperously in recent decades and have become the main means of communication between people such as Wechat, Facebook, Twitter, and LinkedIn. More and more people participate to discuss the topics that they are interested in on

This work is partly supported by the National Science Foundation under grants 1747818 and 1907472.
Authors' address: J. Guo (corresponding author) and W. Wu, Department of Computer Science, The University of Texas at Dallas, 800 W Campbell Rd, Richardson, Texas, 75080; emails: {jianxiong.guo, weiliwu}@utdallas.edu.

1556-4681/2021/05-ART84 $15.00
https://doi.org/10.1145/3447396

these social platforms. Many companies or advertisers exploit the relations established in OSNs to spread their products, opinions, or innovations. They provide those influential individuals (called "seed nodes") with free or discounted samples, in order to create widespread influence across the whole network via word-of-mouth effect [7], [20]. Based on that, **influence maximization (IM)** problem [17] was formulated, which selects a subset of users (called "seed set") for an information cascade to maximize the expected follow-up adoptions (influence spread). It is a general model for a number of realistic scenarios, such as viral marketing. In [17], they created two discrete influence diffusion models, **independent cascade model (IC-model)** and **linear threshold model (LT-model)**, where IC-model relies on peer-to-peer communication but LT-model computes the total influence from user's neighbors. Then, they proved that the IM problem is NP-hard and provided a $(1-1/e)$-approximate algorithm by simple the greedy strategy in the framework of submodularity. After this groundbreaking work, a sequence of derivative problems appeared and were solved under the different constraints and scenarios [2], [3], [11], [12], such as profit maximization [13], community partition, and rumor blocking (detection).

Despite these developments, the existing research studies on the IM problem have a crucial drawback. When selecting a seed set at the beginning, they all seem to take it for granted that every user in their selected seed set can be activated successfully to become an active seed and then spread the influence as they wish. However, there are some users unwilling, even impossible, to be the influencers. For example, the hottest topic at the moment, Coronavirus in Wuhan, China. Some non-profit organizations or official media are trying to make celebrities speak out to ease the panic among the people. However, due to self-interest or other factors, some celebrities are not willing to be their "seed nodes." Then, we have two options, one is trying to persuade those who stand on the opposite side sentimentally and rationally, the other is giving up and look for other potential "seed nodes." Based on that, we design a new **IM with multiple activations (IMMA)** problem, where a node can be activated to be an influencer with a probability when we select it as a seed and we can attempt to activate it many times. For the same node, each attempt is referred to as a "trial" and each trial has a cost. If the first trial fails, we can conduct the second trial, the third trial, and so on, but their cost is higher than the first.

Most existing techniques on the IM problem concentrate on non-adaptive strategies, which are required to select all seed nodes at once without observing actual node status and diffusion process. In other words, we need to point out the seed set and the number of trials for each node in this seed set in one batch. As a result, it may return a seed that cannot be activated actually or assign too many trials to this node. For example, we give a seed three trials and it is activated in the first trial, so the remaining two waste our budget. Thus, the non-adaptive seeding strategy is not the best choice to solve our IMMA problem. Golovin et al. [8] studies the IM problem under the adaptive strategy, they select the $(i+1)$-th node after observing the influence diffusion of the first i nodes until all seed nodes are chosen. Based on that, the Adaptive-IMMA problem is proposed in this article, which selects a seed and attempts to activate it in each iteration. If successful, wait to observe its influence process; if failed, record this failed trial. Those nodes on which all the trials are unsuccessful can be considered as seed again in a later step.

Golovin et al. [8] provided a $(1-1/e)$-approximate algorithm by an adaptive greedy policy for the adaptive IM problem in the framework of adaptive monotonicity and adaptive submodularity. However, for our Adaptive-IMMA problem, its solution is a seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$, not a seed set $S \subseteq V$, where each component $\boldsymbol{x}(u)$ means how many activation attempts we give to user u. It will be executed sequentially. For example, $\boldsymbol{x}(u) = i$, we will do trial $\langle u, 1\rangle$, trial $\langle u, 2\rangle$ until trial $\langle u, i\rangle$ on user u. The domain of the objective function is defined on integer lattice, not generally on set. Thus, traditional analytical methods based on adaptive submodularity cannot be applied to analyze our problem. The submodularity shows us with diminishing marginal gain property

in set function. For functions defined on integer lattice, such a property exists as well, called dr-submodularity. Based on that, we define the concepts of adaptive monotonicity on integer lattice and adaptive dr-submodularity formally, which are extended from adaptive submodularity on set function [8] and dr-submodularity on integer lattice [22]. Then, we formulate the objective function of our Adaptive-IMMA problem and prove it is adaptive monotone on integer lattice as well as adaptive dr-submodular. Each trial $\langle u, i\rangle$ is associated with a cost $c(\langle u, i\rangle)$ and the costs of different trials are different. Given a randomized policy $\pi(\kappa)$, the total budget k is an expected knapsack constraint such that the total cost of the seeding vector returned by $\pi(\kappa)$ is less than k expectedly. Then, we study the approximate performance of adaptive greedy policy for maximizing adaptive monotone and adaptive dr-submodular functions under the expected knapsack constraint, which is a non-trivial generalization of existing analysis about the approximate performance of adaptive submodular functions. Assume $c(\langle u, i\rangle) \leq c(\langle u, i+1\rangle)$, it returns an acceptable solution with $(1-1/e)$-approximate ratio.

Besides, it is #P-hard to compute the expected influence spread given a seed set under the IC-model [4] and LT-model [6]. The complexity of our objective function is higher. In order to overcome this shortcoming, we design an unbiased estimator of the conditional expected marginal gain for our problem based on the **reverse influence sampling (RIS)** [1]. Adapted from state-of-the-art EPIC algorithm [16] for the IM problem, combined it with our adaptive greedy policy, we formulate a sampled adaptive greedy policy and achieve a $(1-\exp(-1+\varepsilon))$ expected approximation guarantee. Its time complexity is reduced significantly. Finally, we conduct several experiments to evaluate the superiority of our adaptive policies over their corresponding non-adaptive algorithms and the sampled adaptive greedy policy over other heuristic adaptive policies, which support the effectiveness and efficiency of our approaches strongly.

2 RELATED WORK

IM: The IM problem has been studied extensively. Kempe et al. [17] formulated IM as a combinatorial optimization problem, proposed the triggering model, including IC-model and LT-model, and gave us a greedy algorithm with $(1-1/e-\varepsilon)$-approximation. It was implemented by **Monte Carlo (MC)** simulations with high time complexity. Chen et al. [4], [6] followed Kempe's work and proved its #P-hardness to compute the expected influence spread. Thus, the running time was too slow to apply to larger real networks. After those seminal works, a lot of researchers made an effort to improve its time complexity. Brogs et al. [1] proposed the concept of RIS to estimate the expected influence spread, which is scalable in practice and guaranteed theoretically at the same time. Then, a series of more efficient randomized algorithms were arisen, such as TIM/TIM+ [30], IMM [29], SSA/D-SSA [19], and OPIM-C [27]. Recently, Han et al. [14] provided us with an EPIC algorithm with an expected approximation guarantee. Then, the EPIC was improved further based on the OPIM-C [27], which is the most efficient algorithm to solve the IM problem until now [16]. They were scalable algorithms for the IM problem and can be adapted to other relative problems. However, all of these are used to solve the IM problem under the non-adaptive setting.

Dr-submodular maximization and its applications in social networks: Defined on integer lattice, dr-submodular maximization problem is a hot topic that attracts a lot of researchers recently. Soma et al. [22] formalized dr-submodularity on integer lattice inspired by the diminishing return property on set and addressed a submodular cover problem. Soma et al. [24] studied the monotone dr-submodular maximization problem on integer lattice systematically, where they proposed a series of $(1-1/e-\varepsilon)$-approximate algorithms for the cardinality constraint, the knapsack constraint, and the polymatroid constraint. Applied to solve problems social networks, Chen et al. [5] gave a lattice IM problem defined on integer lattice, whose objective function is monotone and dr-submodular. Then, they proposed a scalable algorithm with $(1-1/e-\varepsilon)$ approximation ratio that

is adapted from the IMM [29]. Guo et al. [10] proposed a continuous activity maximization problem and provided a solution framework for maximizing a monotone but not dr-submodular function by the sandwich approximation approach. Other literature and results about dr-submodular maximization and its applications are shown in [9], [23], [15].

Adaptive influence maximization: Golovin et al. [8] extended the submodularity to adaptive settings and obtained the same approximation ratio for the adaptive IM problem because its objective function is adaptive monotone and adaptive submodular under the full-adoption feedback model where only one node can be selected in each iteration. However, the objective function of the adaptive IM problem is not adaptive submodular under the myopic feedback model [8] or the partial feedback model [32], [28]. Tong et al. [31] gave us a systematic framework about the adaptive IM problem when it is not adaptive submodular, where they designed a seeding strategy and showed the approximation analysis by introducing the concept of regret ratio. Unfortunately, all these works were based on a fact that the expected influence spread can be computed accurately in polynomial time, which is an unrealistic assumption. To improve its scalability, Han et al. [14] proposed an AdaptGreedy framework instantiated by scalable IM algorithms to solve the adaptive IM problem where it can select a batch of seed nodes in each iteration, which returns a worst-case approximation guarantee with high probability. Sun et al. [25] studied a **multi-round influence maximization (MRIM)** problem where information diffuses in multiple rounds independently from different seed sets. They considered MRIM problem under the adaptive setting and designed an adaptive algorithm with $(1-\exp(1/e-1)-\varepsilon)$ approximation instantiated by the IMM [29]. Tang et al. [26] considered the seed minimization problem under the adaptive setting and proposed an ASTI algorithm that offers an expected approximation ratio. Recently, Huang et al. [16] pointed out there are some mistakes in the approximation analysis of adaptive policies in [14], [25]. They fixed the previous AdaptGreedy framework in [14] and proved it has a $(1-\exp(\rho_b(\varepsilon-1)))$ expected approximation guarantee instantiated by their improved EPIC in [16].

Nevertheless, none of them considered a problem that is adaptive dr-submodular, especially under the expected knapsack constraint. This is the main contribution in this article.

3 PROBLEM FORMULATION

In this section, we define the problem of our adaptive influence maximization on multiple activations formally and introduce some preliminary knowledges.

3.1 Influence Model and Graph Realization

A social network can be denoted by a directed graph $G = (V, E)$ where $V = \{v_1, v_2, \ldots, v_n\}$ is the node (user) set with $|V| = n$, $E = \{e_1, e_2, \ldots, e_m\}$ is the edge set with $|E| = m$, which describes the relationship between users. For any edge $e = (u, v) \in E$, v is an outgoing neighbor of u and u is an incoming neighbor of v. For any node $v \in N$, we denote by $N^-(v)$ its set of incoming neighbors and $N^+(v)$ its set of outgoing neighbors. Each edge (u, v) is associated with a diffusion probability $p_{uv} \in (0, 1]$. Thus, the influence diffusion on this network is stochastic.

Let $S \subseteq V$ be a given node set, the influence diffusion initiated by S can be described as a discrete-time stochastic process under the IC-model [17]. Let $S_i \subseteq V$ be the active node set at time step t_i. At time step t_0, all nodes in S are active, namely $S_0 := S$. We call S_0 the seed set, and node in this set is the seed of this cascade. At time step t_i, $i \geq 1$, we set $S_i := S_{i-1}$ first; then, for those nodes activated first at time step t_{i-1}, $u \in (S_{t-1} \backslash S_{t-2})$, it activates its each inactive outgoing neighbor v with the probability p_{uv} by one chance. If u activates v at t_i successfully, we add v into S_i. The influence diffusion terminates when no more inactive nodes can be activated.

The above influence diffusion process can be interpreted by sampling a graph realization. Given a directed network $G = (V, E)$, we can decide whether an edge $(u, v) \in E$ is live or blocked with

probability p_{uv}. To remove these blocked edges, the remaining graph is a subgraph g of G. This subgraph g is called "graph realization." These edges existed in g are known as live edges, or else called blocked edges. For each edge $(u, v) \in E$, it exists in a graph realization g with probability p_{uv} under the IC-model. There are 2^m possible graph realizations altogether under the IC-model. Let $\mathcal{G}$ be the set of all possible graph realizations with $|\mathcal{G}| = 2^m$, and g be a graph realization sampled from $\mathcal{G}$, denoted by $g \leftarrow \mathcal{G}$, with probability as follows:

$$\Pr[g|g \leftarrow \mathcal{G}] = \prod_{e \in E(g)} p_e \prod_{e \in E(G) \setminus E(g)} (1 - p_e). \tag{1}$$

Remark 1. In most references, they usually called "graph realization" as "realization" or "possible world." They all refer to an instance of a probabilistic social network. We will discuss a different concept "realization" later. To avoid ambiguity, we use "graph realization" here.

The problem and algorithms discussed later in this article are based on the IC-model, because the adaptive IM problem can be adaptive submodular only under the IC-model.

3.2 Adaptive Influence Maximization

Given a seed set $S \subseteq V$ and a graph realization $g \in \mathcal{G}$, the size of final active set that can be reached by the seed set S under the graph realization g is denoted by $\sigma_g(S)$. Thus, the expected influence spread $\sigma_G(S)$ under the IC-model can be defined as follows:

$$\sigma_G(S) = \mathbb{E}_{g \leftarrow \mathcal{G}}[\sigma_g(S)] = \sum_{g \in \mathcal{G}} \sigma_g(S) \cdot \Pr[g|g \leftarrow \mathcal{G}], \tag{2}$$

where it is the weighted expectation of influence spread over all possible graph realizations. The IM problem aims to find a seed set $S \subseteq V$, such that $|S| \leq k$, to maximize the expected influence spread $\sigma_G(S)$.

From the above, in this non-adaptive setting, the seed set S is selected once without the knowledge of what graph realization happens in the actual diffusion process. Thus, the actual influence spread of S may be much worse than our expectation. Instead, in an adaptive manner, we select a node u from V at a time and wait to observe the actual diffusion result. Relied on this observation, we select the next node that could activate those inactive nodes as much as possible. It is called full-adoption feedback model [8]. In other words, when we select a node u as seed, we are able to know the status of all edges going out from those nodes that can be reached by u through live edges in current graph realization. Golovin et al. [8] introduced two important concepts, adaptive monotonicity and adaptive submodularity, and showed that the simple adaptive greedy policy has a $(1 - 1/e)$-approximation guarantee.

3.3 Problem Definition

In the traditional IM problem, it assumes that each user in the seed set we select is activated successfully and then spread our given information cascade. However, in the real scenario, not all users in the seed set are willing to be an influencer. Based on that, we consider a user can be activated as a seed with a certain probability and we can attempt to activate her many times. For each user $u \in V$, there is a probability $\beta_u \in (0, 1]$ with which she can be activated successfully when we select her as a seed. Let $\mathbb{Z}_+^V$ be the collection of non-negative integer vector, each component is indexed by a node in V. For a vector $\boldsymbol{x} \in \mathbb{Z}_+^V$, if $\boldsymbol{x}(u) = i$, it means that we select user u and try to activate her as a seed i times.

Given a social graph $G = (V, E)$, we have a total budget $k \in \mathbb{R}_+$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$, and a cost function $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$. Here, for each user $u \in V$, we assume she can be tried to activate as a seed at most $\boldsymbol{b}(u)$ times, and it costs $c(\langle u, i \rangle)$ when the i-th trial of activating user u as a seed

happens. Given a seeding vector $\boldsymbol{x}$ and a graph realization $g \in \mathcal{G}$, the expected number of active nodes $\mu_g(\boldsymbol{x})$ under the graph realization g can be defined as follows:

$$\mu_g(\boldsymbol{x}) = \mathbb{E}_{S \leftarrow \boldsymbol{x}}[\sigma_g(S)] = \sum_{S \subseteq V} \sigma_g(S) \cdot \Pr[S|S \leftarrow \boldsymbol{x}] \tag{3}$$

$$= \sum_{S \subseteq V} \sigma_g(S) \cdot \prod_{u \in S} \left(1 - (1 - \beta_u)^{\boldsymbol{x}(u)}\right) \cdot \prod_{u \in V \setminus S} (1 - \beta_u)^{\boldsymbol{x}(u)}. \tag{4}$$

Similar to Equation (2), we have

$$\mu_G(\boldsymbol{x}) = \mathbb{E}_{g \leftarrow \mathcal{G}}[\mu_g(\boldsymbol{x})] = \sum_{g \in \mathcal{G}} \mu_g(x) \cdot \Pr[g|g \leftarrow \mathcal{G}], \tag{5}$$

where $\mu_G(\boldsymbol{x})$ is the expected influence spread over all possible graph realizations given a seeding vector $\boldsymbol{x}$. The IMMA problem is formulated, which seeks a seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$ that maximizes $\mu_G(\boldsymbol{x})$ subject to $c(\boldsymbol{x}) \leq k$ and $\boldsymbol{x} \leq \boldsymbol{b}$. Here, we denote $c(\boldsymbol{x})$ by $c(\boldsymbol{x}) = \sum_{u \in V} \sum_{i \in [\boldsymbol{x}(u)]} c(\langle u, i \rangle)$, where $[j] = \{1, 2, \ldots, j\}$. Each trial is independent.

In the adaptive setting, the IMMA problem can be transformed to find a policy π, where we select seed nodes step by step. The parameter setting is the same as before. A seeding vector is initialized to $\boldsymbol{x} = \boldsymbol{0} \in \mathbb{Z}_+^V$. When selecting an inactive user $u \in V$ with $\boldsymbol{x}(u) < \boldsymbol{b}(u)$, we increase $\boldsymbol{x}(u)$ by 1 and attempt to activate u to be an active seed with probability β_u. At this moment, we need to observe two states as follows: (1) Node state: whether user u can be activated to be an active seed successfully; and (2) Edge state: If u becomes an active seed, wait to observe the influence diffusion process (related edges is live or blocked) until no new nodes can be activated. We repeated this process until no remaining budget exists.

Next, we define the states of the given network. Given a social graph $G = (V, E)$, for each node $u \in V$, the state of u can be denoted by $X_u \in \{0, 1, ?\}^{\boldsymbol{b}(u)}$, where $X_u(i) = 1$ means user u is activated as a seed successfully in the i-th trial, or not succeed, $X_u(i) = 0$. $X_u(i) =?$ if the result of i-th trial is unknown. Similar, for each edge $(u, v) \in E$, the state of (u, v) can be denoted by $Y_{uv} \in \{0, 1, ?\}$, where $Y_{uv} = 1$ means edge (u, v) is live, and $Y_{uv} = 0$ means edge (u, v) is blocked. $Y_{uv} =?$ if the state of (u, v) is unknown. At the beginning, the states of all nodes and edges are ?. After defining the state variables, we have a function ϕ mapping like

$$\phi : \{X_u\}_{u \in V} \cup \{Y_{uv}\}_{(u,v) \in E} \rightarrow \left\{\{0, 1\}^{\boldsymbol{b}(u)}\right\}_{u \in V} \cup \{0, 1\}^E, \tag{6}$$

where ϕ is called a realization (full realization), where the states of all items are known. We say that $\phi(u) \in \{0, 1\}^{\boldsymbol{b}(u)}$ is the state of user $u \in V$, $\phi(u)(i) \in \{0, 1\}$ is the state of the i-th trial for user u, and $\phi((u, v)) \in \{0, 1\}$ is the state of edge $(u, v) \in E$ under the realization ϕ. Let Φ be the set of all possible realizations. We define ϕ as a realization sampled from Φ, denoted by $\phi \leftarrow \Phi$, with probability $\Pr[\phi|\phi \leftarrow \Phi]$. That is

$$\Pr[\phi|\phi \leftarrow \Phi] = \prod_{\substack{e \in E \\ \phi(e)=1}} p_e \prod_{\substack{e \in E \\ \phi(e)=0}} (1 - p_e) \cdot \prod_{u \in V} \left[\prod_{\substack{i \in [\boldsymbol{b}(u)] \\ \phi(u)(i)=1}} \beta_u \prod_{\substack{i \in [\boldsymbol{b}(u)] \\ \phi(u)(i)=0}} (1 - \beta_u) \right]. \tag{7}$$

In this adaptive seeding process, after the i-th trial to activate node u is finished, its state and the states of those related edges could be updated. Our observation until now can be described by a partial realization ψ. It is a function of observed items to their states. For $u \in V$, $\psi(u) \in \{0, 1, ?\}^{\boldsymbol{b}(u)}$, and $\psi(u)(i) =?$ if the result i-th trial to node u is not yet observed. For $(u, v) \in E$, $\psi((u, v)) \in \{0, 1, ?\}$ as well. The domain of a partial realization ψ can be defined as $\text{dom}(\psi) = \{\langle u, i \rangle | \psi(u)(i) \neq ?\}$, which is the trials that have been done. We say ψ is consistent with a realization ϕ if the states

of items in the domain of ψ are equal between them, denoted by $\phi \sim \psi$. Given ψ, ψ' and ϕ, if $\mathrm{dom}(\psi) \subseteq \mathrm{dom}(\psi')$ and $\phi \sim \psi, \psi'$, we say ψ is a subrealization of ψ', denoted by $\psi \subseteq \psi'$.

Let $\pi(\kappa)$ be a randomized policy based on a random variable κ that represents a random source of this randomized policy. The $\pi(\kappa)$ is a function mapping from current seeding vector $\boldsymbol{x}$ and one of its possible partial realizations ψ to a node u^*, then it executes the $(\boldsymbol{x}(u^*)+1)$-th trial that tries to select node u^* as a seed. Here, we denote by $u^* = \pi(\kappa, \boldsymbol{x}, \psi)$ where u^* is the next potential seed that policy $\pi(\kappa)$ will select based on $\boldsymbol{x}$ and ψ. The influence spread gained from policy $\pi(\kappa)$ under the realization ϕ can be defined as follows:

$$f(\eta(\pi(\kappa), \phi), \phi) = \sigma_{g_\phi}\left(\{u | \exists_{1 \le j \le \eta(\pi(\kappa),\phi)(u)} \phi(u)(j) = 1\}\right), \tag{8}$$

where $g_\phi \in \{0,1\}^E$ is the graph realization g contained in realization ϕ and $\eta(\pi(\kappa), \phi)$ is the seeding vector returned by policy π under the realization ϕ. The expected influence spread of policy $\pi(\kappa)$ can be shown as follows:

$$\mathbb{E}_\kappa\left[f_{avg}(\pi(\kappa))\right] = \mathbb{E}_\kappa\left[\mathbb{E}_{\phi \leftarrow \Phi}\left[f(\eta(\pi(\kappa), \phi), \phi)\right]\right]. \tag{9}$$

Therefore, the adaptive IM on multiple activations (Adaptive-IMMA) problem is formulated, which can be defined in Problem 1.

Problem 1 (Adaptive-IMMA Problem). *Given a social graph $G = (V, E)$, a budget $k \in \mathbb{R}_+$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$, and a cost function $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$, it aims to find a policy $\pi^*(\kappa)$ that maximizes its expected influence spread defined in Equation (9), i.e., $\pi^* \in \arg\max_\pi \mathbb{E}_\kappa[f_{avg}(\pi(\kappa))]$ subject to $\eta(\pi(\kappa), \phi) \le \boldsymbol{b}$ and $\mathbb{E}_\kappa[c(\eta(\pi(\kappa), \phi))] \le k$ for all realizations ϕ.*

Given a seed vector $\boldsymbol{x} \in \mathbb{Z}_+^V$, we say "increase $\boldsymbol{x}(u)$ by 1" is equivalent to execute the trial $\langle u, \boldsymbol{x}(u)+1\rangle$. For each node $u \in V$, we assume $c(\langle u,1\rangle) \le c(\langle u,2\rangle) \le \cdots \le c(\langle u, \boldsymbol{b}(u)\rangle)$. It is valid becuase in general, we execute trial $\langle u, i+1\rangle$ only when trial $\langle u, i\rangle$ fails to activate node u as a seed, thereby the cost of trial $\langle u, i+1\rangle$ is larger than the cost of $\langle u, i\rangle$.

4 THE PROPERTIES

In this section, we first introduce some concepts of submodularity on integer lattice, and then, generalize several properties of our Adaptive-IMMA problem.

4.1 Submodular Function on Integer Lattice

Usually, for two sets $S, T \subseteq V$, a set function $h : 2^V \to \mathbb{R}_+$ is monotone if $h(S) \le f(T)$ for any $S \subseteq T \subseteq V$ and submodular if $h(S) + h(T) \ge h(S \cup T) + h(S \cap T)$. The submodularity of set function can be generalized by diminishing return property, in other words, submodular if $h(S \cup \{u\}) - f(S) \ge f(T \cup \{u\}) - f(T)$ for any $S \subseteq T \subseteq V$ and $u \notin T$. On integer lattice, for two vectors $\boldsymbol{s}, \boldsymbol{t} \in \mathbb{Z}_+^V$, let $\boldsymbol{s} \vee \boldsymbol{t} \in \mathbb{Z}_+^V$ be defined as $(\boldsymbol{s} \vee \boldsymbol{t})(u) = \max\{\boldsymbol{s}(u), \boldsymbol{t}(u)\}$, and $\boldsymbol{s} \wedge \boldsymbol{t} \in \mathbb{Z}_+^V$ be defined as $(\boldsymbol{s} \wedge \boldsymbol{t})(u) = \min\{\boldsymbol{s}(u), \boldsymbol{t}(u)\}$ for any $u \in V$. A vector function $f : \mathbb{Z}_+^V \to \mathbb{R}_+$ is defined on the integer lattice $\mathbb{Z}_+^V$. This vector function f is monotone if $f(\boldsymbol{s}) \le f(\boldsymbol{t})$ for any $\boldsymbol{s} \le \boldsymbol{t} \in \mathbb{Z}_+^V$ and lattice submodular if $f(\boldsymbol{s}) + f(\boldsymbol{t}) \ge f(\boldsymbol{s} \vee \boldsymbol{t}) + f(\boldsymbol{s} \wedge \boldsymbol{t})$ for any $\boldsymbol{s}, \boldsymbol{t} \in \mathbb{Z}_+^V$. When the domain of vector are restricted to binary lattice $\{0,1\}^V$, the vector function f is reduced to set function h. Thus, the submodularity on set function is a special case of submodularity on integer lattice.

Besides, we consider a vector function $f : \mathbb{Z}_+^V \to \mathbb{R}_+$ is diminishing return submodular (dr-submodular) if $f(\boldsymbol{s} + \boldsymbol{e}_u) - f(\boldsymbol{s}) \ge f(\boldsymbol{t} + \boldsymbol{e}_u) - f(\boldsymbol{t})$ for any $\boldsymbol{s} \le \boldsymbol{t}$ and $u \in V$, where $\boldsymbol{e}_u \in \mathbb{Z}^V$ is the u-th unit vector with the u-th component being 1 and others being 0. Here, there is a little different from the submodularity on set function. f is lattice submodular does not mean it is dr-submodular on integer lattice, but the opppsite is true. Thus, dr-submodularity is a stronger property than lattice submodular. We consider the dr-submodularity later.

4.2 Properties of the Adaptive-IMMA

Assume that $\beta_u = 1$ for each node $u \in V$ and seeding vector $\boldsymbol{x} \in \{0,1\}^V$, the IMMA problem can be reduced to the IM problem naturally. Therefore, the IMMA problem is more general and inherits the NP-hardness of IM. In the traditional IM problem, the expected influence spread shown as Equation (2) is monotone and submodular on the seed set [17]. In order to study the properties of our Adaptive-IMMA problem, we define its marginal gain first, that is

Definition 1 (Conditional Expected Marginal Gain on Integer Lattice). Given a seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$ and a partial realization ψ generated by it, the conditional expected marginal gain of increasing $\boldsymbol{x}(u)$ by 1 is defined as

$$\Delta(u|\boldsymbol{x},\psi) = \mathbb{E}_{\phi\sim\psi}[f(\boldsymbol{x}+\boldsymbol{e}_u,\phi) - f(\boldsymbol{x},\phi)], \tag{10}$$

where the expectation is on $\Pr[\phi|\phi\sim\psi]$. The conditional expected marginal gain of policy $\pi(\kappa)$ is defined as

$$\Delta(\pi(\kappa)|\boldsymbol{x},\psi) = \mathbb{E}_{\phi\sim\psi}[f(\boldsymbol{x}\vee\eta(\pi(\kappa),\phi),\phi) - f(\boldsymbol{x},\phi)]. \tag{11}$$

Here, $\Delta(u|\boldsymbol{x},\psi)$ is the expected gain by increasing $\boldsymbol{x}(u)$ by 1 conditioned on current partial realization ψ of $\boldsymbol{x}$ and $\Delta(\pi(\kappa)|\boldsymbol{x},\psi)$ is the expected gain by running $\pi(\kappa)$ after observing partial realization ψ but neglect it. Then, adapted from [8], the concepts of adaptive monotonicity and adaptive submodularity are described as follows:

Definition 2 (Adaptive Monotonicity). A vector function $f(\cdot,\phi)$ is adaptive monotone if the conditional expected marginal gain with respect to distribution $\Pr[\phi]$ of any node u, seeding vector $\boldsymbol{x}$, and its possible partial realization ψ is nonnegative, that is

$$\Delta(u|\boldsymbol{x},\psi) \geq 0. \tag{12}$$

Definition 3 (Adaptive dr-submodularity). A vector function $f(\cdot,\phi)$ is adaptive dr-submodular if the conditional expected marginal gain with respect to distribution $\Pr[\phi]$ of any node u, seeding vectors $\boldsymbol{x},\boldsymbol{y}$ with $\boldsymbol{x} \leq \boldsymbol{y}$, and their possible partial realizations ψ (generated by $\boldsymbol{x}$), ψ' (generated by ψ') with $\psi \subseteq \psi'$ satisfies the following inequality, that is

$$\Delta(u|\boldsymbol{x},\psi) \geq \Delta(u|\boldsymbol{y},\psi'). \tag{13}$$

For our Adaptive-IMMA problem, the function $f(\cdot,\phi)$ is adaptive monotone and adaptive submodular according to Theorem 1 and 2.

Theorem 1. *The objective function $f(\cdot,\phi)$ is adaptive monotone.*

Proof. To prove adaptive monotonicity, we are required to show $\Delta(u|\boldsymbol{x},\psi) \geq 0$. Given a seeding vector $\boldsymbol{x}$ and its partial realization ψ, we denote the marginal gain under the realization $\phi\sim\psi$ as follows:

$$\Delta(u|\boldsymbol{x},\phi\sim\psi) = f(\boldsymbol{x}+e_u,\phi) - f(\boldsymbol{x},\phi). \tag{14}$$

If node u has been activated as a seed under the partial realization ψ, there is no marginal gain. Otherwise, it is possible to be activated by increasing $\boldsymbol{x}(u)$ by 1, namely trial $\langle u, \boldsymbol{x}(u)+1\rangle$ succeeds. Thus, we have $\Delta(u|\boldsymbol{x},\phi\sim\psi) \geq 0$. The conditional expected marginal gain $\Delta(u|\boldsymbol{x},\psi)$ is a linear combination of all realizations $\phi\sim\psi$, thereby we have $\Delta(u|\boldsymbol{x},\psi) \geq 0$. □

Theorem 2. *The objective function $f(\cdot,\phi)$ is adaptive dr-submodular.*

Proof. To prove its adaptive dr-submodularity, we are required to show $\Delta(u|\boldsymbol{x},\psi) \geq \Delta(u|\boldsymbol{y},\psi')$ for any two partial realizations ψ,ψ' such that $\boldsymbol{x} \leq \boldsymbol{y}$ and $\psi \subseteq \psi'$. Considering two fixed partial

realizations with $\psi \subseteq \psi'$, which are generated by seeding vector $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively. We defined the generative active seed set S under the seeding vector $\boldsymbol{x}$ and its partial realization ψ as

$$S(\boldsymbol{x}, \phi \sim \psi) = \left\{ u \in V | \exists_{1 \le j \le \boldsymbol{x}(u)} \phi(u)(j) = 1 \right\}. \tag{15}$$

Obviously, we have $S(\boldsymbol{x}, \phi \sim \psi) \subseteq S(\boldsymbol{y}, \phi' \sim \psi')$ because of $\boldsymbol{x} \le \boldsymbol{y}$ and $\psi \subseteq \psi'$. Here, we assume two fixed realizations, $\phi \sim \psi$, $\phi' \sim \psi'$, and $d(u) = \boldsymbol{y}(u) - \boldsymbol{x}(u)$. For each $\langle u, i \rangle \notin \mathrm{dom}(\psi')$, we have $\phi(u)(i - d(u)) = \phi'(u)(i)$; for each $(u, v) \notin g(\psi')$, we have $\phi((u, v)) = \phi'((u, v))$. We define the area that these two fixed realizations ϕ, ϕ' share as α. To show $\Delta(u|\boldsymbol{x}, \phi \sim \psi) \ge \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$, we can consider these three cases:

(1) $u \in S(\boldsymbol{x}, \phi \sim \psi)$: We have $S(\boldsymbol{x}, \phi \sim \psi) = S(\boldsymbol{x} + \boldsymbol{e}_u, \phi \sim \psi)$ and $S(\boldsymbol{y}, \phi' \sim \psi') = S(\boldsymbol{y} + \boldsymbol{e}_u, \phi' \sim \psi')$. Thus, $\Delta(u|\boldsymbol{x}, \phi \sim \psi) = \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$.
(2) $u \in S(\boldsymbol{y}, \phi' \sim \psi') \backslash S(\boldsymbol{x}, \phi \sim \psi)$: We have $S(\boldsymbol{x}, \phi \sim \psi) \subseteq S(\boldsymbol{x} + \boldsymbol{e}_u, \phi \sim \psi)$ and $S(\boldsymbol{y}, \phi' \sim \psi') = S(\boldsymbol{y} + \boldsymbol{e}_u, \phi' \sim \psi')$. Thus, $\Delta(u|\boldsymbol{x}, \phi \sim \psi) \ge \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$.
(3) $u \in V \backslash S(\boldsymbol{y}, \phi' \sim \psi')$: When $\boldsymbol{x}(u) = \boldsymbol{y}(u) = i$, we have $S(\boldsymbol{x} + \boldsymbol{e}_u, \phi \sim \psi) = S(\boldsymbol{x}, \phi \sim \psi) \cup \{u\}$ and $S(\boldsymbol{y} + \boldsymbol{e}_u, \phi' \sim \psi') = S(\boldsymbol{y}, \phi' \sim \psi') \cup \{u\}$ if $\phi(u)(i + 1) = \phi'(u)(i + 1) = 1$. It inherits the adaptive submodularity of the adaptive IM problem under the full-adoption feedback model [8], thereby we have $\Delta(u|\boldsymbol{x}, \phi \sim \psi) \ge \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$; or else there is no marginal gain if $\phi(u)(i + 1) = \phi'(u)(i + 1) = 0$. When $\boldsymbol{x}(u) = i < j = \boldsymbol{y}(u) =$, we have $\Delta(u|\boldsymbol{x}, \phi \sim \psi) \ge \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$ apparently as well if $\phi(u)(i + 1) = \phi'(j + 1) = 1$; or else there is no marginal gain if $\phi(u)(i + 1) = \phi'(j + 1) = 0$.

From the above, we have known $\Delta(u|\boldsymbol{x}, \phi \sim \psi) \ge \Delta(u|\boldsymbol{y}, \phi' \sim \psi')$. According to Equation (10) and Definition 1, we have as follows:

$$\Delta(u|\boldsymbol{x}, \psi) = \sum_{\phi \sim \psi} \Pr[\phi|\phi \sim \psi] \Delta(u|\boldsymbol{x}, \phi \sim \psi) \tag{16}$$

$$= \sum_{\phi' \sim \psi'} \Pr[\phi'|\phi' \sim \psi'] \sum_{\phi \sim \alpha} \Pr[\phi|\phi \sim \alpha] \Delta(u|\boldsymbol{x}, \phi \sim \psi). \tag{17}$$

Since $\sum_{\phi \sim \alpha} \Pr[\phi|\phi \sim \alpha] = 1$, we have

$$(17) \ge \sum_{\phi' \sim \psi'} \Pr[\phi'|\phi' \sim \psi'] \sum_{\phi \sim \alpha} \Pr[\phi|\phi \sim \alpha] \Delta(u|\boldsymbol{y}, \phi' \sim \psi') \tag{18}$$

$$\ge \sum_{\phi' \sim \psi'} \Pr[\phi'|\phi' \sim \psi'] \Delta(u|\boldsymbol{y}, \phi' \sim \psi') \sum_{\phi \sim \alpha} \Pr[\phi|\phi \sim \alpha] \tag{19}$$

$$= \sum_{\phi' \sim \psi'} \Pr[\phi'|\phi' \sim \psi'] \Delta(u|\boldsymbol{y}, \phi' \sim \psi') \tag{20}$$

$$= \Delta(u|\boldsymbol{y}, \psi'). \tag{21}$$

Therefore, we have $\Delta(u|\boldsymbol{x}, \psi) \ge \Delta(u|\boldsymbol{y}, \psi')$ for any $\boldsymbol{x} \le \boldsymbol{y}$ and their partial realizations $\psi \subseteq \psi'$. The proof of adaptive submodular is completed. □

5 ALGORITHM AND THEORETICAL ANALYSIS

In this section, we propose algorithms to solve our Adaptive-IMMA problem and give an approximation ratio with necessary theoretical analysis.

5.1 Adaptive Greedy Policy

We define a randomized adaptive greedy policy $\pi^g(\kappa)$ here. The seeding vector $\boldsymbol{x}$ is initialized to $\boldsymbol{x} = \boldsymbol{0} \in \mathbb{Z}_+^V$. In each iteration, the $\pi^g(\kappa)$ selects the node $u^* \in V$ that maximizes

ALGORITHM 1: AdaptiveGreedy $(G, f, k, \boldsymbol{b}, c)$

Input: A graph $G = (V, E)$, a function $f(\cdot, \phi)$, a budget $k \in \mathbb{R}_+$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$ and, a cost function $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$

Output: A seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$ and $f(\boldsymbol{x}, \psi)$

1: Initialize: $\boldsymbol{x} := \mathbf{0}$
2: Initialize: $\psi := \left\{\{?\}^{\boldsymbol{b}(u)}\right\}_{u \in V} \cup \{?\}^E$
3: **while** $c(\boldsymbol{x}) < k$ **do**
4: $u^* \in \arg\max_{u \in V, \boldsymbol{x}(u) < \boldsymbol{b}(u)} \Delta(u | \boldsymbol{x}, \psi) / c(\langle u, \boldsymbol{x}(u) + 1 \rangle)$
5: **if** $c(\boldsymbol{x}) + c(\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle) > k$ **then**
6: **break** with probability $1 - (k - c(\boldsymbol{x}))/c(\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle)$
7: **end if**
8: $\boldsymbol{x}(u^*) := \boldsymbol{x}(u^*) + 1$
9: Observe the state of $\langle u^*, \boldsymbol{x}(u^*) \rangle$
10: Update $\psi := \psi \cup \langle u^*, \boldsymbol{x}(u^*) \rangle$
11: **if** $\psi(u^*)(\boldsymbol{x}(u^*)) = 1$ **then**
12: Update the edge states of ψ observed by u^*'s actual influence diffusion
13: **end if**
14: **end while**
15: **return** $\boldsymbol{x}, f(\boldsymbol{x}, \psi)$

$\Delta(u | \boldsymbol{x}, \psi) / c(\langle u, \boldsymbol{x}(u) + 1 \rangle)$ where $\boldsymbol{x}(u) < \boldsymbol{b}(u)$ and ψ is the partial realization generated by the current $\boldsymbol{x}$, then increases $\boldsymbol{x}(u^*)$ by 1. Then, we need to observe the state of u^* and update this partial realization ψ. The $\pi^g(\kappa)$ repeats above procedure, terminates until $c(\boldsymbol{x}) \geq k$, or terminates with a probability. The main idea of adaptive greedy policy is shown in Algorithm 1. Shown as lines 5–7 of Algorithm 1, it returns with a probability when the remaining budget is not sufficient to do a trial on the selected node u^*. Thereby, the random source κ in this adaptive greedy policy indicates whether contains the selected node in the last iteration. The adaptive greedy policy $\pi^g(\kappa)$ shown as Algorithm 1 satisfies $\eta(\pi(\kappa), \phi) \leq \boldsymbol{b}$ and $\mathbb{E}_\kappa[c(\eta(\pi(\kappa), \phi))] \leq k$ for any realization ϕ.

5.2 Theoretical Analysis

To make the following analysis understandable, we introduce the operations of policy truncation and policy concatenation, which are adapted from [8] but suitable on integer lattice domain. We imagine a randomized policy $\pi(\kappa)$ running over time. In each iteration, it selects node $u^* = \pi(\kappa, \boldsymbol{x}, \psi)$ under the current seeding vector $\boldsymbol{x}$ and its partial realization ψ. It runs trial $\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle$ for $c(\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle)$ units of time and increases $\boldsymbol{x}(u^*)$ by 1.

Definition 4 (Policy Truncation). Let the seeding vector $\boldsymbol{x}$ kept by $\pi(\kappa)$, the policy truncation $\pi_{[t]}(\kappa)$ denotes the randomized policy that runs $\pi(\kappa)$ for t units of time. If the last trial $\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle$ can only be run for $0 \leq \tau < c(\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle)$ time, it will increase $\boldsymbol{x}(u^*)$ by 1 with probability $\tau \backslash c(\langle u^*, \boldsymbol{x}(u^*) + 1 \rangle)$. Under any realization ϕ, we have $\mathbb{E}_\kappa[c(\eta(\pi_{[t]}(\kappa), \phi))] \leq t$.

Definition 5 (Policy Concatenation). For any two adaptive policies $\pi(\kappa)$ and $\pi'(\kappa)$, the policy concatenation $\pi(\kappa) @ \pi'(\kappa)$ denotes the adaptive policy that runs policy $\pi(\kappa)$ first, and then runs $\pi'(\kappa)$ like a fresh start without information from the run of $\pi(\kappa)$. Under any realization ϕ, we have $\eta(\pi(\kappa) @ \pi'(\kappa), \phi) = \eta(\pi(\kappa), \phi) \vee \eta(\pi'(\kappa), \phi)$.

Lemma 1. *The objective function $f(\cdot, \phi)$ is adaptive monotone if and only if for any randomized policies $\pi(\kappa)$ and $\pi'(\kappa)$, we have*

$$\mathbb{E}_\kappa \left[f_{avg}(\pi(\kappa)) \right] \leq \mathbb{E}_\kappa \left[f_{avg}(\pi'(\kappa) @ \pi(\kappa)) \right]. \tag{22}$$

Proof. Given a fixed random source κ, we have $\eta(\pi'(\kappa)@\pi(\kappa),\phi) = \eta(\pi'(\kappa),\phi) \vee \eta(\pi(\kappa),\phi) = \eta(\pi(\kappa)@\pi'(\kappa),\phi)$ under any realization ϕ. Therefore, $f_{avg}(\pi(\kappa)) \leq f_{avg}(\pi'(\kappa)@\pi(\kappa))$ holds if and only if $f_{avg}(\pi(\kappa)) \leq f_{avg}(\pi(\kappa)@\pi'(\kappa))$. Then, we need to show $f_{avg}(\pi(\kappa)) \leq f_{avg}(\pi(\kappa)@\pi'(\kappa))$, which can be inferred from Lemma A.8 in [8], thus we omit here. Take the expectation over random source κ, Inequality (22) can be established. □

Lemma 2. *Given a seeding vector $\boldsymbol{x}$ and a partial realization ψ generated by it, $f(\cdot,\phi)$ is an adaptive monotone and adaptive dr-submodular function. For any policy $\pi^*(\kappa)$ that satisfies $\eta(\pi^*(\kappa),\phi) \leq \boldsymbol{b}$ and $\mathbb{E}_\kappa[c(\eta(\pi^*(\kappa),\phi))] \leq k$ for any realization ϕ, we have*

$$\Delta(\pi^*(\kappa)|\boldsymbol{x},\psi) \leq \mathbb{E}_{\phi\sim\psi}\left[c(\eta(\pi^*(\kappa),\phi))\right] \cdot \max_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \left\{\frac{\Delta(u|\boldsymbol{x},\psi)}{c(\langle u,\boldsymbol{x}(u)+1\rangle)}\right\}. \tag{23}$$

Proof. Consider the seeding vector $\boldsymbol{x}'$ maintained by a policy $\pi'(\kappa)$, we can define this policy $\pi'(\kappa)$ as follow. The seeding vector $\boldsymbol{x}'$ is initialized to $\boldsymbol{x}' = \boldsymbol{0} \in \mathbb{Z}_+^V$. The policy $\pi'(\kappa)$ increases $\boldsymbol{x}'(u)$ from 0 to $\boldsymbol{x}(u)$ step by step for each node $u \in V$. It will terminate if the state of trial $\langle u,i\rangle$ with $i \leq \boldsymbol{x}(u)$ is different from $\psi(u)(i)$ or the state of edge (u,v) is different from $\psi((u,v))$. If reaching $\boldsymbol{x}' = \boldsymbol{x}$ and not stopping, it will begin to run policy $\pi^*(\kappa)$ like a fresh start without information from before. Here, we can imagine there is a virtual vector $\boldsymbol{x}^*$ associated with $\pi^*(\kappa)$ updated from $\boldsymbol{0}$ and $\boldsymbol{x}' = \boldsymbol{x} \vee \boldsymbol{x}^*$ under the realization $\phi \sim \psi$.

For each trial $\langle u,i\rangle$, we define $w(\langle u,i\rangle) = \Pr[i \leq \eta(\pi'(\kappa),\phi)(u)|\phi\sim\psi]$ as the probability that u is selected by $\pi'(\kappa)$ and increases $\boldsymbol{x}'(u)$ from $i-1$ to i. When the policy $\pi^*(\kappa)$ selects a node $u \in V$ with $\boldsymbol{x}(u) \leq \boldsymbol{x}^*(u) < \boldsymbol{b}(u)$, namely $\langle u,\boldsymbol{x}^*(u)+1\rangle \notin \mathrm{dom}(\psi)$, the partial realization ψ' generated by current $\boldsymbol{x}'$ satisfies $\psi \subseteq \psi'$, thereby we have $\Delta(u|\boldsymbol{x}',\psi') \leq \Delta(u|\boldsymbol{x},\psi)$ because of adaptive dr-submodularity. Thus, the total contribution to $\Delta(\pi^*(\kappa)|\boldsymbol{x},\psi)$ is bounded by $\Delta(\pi^*(\kappa)|\boldsymbol{x},\psi) \leq \sum_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \sum_{i=\boldsymbol{x}(u)}^{\boldsymbol{b}(u)-1} w(\langle u,i+1\rangle)\cdot\Delta(u|\boldsymbol{x},\psi)$. From the above, we have

$$\Delta(\pi^*(\kappa)|\boldsymbol{x},\psi) \leq \sum_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \sum_{i=\boldsymbol{x}(u)}^{\boldsymbol{b}(u)-1} w(\langle u,i+1\rangle)\cdot\Delta(u|\boldsymbol{x},\psi) \tag{24}$$

$$= \sum_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \sum_{i=\boldsymbol{x}(u)}^{\boldsymbol{b}(u)-1} w(\langle u,i+1\rangle)\cdot c(\langle u,i+1\rangle)\cdot\frac{\Delta(u|\boldsymbol{x},\psi)}{c(\langle u,i+1\rangle)}. \tag{25}$$

Since $c(\langle u,i\rangle) \leq c(\langle u,i+1\rangle)$, we have

$$(25) \leq \sum_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \frac{\Delta(u|\boldsymbol{x},\psi)}{c(\langle u,\boldsymbol{x}(u)+1\rangle)} \sum_{i=\boldsymbol{x}(u)}^{\boldsymbol{b}(u)-1} w(\langle u,i+1\rangle)\cdot c(\langle u,i+1\rangle) \tag{26}$$

$$\leq \left(\sum_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \sum_{i=\boldsymbol{x}(u)}^{\boldsymbol{b}(u)-1} w(\langle u,i+1\rangle)\cdot c(\langle u,i+1\rangle)\right) \cdot \max_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \left\{\frac{\Delta(u|\boldsymbol{x},\psi)}{c(\langle u,\boldsymbol{x}(u)+1\rangle)}\right\} \tag{27}$$

$$\leq \mathbb{E}_{\phi\sim\psi}[c(\eta(\pi^*(\kappa),\phi))] \cdot \max_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)} \left\{\frac{\Delta(u|\boldsymbol{x},\psi)}{c(\langle u,\boldsymbol{x}(u)+1\rangle)}\right\}, \tag{28}$$

where Inequality (28) is correct because it only count a subset of trials contained in $\eta(\pi^*(\kappa),\phi)$. Thus, this lemma is proven. □

Theorem 3. *The adaptive greedy policy $\pi^g(\kappa)$ shown as Algorithm 1 achieves a $(1-e^{-1})$ expected approximation guarantee. Thus, for any policy $\pi^*(\kappa)$ that satisfies $\eta(\pi^*(\kappa),\phi) \leq \boldsymbol{b}$ and*

$\mathbb{E}_\kappa[c(\eta(\pi^*(\kappa),\phi))] \le k$ *for any realization* ϕ*, we have*

$$\mathbb{E}_\kappa\left[f_{avg}(\pi^g(\kappa))\right] \ge \left(1-e^{-1}\right)\cdot \mathbb{E}_\kappa\left[f_{avg}(\pi^*(\kappa))\right]. \tag{29}$$

Proof. Consider the policy $\pi^g_{[i+1]}(\kappa)$ given any $i \in [0, k-1]$, its current seeding vector and partial realization when it enters the last iteration before termination (line 3 of Algorithm 1) are denoted by $\boldsymbol{x}$ and ψ. In the last iteration, the node u^* is selected in line 4 of Algorithm 1. The expected marginal gain of the last iteration is $\Delta(u^*|\boldsymbol{x},\psi)\cdot(i+1-c(\boldsymbol{x}))/c(\langle u^*, \boldsymbol{x}(u^*)+1\rangle)$. Consider the policy $\pi^g_{[i]}(\kappa)$, there are two cases could happen.

(1) If $i \ge c(\boldsymbol{x})$, its execution will be the same as the policy $\pi^g_{[i+1]}(\kappa)$ until entering the last iteration. It updates $\boldsymbol{x}$ to $\boldsymbol{x}+\boldsymbol{e}_{u^*}$ with probability $(i-c(\boldsymbol{x}))/c(\langle u^*, \boldsymbol{x}(u^*)+1\rangle)$, which has $\Delta(u^*|\boldsymbol{x},\psi)\cdot(i-c(\boldsymbol{x}))/c(\langle u^*, \boldsymbol{x}(u^*)+1\rangle)$ expected marginal gain in the last iteration.
(2) If $i \le c(\boldsymbol{x})$, the $\pi^g_{[i]}(\kappa)$ will not enter the last iteration of the policy $\pi^g_{[i+1]}(\kappa)$ obviously.

According to the above analysis, the gap of the expected value of objective function returned by $\pi^g_{[i+1]}(\kappa)$ and $\pi^g_{[i]}(\kappa)$ can be bounded. We have

$$\mathbb{E}_\kappa\left[\mathbb{E}_{\phi\sim\psi}\left[f(\eta(\pi^g_{[i+1]}(\kappa),\phi),\phi)\right]\right] - \mathbb{E}_\kappa\left[\mathbb{E}_{\phi\sim\psi}\left[f(\eta(\pi^g_{[i]}(\kappa),\phi),\phi)\right]\right] \ge \frac{\Delta(u^*|\boldsymbol{x},\psi)}{c(\langle u^*, \boldsymbol{x}(u^*)+1\rangle)}. \tag{30}$$

Here, the $\boldsymbol{x}$ and ψ are fixed, which are determined by potential realization ϕ. Take the expectation over all realizations, we have

$$\mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i+1]}(\kappa))\right] - \mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa))\right] \ge \mathbb{E}_{\phi\leftarrow\Phi}\left[\frac{\Delta(u^*|\boldsymbol{x}_\phi,\psi_\phi)}{c(\langle u^*, \boldsymbol{x}_\phi(u^*)+1\rangle)}\right], \tag{31}$$

where the $\boldsymbol{x}_\phi$ (ψ_ϕ) is the current seeding vector (partial realization) of the policy $\pi^g_{[i+1]}(\kappa)$ at the beginning of its last iteration under the potential realization ϕ and the node u^* can be considered as the one that is able to get the maximum marginal gain based on the seed vector $\boldsymbol{x}_\phi$ and its partial realization ψ_ϕ.

Then, the definition of $\boldsymbol{x}$ and ψ are the same as above. We can define the seeding vector $\boldsymbol{y}$ and its partial realization ψ' as that returned by the policy $\pi^g_{[i]}(\kappa)$, where we have $\psi \subseteq \psi'$ and $\boldsymbol{x} \le \boldsymbol{y}$. Policy $\pi^g_{[i]}(\kappa)@\pi^*(\kappa)$ increase the value of objective function of policy $\pi^g_{[i]}(\kappa)$ by $\mathbb{E}_\kappa[\Delta(\pi^*(\kappa)|\boldsymbol{y},\psi')]$ expectedly. Besides, we have $\mathbb{E}_\kappa[\Delta(\pi^*(\kappa)|\boldsymbol{y},\psi')] \le \mathbb{E}_\kappa[\Delta(\pi^*(\kappa)|\boldsymbol{x},\psi)]$ due to the adaptive dr-submodularity of $f(\cdot,\phi)$. Here, the $\boldsymbol{x}$ and ψ are fixed, which are determined by potential realization ϕ. Take the expectation over all realizations, we have

$$\mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa)@\pi^*(\kappa))\right] - \mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa))\right] \le \mathbb{E}_\kappa\left[\mathbb{E}_{\phi\leftarrow\Phi}\left[\Delta(\pi^*(\kappa)|\boldsymbol{x}_\phi,\psi_\phi)\right]\right]. \tag{32}$$

According to Inequality (23) in Lemma 2, we have

$$\mathbb{E}_\kappa\left[\Delta(\pi^*(\kappa)|\boldsymbol{x}_\phi,\psi_\phi)\right] \le \mathbb{E}_\kappa\left[\mathbb{E}_{\phi\sim\psi_\phi}\left[c(\eta(\pi^*(\kappa),\phi))\right]\right]\cdot \max_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)}\left\{\frac{\Delta(u|\boldsymbol{x}_\phi,\psi_\phi)}{c(\langle u,\boldsymbol{x}_\phi(u)+1\rangle)}\right\} \tag{33}$$

$$\le k\cdot \max_{u\in V,\boldsymbol{x}(u)<\boldsymbol{b}(u)}\left\{\frac{\Delta(u|\boldsymbol{x}_\phi,\psi_\phi)}{c(\langle u,\boldsymbol{x}_\phi(u)+1\rangle)}\right\} = k\cdot\frac{\Delta(u^*|\boldsymbol{x}_\phi,\psi_\phi)}{c(\langle u^*,\boldsymbol{x}_\phi(u^*)+1\rangle)}, \tag{34}$$

where Inequality (34) is from $\mathbb{E}_\kappa[\mathbb{E}_{\phi\sim\psi_\phi}[c(\eta(\pi^*(\kappa),\phi))]] = \mathbb{E}_{\phi\sim\psi_\phi}[\mathbb{E}_\kappa[c(\eta(\pi^*(\kappa),\phi))]] \le k$ since $\mathbb{E}_\kappa[c(\eta(\pi^*(\kappa),\phi))] \le k$ for any realization ϕ. Thus, we have

$$(32) = \mathbb{E}_{\phi\leftarrow\Phi}\left[\mathbb{E}_\kappa\left[\Delta(\pi^*(\kappa)|\boldsymbol{x}_\phi,\psi_\phi)\right]\right] \tag{35}$$

$$\le k\cdot\mathbb{E}_{\phi\leftarrow\Phi}\left[\frac{\Delta(u^*|\boldsymbol{x}_\phi,\psi_\phi)}{c(\langle u^*,\boldsymbol{x}_\phi(u^*)+1\rangle)}\right] \le k\cdot\left(\mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i+1]}(\kappa))\right] - \mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa))\right]\right). \tag{36}$$

Based on Lemma 1, we have $\mathbb{E}_\kappa[f_{avg}(\pi^*(\kappa))] \leq \mathbb{E}_\kappa[f_{avg}(\pi^g_{[i]}(\kappa)@\pi^*(\kappa))]$ because of its adaptive monotonicity. According to Inequality (31), (32), and (36), we have

$$\mathbb{E}_\kappa\left[f_{avg}(\pi^*(\kappa))\right] - \mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa))\right] \leq k \cdot \left(\mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i+1]}(\kappa))\right] - \mathbb{E}_\kappa\left[f_{avg}(\pi^g_{[i]}(\kappa))\right]\right). \quad (37)$$

Now, we can define $\theta_i := \mathbb{E}_\kappa[f_{avg}(\pi^*(\kappa))] - \mathbb{E}_\kappa[f_{avg}(\pi^g_{[i]}(\kappa))]$, which means $\theta_i \leq k \cdot (\theta_i - \theta_{i+1})$ and $\theta_{i+1} \leq (1 - 1/k) \cdot \theta_i$. Here, we have $\theta_k \leq (1 - 1/k)^k \cdot \theta_0 \leq (1/e) \cdot \theta_0$, therefore $\mathbb{E}_\kappa[f_{avg}(\pi^*(\kappa))] - \mathbb{E}_\kappa[f_{avg}(\pi^g(\kappa))] \leq (e^{-1}) \cdot (\mathbb{E}_\kappa[f_{avg}(\pi^*(\kappa))] - \mathbb{E}_\kappa[f_{avg}(\pi^g_{[0]}(\kappa))])$ when k is relatively large. That is $\mathbb{E}_\kappa[f_{avg}(\pi^g(\kappa))] \geq (1 - e^{-1}) \cdot \mathbb{E}_\kappa[f_{avg}(\pi^*(\kappa))]$. The proof of this theorem is completed. □

6 SOLUTION FRAMEWORK BY SAMPLING

In the last section, our adaptive greedy policy shown as Algorithm 1 can achieve a $(1-1/e)$ expected approximation guarantee, which has been proved by Theorem 3. However, it is based on a basic assumption that we are able to compute the exact value of $\Delta(u|\boldsymbol{x}, \psi)$ and get the feasible node with maximum unit marginal gain in line 4 of Algorithm 1 in each iteration. In fact, this is an impossible task because it is #P-hard [4] to compute marginal gain $\Delta(u|\boldsymbol{x}, \psi)$ for each node $u \in V$ under the IC-model. Thus, the true value of $\Delta(u|\boldsymbol{x}, \psi)$ is difficult to obtain. MC simulations is a general method to estimate this value, but its running time is unacceptable. To overcome that, we are able to seek an estimator of $\Delta(u|\boldsymbol{x}, \psi)$ through the RIS [1] then maximize this estimator. If maximizing this estimator through sampling technique, it will be possible to get a extremely worse node with some probability, even though very small. In other words, the selected node u^* in line 4 of Algorithm 1 is not optimal such that $u^* \notin \arg\max_{u \in V, \boldsymbol{x}(u) < \boldsymbol{b}(u)} \Delta(u|\boldsymbol{x}, \psi)/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in actual execution. Like this, the expected approximation ratio shown in Theorem 3 will not be ensured.

6.1 Sampling Technique

Consider the traditional IM problem, we need to introduce the concept of reverse reachable sets (RR-sets) first. Given a graph $G = (V, E)$, a random RR-set of G can be generated by selecting a node $u \in V$ uniformly and sampling a graph realization g from $\mathcal{G}$, then collecting those nodes can reach u in g. A RR-set rooted at 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 $\sigma_G(S) = n \cdot \Pr[R \cap S \neq \emptyset]$. Let $\mathcal{R} = \{R_1, R_2, \ldots, R_\theta\}$ be a collection of random RR-sets and $z(S, R)$ be the indicator, where $z(S, R) = 1$ if $S \cap R \neq \emptyset$, or else $z(S, R) = 0$. Denoted by $F_\mathcal{R}(S) = \sum_{i=1}^{\theta} z(S, R_i)/\theta$, the $n \cdot F_\mathcal{R}(S)$ is an unbiased estimator of the expected influence spread $\sigma_G(S)$. When the $|\mathcal{R}|$ is large, the $n \cdot F_\mathcal{R}(S)$ will converge to the true value $\sigma_G(S)$. Thus, how to set the value of θ is flexible, we need to balance between accuracy and running time carefully.

For our adaptive greedy policy, its current seeding vector and partial realization at the beginning of each iteration (when entering line 3 of Algorithm 1) are denoted by $\boldsymbol{x}$ and ψ. Let $G(\psi) = (V(\psi), E(\psi))$ be the subgraph induced by all inactive nodes under the current partial realization ψ. Here, computing $\Delta(u|\boldsymbol{x}, \psi)$ is equivalent to computing $\beta_u \cdot \sigma_{G(\psi)}(\{u\})$. We can note that $\Delta(u|\boldsymbol{x}, \psi) = 0$ if node $u \notin V(\psi)$. Thus, for a node $u \in V(\psi)$ and a random RR-sets $R(\psi)$ of $G(\psi)$, we can get an unbiased estimator of $\Delta(u|\boldsymbol{x}, \psi)$. That is

$$\Delta(u|\boldsymbol{x}, \psi) = \beta_u \cdot \sigma_{G(\psi)}(\{u\}) \quad (38)$$

$$= \beta_u \cdot |V(\psi)| \cdot \Pr[\{u\} \cap R(\psi) \neq \emptyset]. \quad (39)$$

Then, we can reformulate our adaptive greedy policy through the above sampling, which is shown in Algorithm 2. It is called "sampled adaptive greedy policy" and denoted by $\pi^{gs}(\kappa, \omega)$,

ALGORITHM 2: Sampled-AdaptiveGreedy $(G, f, k, \boldsymbol{b}, c, \varepsilon)$

Input: A graph $G = (V, E)$, a function $f(\cdot, \phi)$, a budget $k \in \mathbb{R}_+$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$, a cost function $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$, and an error parameter ε
Output: A seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$ and $f(\boldsymbol{x}, \psi)$

1: Initialize: $\boldsymbol{x} := \mathbf{0}$
2: Initialize: $\psi := \left\{\{?\}^{\boldsymbol{b}(u)}\right\}_{u \in V} \cup \{?\}^E$
3: Initialize: $G(\psi) := G$
4: Initialize: r be defined as Equation (41)
5: **while** $c(\boldsymbol{x}) < k$ **do**
6: $\quad u^\circ \leftarrow$ Generalized-EPIC $(G(\psi), \boldsymbol{x}, \boldsymbol{b}, c, \varepsilon)$
7: $\quad$ **if** $c(\boldsymbol{x}) + c(\langle u^\circ, \boldsymbol{x}(u^\circ) + 1\rangle) > k$ **then**
8: $\quad\quad$ **break** with probability $1 - (k - c(\boldsymbol{x}))/c(\langle u^\circ, \boldsymbol{x}(u^\circ) + 1\rangle)$
9: $\quad$ **end if**
10: $\quad \boldsymbol{x}(u^\circ) := \boldsymbol{x}(u^\circ) + 1$
11: $\quad$ Observe the state of $\langle u^\circ, \boldsymbol{x}(u^\circ)\rangle$
12: $\quad$ Update $\psi := \psi \cup \langle u^\circ, \boldsymbol{x}(u^\circ)\rangle$
13: $\quad$ **if** $\psi(u^\circ)(\boldsymbol{x}(u^\circ)) = 1$ **then**
14: $\quad\quad$ Update the edge states of ψ observed by u°'s actual influence diffusion
15: $\quad\quad$ Update $G(\psi)$ by removing all active nodes
16: $\quad$ **end if**
17: **end while**
18: **return** $\boldsymbol{x}, f(\boldsymbol{x}, \psi)$

where the random variable ω indicates the random source of sampling for estimations. In each iteration, it generates a collection of random RR-sets $\mathcal{R}(\psi) = \{R_1(\psi), R_2(\psi), \ldots, R_\theta(\psi)\}$ based on current subgraph $G(\psi)$ first. Then, it select a feasible node $u^\circ \in V(\psi)$ that maximizes $\beta_u \cdot |V(\psi)| \cdot F_{\mathcal{R}(\psi)}(\{u\})/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ where $\boldsymbol{x}(u) < \boldsymbol{b}(u)$ and increases $\boldsymbol{x}(u^\circ)$ by 1. Finally, we need to observe the state of u°, update this partial realization ψ, and update the subgraph $G(\psi)$. The $\pi^{gs}(\kappa, \omega)$ repeats above procedure, terminates until $c(\boldsymbol{x}) \geq k$, or terminates with a probability.

6.2 Theoretical Analysis and Time Complexity

According to the current seeding vector $\boldsymbol{x}$ and its partial realization ψ at the beginning of each iteration (line 5 of Algorithm 2), we can get a subgraph $G(\psi)$ and a collection of random RR-sets $\mathcal{R}(\psi)$. Now, we define a function $H_{\mathcal{R}(\psi)}(\{u\}|\boldsymbol{x}) = \beta_u \cdot F_{\mathcal{R}(\psi)}(\{u\})/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$, thereby the $|V(\psi)| \cdot H_{\mathcal{R}(\psi)}(\{u\}|\boldsymbol{x})$ is an unbiased estimator of $\Delta(u|\boldsymbol{x}, \psi)/c(\langle u, \boldsymbol{x}(u)+1\rangle)$. Next, a natural question is how to determine the number of RR-sets in $\mathcal{R}(\psi)$. The procedure of generating enough random RR-sets of $G(\psi)$ and returning the approximately optimal node $u^\circ \in V(\psi)$ (line 7 of Algorithm 2) in each iteration is shown in Algorithm 3. It is adapted from the sampling process of EPIC in [16], but there are several differences: (1) The seed size is fixed to one; and (2) The targeted estimator is the function $H_{\mathcal{R}(\psi)}(\{u\}|\boldsymbol{x})$ we defined before instead of $F_{\mathcal{R}(\psi)}(\{u\})$. Thus, the sampling process shown as algorithm 3 is called "Generalized-EPIC."

From line 1 to line 5 of Algorithm 3, it initializes those parameters similar to EPIC in [16] but fixes the seed set to one, then generate two collections $\mathcal{R}_1(\psi)$ and $\mathcal{R}_2(\psi)$ of random RR-sets with the same size. In each iteration, it select the feasible node $u^\circ \in V(\psi)$ that maximizes the estimator $H_{\mathcal{R}_1(\psi)}(\{u\}|\boldsymbol{x})$, which can be computed in polynomial time. Denoted by u^* the optimal feasible node that maximizes the unit marginal gain $\Delta(u|\boldsymbol{x}, \psi)/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$, the $H^u(\{u^*\})$ is an upper bound on $H_{\mathcal{R}_1(\psi)}(\{u^*\}|\boldsymbol{x})$. Thus, we have $H^u(\{u^*\}) = H_{\mathcal{R}_1(\psi)}(\{u^\circ\}|\boldsymbol{x}) \geq H_{\mathcal{R}_1(\psi)}(\{u^*\}|\boldsymbol{x})$. Moveover,

ALGORITHM 3: Generalized-EPIC $(G(\psi), \boldsymbol{x}, \boldsymbol{b}, c, \varepsilon)$ [16]

Input: A graph $G(\psi) = (V(\psi), E(\psi))$, the current seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$, a cost $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$, and an error parameter ε

Output: An approximately optimal node $u^\circ \in V(\psi)$

1: Initialize: $\delta := 0.01 \cdot \varepsilon / |V(\psi)|$
2: Initialize: $\bar{\varepsilon} := (\varepsilon - \delta \cdot |V(\psi)|)/(1 - \delta \cdot |V(\psi)|)$
3: Initialize: $\hat{\varepsilon} := \bar{\varepsilon}/(1 - \bar{\varepsilon})$
4: Initialize: $i_{max} := \left\lceil \log_2 \frac{(2+2\cdot\hat{\varepsilon}/3)\cdot|V(\psi)|}{\hat{\varepsilon}^2} \right\rceil + 1$ and $a = \ln\left(\frac{2\cdot i_{max}}{\delta}\right)$
5: Initialize: $\theta := \ln\left(\frac{2}{\delta}\right) + \ln\left(\binom{|V(\psi)|}{1}\right)$
6: Generate two collections $\mathcal{R}_1(\psi)$ and $\mathcal{R}_2(\psi)$ of random RR-sets with $|\mathcal{R}_1(\psi)| = |\mathcal{R}_2(\psi)| = \theta$
7: **for** $i = 1$ to i_{max} **do**
8: $\quad u^\circ \in \arg\max_{u \in V(\psi), \boldsymbol{x}(u) < \boldsymbol{b}(u)} H_{\mathcal{R}_1(\psi)}(\{u\}|\boldsymbol{x})$
9: $\quad H^u(\{u^*\}) \leftarrow H_{\mathcal{R}_1(\psi)}(\{u^\circ\}|\boldsymbol{x})$
10: $\quad H^l(\{u^\circ\}) \leftarrow \left(\sqrt{H_{\mathcal{R}_2(\psi)} + \frac{2\cdot a}{9\cdot|\mathcal{R}_2(\psi)|}} - \sqrt{\frac{a}{2\cdot|\mathcal{R}_2(\psi)|}}\right)^2 - \frac{a}{18\cdot|\mathcal{R}_2(\psi)|}$
11: $\quad$ **if** $\frac{H^l(\{u^\circ\})}{H^u(\{u^*\})} \geq 1 - \bar{\varepsilon}$ or $i = i_{max}$ **then**
12: $\quad\quad$ **return** u°
13: $\quad$ **end if**
14: $\quad$ Double the size of $\mathcal{R}_1(\psi)$ and $\mathcal{R}_2(\psi)$ with new random RR-sets
15: **end for**

the $|V(\psi)| \cdot H^l(\{u^\circ\})$ gives an accurate lower bound on $\Delta(u^\circ|\boldsymbol{x}, \psi)/c(\langle u^\circ, \boldsymbol{x}(u^\circ) + 1\rangle)$ with high probability. After that, it checks whether the stopping condition in line 11 can be satisfied. If true, it will return an approximate optimal node u° definitely.

Lemma 3. *Given the current seeding vector $\boldsymbol{x}$ and its partial realization ψ, the feasible node u° returned by Algorithm 3 achieves a $(1-\varepsilon)$ expected approximation guarantee within $O((|V(\psi)| + |E(\psi)|) \cdot (\log(|V(\psi)|) + \log(1/\varepsilon))/\varepsilon^2)$ expected time. That is*

$$\mathbb{E}_\omega\left[\frac{\Delta(u^\circ|\boldsymbol{x}, \psi)}{c(\langle u^\circ, \boldsymbol{x}(u^\circ) + 1\rangle)}\right] \geq (1-\varepsilon) \cdot \max_{u \in V(\psi), \boldsymbol{x}(u) < \boldsymbol{b}(u)} \left\{\frac{\Delta(u|\boldsymbol{x}, \psi)}{c(\langle u, \boldsymbol{x}(u) + 1\rangle)}\right\} \tag{40}$$

Proof. Given the current seeding vector $\boldsymbol{x}$ and its partial realization ψ, let us look at the targeted function $H_{\mathcal{R}(\psi)}(\{u\}|\boldsymbol{x}) = \beta_u \cdot F_{\mathcal{R}(\psi)}(\{u\})/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$. It is a weighted coverage on the collection $\mathcal{R}(\psi)$, where we can consider the $\beta_u/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ as the weight of each node $u \in V(\psi)$. The weighted coverage function is submodular, thereby we can compute the node $u^\circ \in \arg\max_{u \in V(\psi), \boldsymbol{x}(u) < \boldsymbol{b}(u)} H_{\mathcal{R}_1(\psi)}(\{u\}|\boldsymbol{x})$ accurately shown as line 8 of Algorithm 3 in polynomial time. Because of its submodularity, Lemma 3 can be obtained by adapting from the expected approximation guarantee of EPIC in [16]. □

Let us look back at Algorithm 2. The actual number of activated seeds should be much less than the number of actual iterations in Algorithm 2, since there are some iterations that fail to activate its selected node. Based on that, we can make the following assumptions:

(1) Generate an active seed successfully in each iteration, namely we suppose $\beta_u = 1$ for each node $u \in V$.
(2) The node we select in each iteration has the lowest cost until now.
(3) We sort the node set V as $\{v_1', v_2', \ldots, v_n'\}$ with $c(\langle v_1', 1\rangle) \leq c(\langle v_2', 1\rangle) \leq \cdots \leq c(\langle v_n', 1\rangle)$.

Table 1. The Statistics of four Datasets in our Simulations ($K = 10^3$)

Dataset	n	m	Type	Avg. Degree
NetScience	0.4K	1.01K	undirected	5.00
Wiki	1.0K	3.15K	directed	6.20
HetHEPT	12.0K	118.5K	undirected	19.8
Epinions	75.9K	508.8K	directed	13.4

Given a graph $G = (V, E)$ and a budget k, we can define the maximum number of iterations in Algorithm 2 as r. That is

$$r = \begin{cases} n & \text{if } \sum_{i=1}^{n} c(\langle v_i', 1\rangle) \le k \\ q & \text{else } q = \min\{q | \sum_{i=1}^{q} c(\langle v_i', 1\rangle) \ge k\} \end{cases} \tag{41}$$

By finding the smallest r such that $\sum_{i=1}^{r} c(\langle v_i', 1\rangle) \ge k$, it is obvious that the actual number of iterations in Algorithm 2 must be less than r defined in Equation (41).

Theorem 4. *The sampled adaptive greedy policy $\pi^{gs}(\kappa, \omega)$ shown as Algorithm 2 achieved a $(1 - e^{-1+\varepsilon})$ expected approximation guarantee within $O(r \cdot (n + m) \cdot (\log(n) + \log(1/\varepsilon))/\varepsilon^2)$ expected time. Thus, for any policy $\pi^*(\kappa)$ that satisfies $\eta(\pi^*(\kappa), \phi) \le \boldsymbol{b}$ and $\mathbb{E}_\kappa[c(\eta(\pi^*(\kappa), \phi))] \le k$ for any realization ϕ, we have*

$$\mathbb{E}_\kappa \left[\mathbb{E}_\omega \left[f_{avg}(\pi^{gs}(\kappa, \omega))\right]\right] \ge \left(1 - e^{-1+\varepsilon}\right) \cdot \mathbb{E}_\kappa \left[f_{avg}(\pi^*(\kappa))\right] \tag{42}$$

Proof. According to the above assumptions, the maximum number of iterations can be executed in Algorithm 2 is r. Based on Lemma 3, the selected node in each iteration satisfies $(1 - \varepsilon)$ expected approximation. In this extreme case, the total expected error over all iterations is $\varepsilon = (1/r) \cdot \sum_{i=1}^{r} \varepsilon$. Actually, the total expected error will be much less than ε due to the $\beta_u \le 1$ for each node $u \in V$. Here, there is no node can be activated in many iterations. Based on Theorem 3 and Lemma 3, Theorem 4 holds by inferring from Theorem 6 in [16]. □

7 EXPERIMENT

In this section, we carry out several experiments on different datasets to validate the performance of our proposed policy. It aims to test the efficiency of our sampled adaptive greedy policy shown as Algorithm 2 and its effectiveness compared to other adaptive heuristic policies. All of our experiments are programmed by python and run on a Windows machine with a 3.40 GHz, 4 core Intel CPU and 16 GB RAM.

7.1 Dataset Description and Statistics

There are four datasets used in our experiments: (1) NetScience [21]: a co-authorship network, co-authorship among scientists to publish papers about network science; (2) Wiki [21]: a who-votes-on-whom network, which come from the collection Wikipedia voting; (3) HetHEPT [18]: an academic collaboration relationship on high energy physics area; and (4) Epinions [18]: a who-trust-whom OSN on Epinions.com, a general consumer review site. The statistics information of these four datasets is represented in Table 1. For the undirected graph, each undirected edge is replaced with two reversed directed edges.

7.2 Experimental Setting

The diffusion model used in our experiments relies on the IC-model. For each edge $(u, v) \in E$, we set $p_{uv} = 1/|N^-(v)|$, which is widely used by prior works about influence maximization [17],

ALGORITHM 4: Greedy $(G, \mu, k, \boldsymbol{b}, c)$

Input: A graph $G = (V, E)$, a function $\mu(\boldsymbol{x})$, a budget $k \in \mathbb{R}_+$, a vector $\boldsymbol{b} \in \mathbb{Z}_+^V$ and, a cost function $c : V \times \mathbb{Z}_+ \to \mathbb{R}_+$

Output: A seeding vector $\boldsymbol{x} \in \mathbb{Z}_+^V$ and $\mu(\boldsymbol{x})$

Initialize: $\boldsymbol{x} := \boldsymbol{0}$
while $c(\boldsymbol{x}) < k$ **do**
  $u^* \in \arg\max_{u \in V, \boldsymbol{x}(u) < \boldsymbol{b}(u)} (\mu_G(\boldsymbol{x} + \boldsymbol{e}_u) - \mu_G(\boldsymbol{x}))/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$
  if $c(\boldsymbol{x}) + c(\langle u^*, \boldsymbol{x}(u^*) + 1\rangle) > k$ **then**
    break with probability $1 - (k - c(\boldsymbol{x}))/c(\langle u^*, \boldsymbol{x}(u^*) + 1\rangle)$
  end if
  $\boldsymbol{x}(u^*) := \boldsymbol{x}(u^*) + 1$
end while
return $\boldsymbol{x}$, $\mu(\boldsymbol{x})$

[1], [30], [29], [19]. There are several parameters associated with the objective function of our Adaptive-IMMA problem. Here, we set the vector $\boldsymbol{b} = \{5\}^V$ where each node can be attempted to activate as a seed at most 5 times; the cost of each trial $c(\langle u, 1\rangle) = 1$ and $c(\langle u, i+1\rangle) = 1.2 \times c(\langle u, i\rangle)$; and variable budget $k \in \{0, 10, 20, 30, 40, 50\}$. Besides, for each node $u \in V$, its probability β_u is sampled from a normal distribution within given a mean, variance and interval. For each adaptive policies, we generate 20 realizations (test it 20 times) randomly and take the average of their results as its final performance.

We perform two experiments with different purposes in this section. The first experiment is to test the time efficiency of the adaptive greedy policy and sampled adaptive greedy policy (Algorithm 2), then validate the superiority over their non-adaptive settings. The corresponding non-adaptive versions of adaptive greedy policy and sampled adaptive greedy policy are referred to as greedy and sampled greedy algorithm, respectively. Here, the greedy algorithm and adaptive greedy policy are implemented by MC simulations. They can be shown as follows:

(1) Greedy algorithm: Shown as Algorithm 4, it selects a node $u \in V$ with $\boldsymbol{x}(u) < \boldsymbol{b}$ such that maximizes the unit marginal gain $(\mu_G(\boldsymbol{x} + \boldsymbol{e}_u) - \mu_G(\boldsymbol{x}))/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in each iteration. The selected node in the last iteration will be contained with a probability. To estimate the value of $\mu_G(\boldsymbol{x})$, we have

$$\mu_G(\boldsymbol{x}) = \sigma_{\widetilde{G}}(\widetilde{V} - V) - |V|, \tag{43}$$

where we need to create a constructed graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ by adding a new node $\widetilde{u}$ and a new directed edge $(\widetilde{u}, u)$ for each node $u \in V$ to G, where $(\widetilde{u}, u)$ is with activation probability $p_{\widetilde{u}u} = 1 - (1 - \beta_u)^{\boldsymbol{x}(u)}$. Here, the $\sigma_{\widetilde{G}}(\widetilde{V} - V)$ can be estimated by MC simulations, which is an effective methods to estimate the value of $\mu(\boldsymbol{x})$ [10].

(2) Adaptive greedy policy: Shown as Algorithm 1, we can compute the unit marginal gain $\Delta(u|\boldsymbol{x}, \psi)/c(\langle u, \boldsymbol{x}(u)+1\rangle)$ through $\beta_u \cdot \sigma_{G(\psi)}(\{u\})$ according to Equation (38), where $\sigma_{G(\psi)}(\{u\})$ can be estimated by MC simulations.

(3) Sampled greedy algorithm: Here, we require to obtain an unbiased estimator of $\mu(\boldsymbol{x})$. Let $\mathcal{R}$ be a collection of random RR-sets sampled from G, we have

$$\mu_G(\boldsymbol{x}) = |V| \cdot \mathbb{E}_R \left[1 - \prod_{u \in R} (1 - \beta_u)^{\boldsymbol{x}(u)} \right]. \tag{44}$$

Let $F_{\mathcal{R}}(\boldsymbol{x}) = (\theta - \sum_{i=1}^{\theta} \prod_{u \in R_i} (1 - \beta_u)^{\boldsymbol{x}(u)})/\theta$, thereby we have $|V| \cdot F_{\mathcal{R}}(\boldsymbol{x})$ is an unbiased estimator of $\mu(\boldsymbol{x})$. Because there is no existing algorithm to determine the number of random RR-sets in this case, we will guess a size of $\mathcal{R}$ according to datasets and budgets. Given a collection $\mathcal{R}$, it selects a node $u^\circ \in V$ with $\boldsymbol{x}(u) < \boldsymbol{b}(u)$ such that maximizes the unit marginal coverage $(F_{\mathcal{R}}(\boldsymbol{x} + \boldsymbol{e}_u) - F_{\mathcal{R}}(\boldsymbol{x}))/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in each iteration. The selected node in the last iteration will be contained with a probability, which is similar to Algorithm 4.

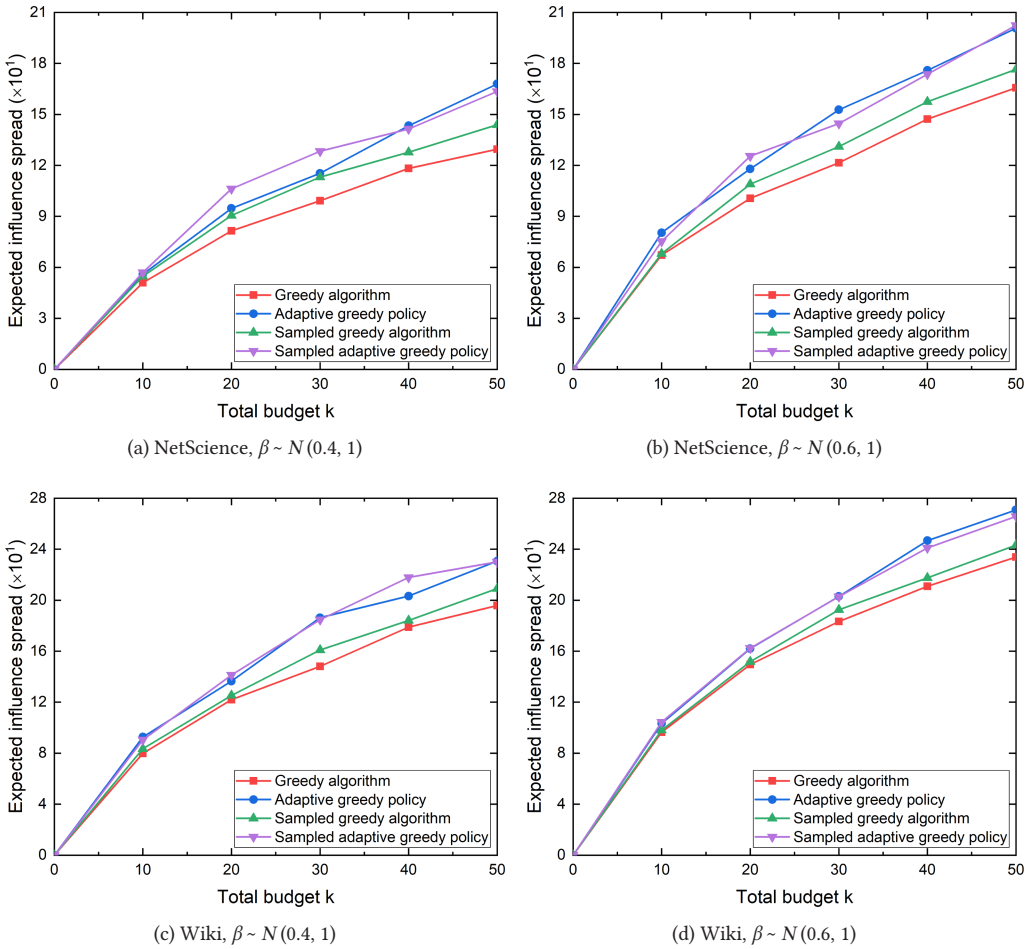
(4) Sampled adaptive greedy policy: It can be implemented by Algorithm 2 with the error parameter $\varepsilon = 0.5$.

The second experiment is to test the performance of our sampled adaptive greedy policy compared with other heuristic adaptive policies, which aims to evaluate its effectiveness. The difference between these heuristic adaptive policies and our sampled adaptive greedy policy lies in how to select a node u° from the feasible node set that satisfies $u \in V(\psi)$ and $\boldsymbol{x}(u) < \boldsymbol{b}(u)$ in each iteration. Thus, the only difference is in line 6 of Algorithm 2 and other procedures are totally identical. In other words, they are obtained by replacing line 6 of Algorithm 2 with these heuristic strategies, summarized as follows: (1) Random: select a node u° from the feasible node set uniformly in each iteration; (2) MaxDegree: select a node u° from the feasible node set that maximizes $N^+(u)/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in each iteration; (3) MaxProb: select a node u° from the feasible node set that maximizes $\beta_u/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in each iteration; and (4) MaxDegreeProb: select a node u° from the feasible node set that maximizes $\beta_u \cdot N^+(u)/c(\langle u, \boldsymbol{x}(u) + 1\rangle)$ in each iteration.

7.3 Experimental Results

Figures 1 and 2 are the experimental results of the first experiment. Figure 1 draws the expected influence spread achieved by the (sampled) greedy algorithm and (sampled) adaptive greedy policy under the NetScience and Wiki datasets. Here, the probability $\beta \sim N(a, b)$ means β_u for each node $u \in V$ is sampled from a truncated normal distribution whose mean is a and variance is b within the interval $[0, 1]$. Because the greedy algorithm and adaptive greedy policy are implemented by MC simulations, and its time complexity is too high, thereby we only use these two small graphs to test them in this experiment. Here, the number of MC simulations for each estimation is set to 300 in NetScience dataset and 600 in Wiki dataset. This is far from enough, just for performance comparison. For the sampled greedy algorithm, the number of random RR-sets is determined based on experience, where we give $|\mathcal{R}| = 5000 + 1000 \cdot (k/10)$ in NetScience dataset and $|\mathcal{R}| = 10000 + 2000 \cdot (k/10)$ in Wiki dataset.

We note that the expected influence spread obtained by the adaptive greedy policy and sampled adaptive greedy policy is very close, which proves the effectiveness of our sampling techniques. Under the non-adaptive settings, the performance achieved by the sampled greedy algorithm is better than that achieved by the greedy algorithm. This may be because the number of MC simulations we set for each estimation is not enough to get a precise estimation. Thus, we are more inclined to think the results obtained by the sampled greedy algorithm are more precise. Compare the performances shown as Figure 1, we find that the sampled adaptive greedy policy has an obvious advantage, which is much better than the sampled greedy algorithm. This illustrates the effectiveness of our proposed adaptive policy from one aspect. Besides, with the increase of the mean of β, there is no doubt the expected influence spread will increase. However, we observe an interesting phenomenon where the gap between the performance under the adaptive settings and non-adaptive settings seems to be shrinking. This is because the uncertainty of nodes, whether to be an active seed or not, decreases as the mean of β increases, thereby reducing the advantage of our adaptive policies.

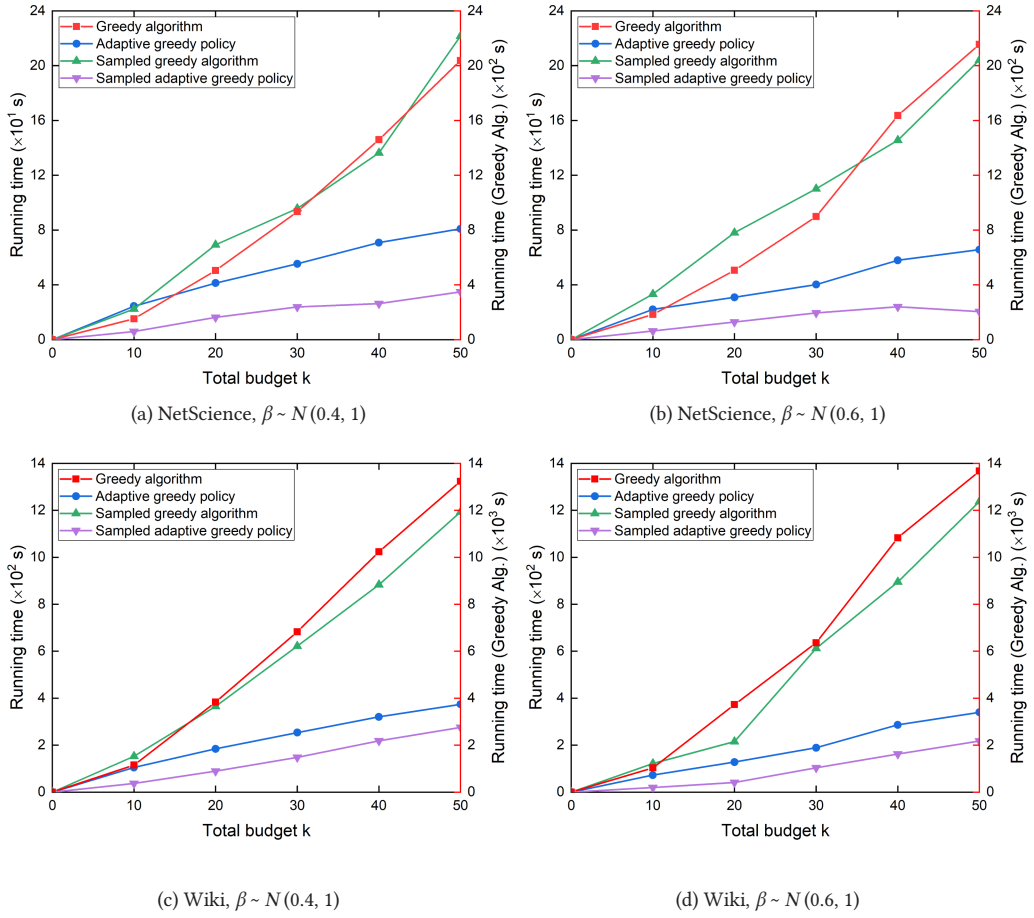


(a) NetScience, $\beta \sim N(0.4, 1)$ (b) NetScience, $\beta \sim N(0.6, 1)$

(c) Wiki, $\beta \sim N(0.4, 1)$ (d) Wiki, $\beta \sim N(0.6, 1)$

Fig. 1. The expected influence spread achieved by the (sampled) greedy algorithm and (sampled) adaptive greedy policy under the NetScience and Wiki datasets.

Figure 2 draws the running time achieved by the (sampled) greedy algorithm and (sampled) adaptive greedy policy under the NetScience and Wiki datasets. Here, in order to compare the running time of different strategies, we do not use parallel acceleration in our implementations. We note that the running time of the sampled adaptive greedy policy is smaller than that of the sampled greedy algorithm, which is counter-intuitive. This looks unreasonable because the sampled greedy algorithm only needs to generate a collection of RR-sets once and selects seed nodes in one batch, but the sampled adaptive greedy policy has to generate a new collection of RR-sets in each iteration. Why does it happen? First, the estimator of $\mu(\boldsymbol{x})$ shown as Equation (44) is more complicated than the estimator of $\Delta(u|\boldsymbol{x}, \psi)$ shown as Equation (38). Second, the number of RR-sets we give in the sampled greedy algorithm may be too much, which exceeds actual needs. At last, the sampling process will be faster and faster as the graph gets smaller in the sampled adaptive greedy policy. Then, we can see that the running time of the sampled adaptive greedy policy is less than that of the adaptive greedy policy even though the number of MC simulations is far from enough, which proves the efficiency of our sampling techniques. Compare to the running times achieved



(a) NetScience, $\beta \sim N(0.4, 1)$

(b) NetScience, $\beta \sim N(0.6, 1)$

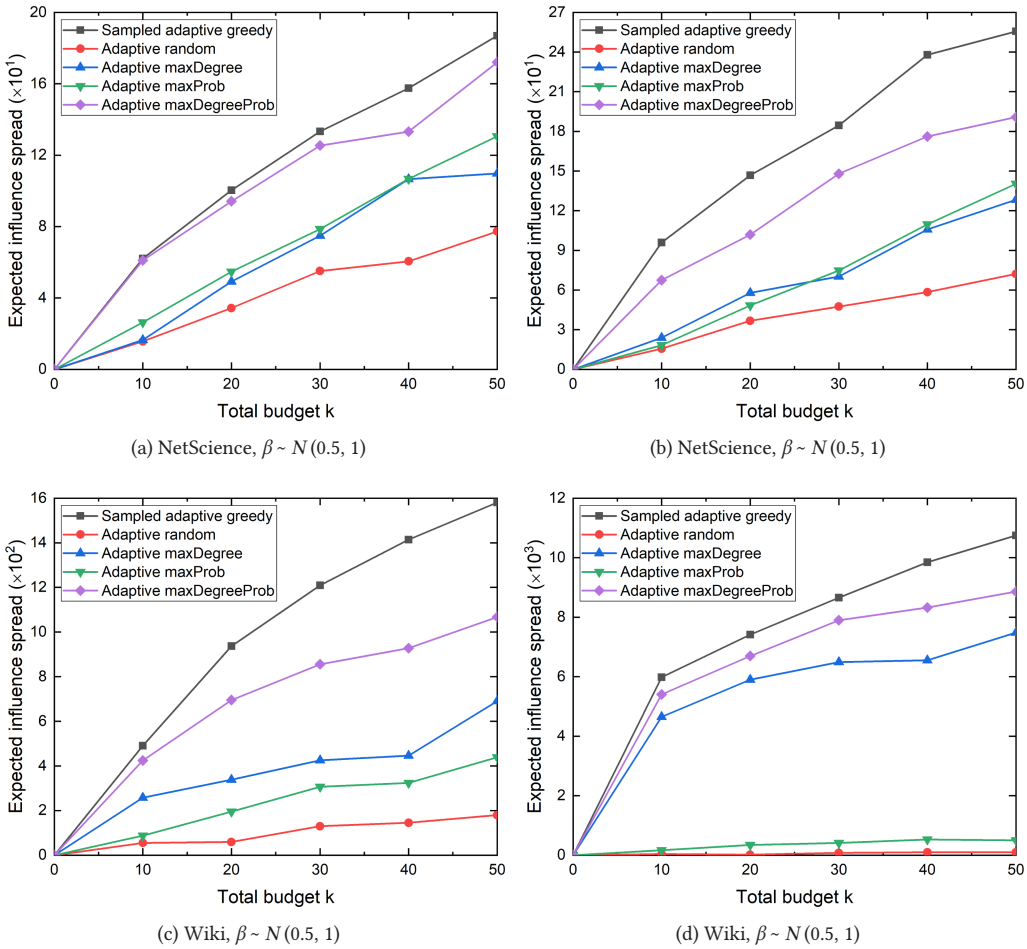
(c) Wiki, $\beta \sim N(0.4, 1)$

(d) Wiki, $\beta \sim N(0.6, 1)$

Fig. 2. The running time achieved by the (sampled) greedy algorithm and (sampled) adaptive greedy policy under the NetScience and Wiki datasets.

by the adaptive greedy policy, the greedy algorithm is very inefficient, nearly 10 times slower than the adaptive greedy policy. There are two reasons to explain this phenomenon. First, the graph that the adaptive greedy policy relies on is shrinking gradually as the number of iterations increases. Secondly, the process of reverse breadth-first search in MC simulations will be more time-consuming when the seed set is large.

Figure 3 draws the performance comparisons between our sampled adaptive greedy policy and other heuristic adaptive policies under the four datasets. We can see that the expected influence spread of any adaptive policy increases with budget k because attempting to select more seed results in a larger influence spread. The expected influence spread returned by our sampled adaptive greedy policy outperforms all other heuristic adaptive policies under any dataset, thereby its performance is the best undoubtedly. This illustrates the effectiveness of our proposed policy from another aspect. Among these heuristic adaptive policies, the adaptive maxDegreeProb policy has the largest expected influence spread, because it considers the node's degree and probability to be a seed comprehensively. The performance of other policies is unstable on different datasets. We can observe that the sampled adaptive greedy policy can obtain at least 10% gain of the expected



(a) NetScience, $\beta \sim N(0.5, 1)$

(b) NetScience, $\beta \sim N(0.5, 1)$

(c) Wiki, $\beta \sim N(0.5, 1)$

(d) Wiki, $\beta \sim N(0.5, 1)$

Fig. 3. The performance comparisons between our sampled adaptive greedy policy and other heuristic adaptive adaptive under the four datasets.

influence spread than the best heuristic adaptive policy. However, the gap between the sampled adaptive greedy policy and other heuristic adaptive policies can be affected by the dataset itself, since there are different topologies and graph realizations associated with different networks.

8 CONCLUSION

In this article, we have studied a variant of adaptive influence maximization, where the seed node we select may be unwilling to be the influencer and we can activate her many times. Because its objective function is defined on integer lattice, we propose the concepts of adaptive monotonicity on integer lattice and adaptive dr-submodularity firstly. Then, we summarize the properties of this problem and give a strict theoretical analysis about the approximation ratio of the adaptive greedy policy. Our approach can be used as a flexible framework to address adaptive monotone and dr-submodular function under the expected knapsack constraint. Combine with the-state-of-art EPIC algorithms, the sampled adaptive greedy policy is formulated, which reduces its running time significantly without losing the approximation guarantee. Eventually, we evaluate our proposed

policies on four real networks and validate the effectiveness and efficiency comparing to their corresponding non-adaptive algorithms and other heuristic adaptive policies.

REFERENCES

[1] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing social influence in nearly optimal time. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 946–957.

[2] Wei Chen, Alex Collins, Rachel Cummings, Te Ke, Zhenming Liu, David Rincon, Xiaorui Sun, Yajun Wang, Wei Wei, and Yifei Yuan. 2011. Influence maximization in social networks when negative opinions may emerge and propagate. In *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM, 379–390.

[3] Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. 2016. Robust influence maximization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 795–804.

[4] Wei Chen, Chi Wang, and Yajun Wang. 2010. 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*. 1029–1038.

[5] Wei Chen, Ruihan Wu, and Zheng Yu. 2020. Scalable lattice influence maximization. *IEEE Transactions on Computational Social Systems* 7, 4 (2020), 956–970.

[6] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *Proceedings of the 2010 IEEE International Conference on Data Mining*. IEEE, 88–97.

[7] Pedro Domingos and Matt Richardson. 2001. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 57–66.

[8] Daniel Golovin and Andreas Krause. 2011. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research* 42 (2011), 427–486.

[9] Corinna Gottschalk and Britta Peis. 2015. Submodular function maximization on the bounded integer lattice. In *Proceedings of the International Workshop on Approximation and Online Algorithms*. Springer, 133–144.

[10] Jianxiong Guo, Tiantian Chen, and Weili Wu. 2020. Continuous activity maximization in online social networks. *IEEE Transactions on Network Science and Engineering* 7, 4 (2020), 2775–2786.

[11] Jianxiong Guo and Weili Wu. 2019. A novel scene of viral marketing for complementary products. *IEEE Transactions on Computational Social Systems* 6, 4 (2019), 797–808.

[12] Jianxiong Guo and Weili Wu. 2020. Influence maximization: Seeding based on community structure. *ACM Transactions on Knowledge Discovery from Data* 14, 6 (2020), 66:1–66:22.

[13] Jianxiong Guo and Weili Wu. 2020. A k-hop collaborate game model: Adaptive strategy to maximize total revenue. *IEEE Transactions on Computational Social Systems* 7, 4 (2020), 1058–1068.

[14] Kai Han, Keke Huang, Xiaokui Xiao, Jing Tang, Aixin Sun, and Xueyan Tang. 2018. Efficient algorithms for adaptive influence maximization. *Proceedings of the VLDB Endowment* 11, 9 (2018), 1029–1040.

[15] Daisuke Hatano, Takuro Fukunaga, and Ken-Ichi Kawarabayashi. 2016. Adaptive budget allocation for maximizing influence of advertisements. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 3600–3608.

[16] Keke Huang, Jing Tang, Kai Han, Xiaokui Xiao, Wei Chen, Aixin Sun, Xueyan Tang, and Andrew Lim. 2020. Efficient approximation algorithms for adaptive influence maximization. *The VLDB Journal* 29, 6 (2020), 1385–1406.

[17] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 137–146.

[18] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. Retrieved from http://snap.stanford.edu/data.

[19] Hung T. Nguyen, My T. Thai, and Thang N. Dinh. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *Proceedings of the 2016 International Conference on Management of Data*. 695–710.

[20] Matthew Richardson and Pedro Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 61–70.

[21] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The network data repository with interactive graph analytics and visualization. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. Retrieved from http://networkrepository.com

[22] Tasuku Soma and Yuichi Yoshida. 2015. A generalization of submodular cover via the diminishing return property on the integer lattice. In *Proceedings of the Advances in Neural Information Processing Systems*. 847–855.

[23] Tasuku Soma and Yuichi Yoshida. 2017. Non-monotone dr-submodular function maximization. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

[24] Tasuku Soma and Yuichi Yoshida. 2018. Maximizing monotone submodular functions over the integer lattice. *Mathematical Programming* 172, 1–2 (2018), 539–563.

[25] Lichao Sun, Weiran Huang, Philip S. Yu, and Wei Chen. 2018. Multi-round influence maximization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2249–2258.

[26] Jing Tang, Keke Huang, Xiaokui Xiao, Laks V.S. Lakshmanan, Xueyan Tang, Aixin Sun, and Andrew Lim. 2019. Efficient approximation algorithms for adaptive seed minimization. In *Proceedings of the 2019 International Conference on Management of Data*. 1096–1113.

[27] Jing Tang, Xueyan Tang, Xiaokui Xiao, and Junsong Yuan. 2018. Online processing algorithms for influence maximization. In *Proceedings of the 2018 International Conference on Management of Data*. 991–1005.

[28] Shaojie Tang and Jing Yuan. 2020. Influence maximization with partial feedback. *Operations Research Letters* 48, 1 (2020), 24–28.

[29] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence maximization in near-linear time: A martingale approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 1539–1554.

[30] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. 75–86.

[31] Guangmo Tong and Ruiqi Wang. 2020. On adaptive influence maximization under general feedback models. *IEEE Transactions on Emerging Topics in Computing* (2020), 1–1. DOI: https://doi.org/10.1109/TETC.2020.3031057

[32] Jing Yuan and Shaojie Tang. 2017. No time to observe: Adaptive influence maximization with partial feedback. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 3908–3914.

Received August 2020; revised November 2020; accepted January 2021