

Efficient Algorithms for Social Influence Problems with Large Networks

Efektivní algoritmy pro problémy se sociálním vlivem u velkých sítí

Nguyen Huy Phuong Pham

PhD Thesis

Supervisor: prof. RNDr. Václav Snášel, CSc.

Ostrava, 2022

Thesis Assignment

This is a sample assignment for a bachelor's thesis, master's thesis or dissertation. You can download the genuine assignment from the Edison system.

Here is the end of a very long assignment over two pages.

Abstrakt a přínos práce

V posledních letech je závratná exploze dat a informací výsledkem sociálních sítí s miliony až miliardami uživatelů, jako jsou Facebook, YouTube, Twitter a LinkedIn. Uživatelé mohou využívat online sociální sítě (OSNs) k rychlému obchodování s informacemi, komunikaci s ostatními uživateli a udržování jejich informací v aktuálním stavu. Výzva šíření informací na sociálních sítích, která se v praxi objevuje, vyžaduje efektivní řešení správy informací, jako je šíření užitečných informací, maximalizace vlivu přenosu informací a zabránění šíření dezinformací, fám a virů. Motivováni výše uvedenými problémy zkoumáme problém šíření informací na OSN. Tento problém studujeme na základě dvou modelů, Independent Cascade (IC) a Linear Threshold (LT) a klasické Influence Maximization (IM) v online sociálních sítích. Kromě toho zkoumáme různé aspekty problémů s rychlým zasíláním zpráv, jako jsou změny rozpočtu, témata zájmu, více konkurentů a další. Kromě toho také zkoumáme a aplikujeme teorii kombinatorických optimalizačních problémů k vyřešení jednoho ze současných problémů v sociálních sítích, maximalizujeme vliv na skupiny a témata v sociálních sítích.

Obecně lze říci, že hlavní cíle Ph.D. návrh diplomové práce je následující.

1. Zkoumáme problém Multi-Threshold pro IM, což je varianta problému IM s prahovými omezeními. Navrhujeme účinný algoritmus, který IM pro více prahů v sociální síti. Zejména vyvíjíme nový algoritmický rámec, který může použít řešení pro menší práh k nalezení prahu většího.
2. Studujeme problém maximalizace vlivu skupiny a zavádíme účinný algoritmus maximalizace vlivu skupiny s více výhodami, než je vliv každého uzlu v sítích, pomocí nové vzorkovací techniky k odhadu funkce skupiny epsilon. Navrhujeme také aproximační algoritmus pro odhad více kandidátních řešení s teoretickou zárukou.
3. Zkoumáme přístup pro maximalizaci vlivu s k-téma pod omezeními v rozsáhlé síti. Konkrétněji budeme studovat novou metriku, která kombinuje optimalizační algoritmus pro zlepšení aproximačního algoritmu z hlediska kvality řešení a doby běhu na základě kliky a komunity v komplexních sítích.

Klíčová slova

Online sociální sítě, maximalizace vlivu, virální marketing, aproximační algoritmy, šíření informací.

Abstract and Contributions

In recent years, the dizzying explosion of data and information results from social networks with millions to billions of users, such as Facebook, YouTube, Twitter, and LinkedIn. Users can use online social networks (OSNs) to quickly trade information, communicate with other users, and keep their information up-to-date. The challenge of spreading information on social networks that arises in practice requires effective information management solutions, such as disseminating useful information, maximizing the influence of information transmission, and preventing disinformation, rumors, and viruses from being disseminated. Motivated by the above issues, we investigate the problem of information diffusion on OSNs. We study this problem based on two models, Independent Cascade (IC) and Linear Threshold (LT), and classical Influence Maximization (IM) in online social networks. In addition, we investigate various aspects of IM problems, such as budget variations, topics of interest, multiple competitors, and others. Moreover, we also investigate and apply the theory of combinatorial optimization problems to solve one of the current concerns in social networks, maximizing the influence on the groups and topics in social networks.

In general, the main goals of the Ph.D thesis proposal are as follows.

1. We investigate the Multi-Threshold problem for IM, which is a variant of the IM problem with threshold constraints. We propose an efficient algorithm that IM for multiple thresholds in the social network. In particular, we develop a novel algorithmic framework that can use the solution to a smaller threshold to find that of larger ones.
2. We study the Group Influence Maximization problem and introduce an efficient group influence maximization algorithm with more advantages than each node's influence in networks, using a novel sampling technique to estimate the epsilon group function. We also devised an approximation algorithm to estimate multiple candidate solutions with theoretical guarantee.
3. We investigate an approach for Influence Maximization problem with k-topic under constraints in social network. More specifically, we also study a streaming algorithm that combines an optimization algorithm to improve the approximation algorithm and theoretical guarantee in terms of solution quality and running time.

Keywords

Online Social Networks, Influence Maximization, Viral Marketing, Approximation Algorithms, Information Diffusion.

Acknowledgement

I would like to express sincere gratitude to my supervisor, prof. RNDr. Václav Snášel, CSc., for the great support and help during this research. In addition, I would like to express my honest thanks to Dr. Pham Van Canh, Phenikaa University, who has always encouraged, supported, and helped me in this challenging research. Furthermore, I also would like to thank VSB-Technical University of Ostrava, Ho Chi Minh City University of Food Industry (HUFİ), European Cooperation Center, Ton Duc Thang University, colleagues and friends who encouraged and supported for me. Especially, I would like to thank my family who always beside me in during my Ph.D study.

Finally, I would like to thank all those who helped me with the work, because without them this work would not have been finished.

Contents

List of symbols and abbreviations	9
List of Figures	10
List of Tables	11
1 Introduction	13
1.1 Motivation and goals	13
1.2 Purpose of the research	14
1.3 Thesis structure	15
2 State-of-the-art	17
2.1 Preliminaries	17
2.2 The approach of Influence Maximization	19
2.3 Related works	22
2.4 Discussion	29
3 Multi-Threshold benefit for Influence Maximization	30
3.1 Motivation	30
3.2 Problem definition	31
3.3 Multi-threshold for Influence Maximization problem	32
3.4 Experiment	38
3.5 Discussion	43
4 Groups Influence Maximization in Large Network	45
4.1 Motivation	45
4.2 Problem definition	48
4.3 Proposed algorithms	51
4.4 Experiment	67
4.5 Discussion	74

5	Influence Maximization with k-topic in Social Network	76
5.1	Introduction	76
5.2	Problem definition	79
5.3	Proposed algorithm	80
5.4	Experiment	84
5.5	Discussion	89
6	Conclusion and Future work	90
6.1	Summary of results	90
6.2	Future work	91
	Bibliography	92
	List of own publication activities	108
	Publications and Outcomes Related to Thesis	108

List of symbols and abbreviations

OSNs	– Online Social Networks
IM	– Influence Maximization
LT	– Linear Threshold
IC	– Independent Cascade
RR	– Reachable Reverse
IT	– Influence Threshold
CELF	– Cost-effective outbreak detection based on a Lazy-forward
IMM	– Influence Maximization via Martingales
TIM/TIM+	– Two-phase Influence Maximization
SSA/D-SSA	– Stop-and-State Algorithm/Dynamic-Stop-and-State Algorithm
CTVM	– Cost-aware Targeted Viral Marketing
BCT	– Billion-scale Cost-aware Targeted
RIS	– Reverse Influence Sketch
SKIS	– Influence Oracle via iis Sketch
SKIM	– Sketch-based Influence Maximization
Tip-Top	– Tiny Integer Program with Theoretically OPTimal results
MGC	– Maximizing Group Coverage
BkSM	– Budgeted k-submodular Maximization
KB-TIM	– Keyword-Based Targeted Influence Maximization
MIA	– Maximum Influence Arborescence
BS	– Benefit Sample
MBT	– Multiple Benefit Threshold
ESSM	– Efficient Sampling for Selecting Multiple
TGI	– Threshold Benefit for Groups Influence
GBS	– Group-Based Sampling
GIM	– Group Influence Maximization
GRR	– Group Reverse Reach
GIA	– Group Influence Algorithm

List of Figures

3.1	Comparison about the Costs of seed sets between ESSM and other algorithms within threshold T_i from 300 to 9,000	41
3.2	Comparison of Running time between ESSM and other algorithms within threshold T_i from 300 to 9,000	42
4.1	Total cost compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800	68
4.2	Running time compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800	69
4.3	Size of seed set returned by GIA, EGI and other algorithms under the UC setting	72
4.4	Ratio of number of influenced groups over K of GIA, EGI and other algorithms under the UC setting	72
4.5	Running time of GIA, EGI and other algorithms under the UC setting	73
4.6	Size of seed set returned by GIA, EGI and other algorithms under the GC setting	74
4.7	Ratio of number of influenced groups over K of GIA, EGI and other algorithms under the GC setting	74
4.8	Running time of GIA, EGI and other algorithms under the GC setting	75
5.1	Performance of algorithms in experiment of small datasets with threshold T from 100 to 1000	86
5.2	Performance of algorithms in experiment of medium datasets with threshold T from 100 to 1000	88

List of Tables

3.1	MBT experimental datasets	40
3.2	Memory usage of compared algorithms ESSM and other algorithms within threshold T_i from 300 to 9,000(MB)	44
4.1	TGI experimental datasets	67
4.2	Memory usage compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800	70
4.3	GIM experimental datasets	71
5.1	Table of the usually used notations	78
5.2	kSC experimental datasets	84
5.3	Num of Solution for each algorithm based-on dataset num nodes	87

Chapter 1

Introduction

This chapter of the thesis discusses the Influence Maximization problem in social networks. The first section shows the motivation and goals of this work. On the basis of motivation, the purpose of this thesis is also introduced in the next section. In addition, the last section presents in more detail the structure of the thesis.

1.1 Motivation and goals

Nowadays, social networks with billions of users have become an essential platform for communication and knowledge sharing. For example, many users of Online Social Networks (OSNs) can exchange information with each other quickly and smoothly. Many businesses have used the word-of-mouth effect to advertise and introduce their products to many potential customers. The authors of the scientific articles used to connect through the same research fields in the citation network. In that context, one of the most exciting research directions has been the information diffusion problem in OSNs recently. Many researchers have introduced information diffusion models and variations used in studying these problems, generally divided into two categories such as influence maximization and influence block. How to select seeds set in a social network, through which the spread of influence under some certain diffusion models can reach the maximum, is a major issue considered in social network analysis. This problem is known as the Influence Maximization problem. The influence maximization problem is a primary problem in viral marketing, which is to find a set of seeds of k individuals/nodes in a social network that could maximize the spread of influence. Due to its $\#NP$ -hard, designing a good algorithm for the Influence Maximization problem is a very challenging task.

In addition, the problem of information diffusion has been studied in various fields in practice, but now there are exciting trends such as group influence, community characteristics, social welfare, misinformation, rumor control, k -topic, and others. For instance, the problem arises of how to influence maximization on multiple thresholds instead of a single threshold,

as in previous works. **The first aim of our study is to solve the problem of influence maximization for the multi-threshold case in the social network.** Furthermore, a matter of great concern today is whether group decisions affect many groups on social networks or whether a group of people implements organizational policies in certain situations. Group influence can therefore play a significant role in an organization. So influencing a group with a minimal budget is a difficult task. Therefore, **our second aim is to study the problem of group influence maximization with budget constraints in terms of minimum cost and benefit in large networks.** On the other hand, how to influence the maximization of k topics in OSNs has been attracting a lot of attention from researchers in recent times. Hence, **our third aim in this study focuses on influence maximization on k independent topics with many types of constraints under information propagation models in social networks.** Therefore, solving these issues using approximate approaches is a challenging task in the literature.

More specifically, combinatorial optimization problems have been studied in various practical scenarios such as data summarization, influence maximization with k topics, monitor placement, and feature selection, which has received more attention in the literature. Since then, solving these issues using approximation algorithms under more constraints has been a challenge on social networks.

1.2 Purpose of the research

The main objective of this doctoral thesis is to study the following issues.

- Proposing an efficient algorithm in terms of multi-threshold Influence Maximization problem for practical application which has previously extensively greedy algorithms and heuristic algorithms.
 - An efficient technique with theoretical guarantee named Efficient Sampling for Selecting Multiple Seed Sets (ESSM) for Multiple Benefit Thresholds (MBT problem).
 - A novel algorithmic framework that can use a solution to a smaller threshold to find that of larger threshold.
 - The sampling technique with martingale theory to estimate benefit function, named Benefit Sample (BS).
 - Conduct some experiments on MBT problem.
- Developing new efficient algorithms and models for group influence maximization that focus on the design of improved optimal solutions and algorithmic complexity.

- Two efficient algorithms for Groups Influence Maximization with approximation guarantees, named Threshold Benefit for Groups Influence (TGI problem) and Groups Influence with Minimum Cost (GIM problem).
- A novel group reachable reverse sample concept that helps to estimate the group influence function.
- A framework algorithmic to find good candidate solutions with provable guarantees.
- A bi-criteria approximation algorithm, named Groups Influence Approximation (GIA) and Exact Influence Groups (EGI), for generating multiple candidate solutions with theoretical bounds.
- Extensive experiments conduct on some real social large networks.
- Proposing an approximation approach for influence maximization with k -topic under more constraint in large-scale network.
 - Submodular maximization and k -submodular function maximization.
 - An efficient streaming maximizing k -submodular function subject to multiple topics with bi-criteria approximation guarantee.
 - Implement the proposed algorithm to evaluate the performance of k SC problem in applying Influence Maximization with k topics.
- Experiment and evaluation of the proposed method on a variety of real-world datasets with large networks.

1.3 Thesis structure

The following is the structure of the research proposal:

- Chapter 1 provides an overview of motivation and goals of this work, purpose of research and structure of Ph.D thesis.
- Chapter 2 presents an overview information diffusion models, influence maximization approach and relate works.
- Chapter 3 provides an overview of the proposed study’s main objectives, Multi-Threshold Benefit for Influence Maximization, including research motivation, issue definition for Multiple Benefit Thresholds, and discuss the result of our proposed algorithm.
- Chapter 4 discusses the aim of Groups Influence Maximization in Large Networks, which mentions two problems, Threshold Benefit for Groups Influence in OSNs and Groups Influence with Minimum Cost in OSNs in terms of benefit and cost.

- We study Influence Maximization with the k kind of topic shown in Chapter 5, focusing on the k -submodular function maximization problem.
- Finally, in Chapter 6, the conclusion and our contribution are presented. In the final section, we present a list of our published works and references.

Chapter 2

State-of-the-art

This chapter shows the related work of influence maximization in OSNs that supports the study of the specific IM problem. In particular, the background of Influence Maximization problems show that in the first section such as information propagation models, and the approach of IM. The last section describes in more detail related work in this field.

2.1 Preliminaries

In this section, we introduce the network model in a social network as a graph for information diffusion models. Information diffusion models are the solid background for studying information propagation problems. Kempe et al. [1] first introduced Influence Maximization as a discrete optimization problem with two well-known models named *Independent Cascade (IC)* and *Linear Threshold (LT)*. In these models, they formulated Influence Maximization (IM) problem which aims to select k nodes so that the expected number of influenced users in information diffusion is maximized and proposed $1 - 1/e$ approximation algorithm for this problem. Due to the commercial values of IM, it has attracted much attention recently and then several works focused on IM problem in the following aspects: proposing scalability and efficiency algorithms [2, 3, 4, 5] and studying variants of IM problems [6, 7, 8, 9, 10, 11].

2.1.1 Information propagation process

First, we describe the information propagation process as the process of propagation between vertices in a directed graph $G = (V, E)$, where V is the set of vertices of the graph with the number of vertices $n = |V|$ and E is the set of associated edges connect the vertices of the graph with the number of edges $m = |E|$.

The propagated information is derived from the seed set $S \subseteq V$. The propagated information is derived from the seed set $S \subseteq V$. When the propagating information reaches each vertex $v \in V$, the activation state of vertex v is changed from inactive to active. The process

of spreading the active state between these vertexes is the information propagation process. The set of vertices $S_t \subseteq V$ is the set of vertices activated at time t with $t = 0, 1, \dots$, the propagation process between t and $t + 1$ is $S_{t+1} = f(F, S_t, t)$. Until at the time $t + 1$ no more vertices are activated, the propagation stops at the time t : $S_t = S_{t+1}$. Finally, the Influence Function $\sigma(S)$ is the number of vertices active after information diffusion from the seed set S .

2.1.2 Information propagation under Independent Cascade model

Given a social network is considered a directed graph $G = (V, E)$ where V is the set of nodes representing individuals and E is the set of edges representing relationships. We also denote $n = |V|$ and $m = |E|$. Let $N_{in}(v)$ and $N_{out}(v)$ as two sets of in-neighbors and out-neighbors of a node v , respectively.

In this model, each edge $e = (u, v) \in E$ has a probability $p(u, v) \in (0, 1)$ that represents the influence propagation from u to v . The diffusion process from a seed set $S \subset V$ happens in following steps.

- At step $t = 0$, all nodes in S is activated.
- At step $t \geq 1$, for an activated node u in previous steps, it has a single chance to activate each inactive neighbour v with the successful probability $p(u, v)$. An activated node remains active till the end of the diffusion process.
- The propagation process ends at step t if there is no new activated node in this step.

2.1.3 Information propagation under Linear Threshold model

In this model, each directed edge $(u, v) \in E$ is associated with an influence weight $w(u, v) \in [0, 1]$ satisfying

$$\sum_{u \in N_{in}(v)} w(u, v) \leq 1$$

Given a set of seed nodes $S \subseteq V$. In LT model, each node $v \in V$ has two possible states, *active* and *inactive* and the influence cascades in G as follows. Firstly, every node $v \in V$ uniformly chooses a threshold $\theta_v \in [0, 1]$, which represents the weighted fraction of v 's neighbors that must be active to activate v . Next the influence propagation happens in round $t = 0, 1, 2, \dots$ as follow:

- At round 0, we activate nodes in S , and set all other nodes inactive.

- At round $t \geq 1$, an inactive node v is activated if weighted number of its activated neighbors are greater than or equal to its threshold, i.e.,

$$\sum_{\text{in-activated neighbors } v} w(u, v) \geq \theta_v$$

- Once a node becomes activated, its status remains in the process of spreading. The influence propagation ends when no more nodes can be activated.

2.1.4 Information propagation under Live-edge model

The authors in [1] showed IC model is equivalent to the reachability in a random graph g , called *live-edge* or *sample graph*. We generate a sample graph g with the set of nodes be V_g and the set of edges be E_g by: (1) setting $V_g \leftarrow V$, and (2) selecting $e = (u, v) \in E$ into E_g with probability $p(e) = p(u, v)$. The selected edges are called *live* and all other edges are called *blocked*.

The live-edge model first generates a sample graph $g = (V_g, E_g)$ by selecting $e = (u, v) \in E$ into with probability $p(e) = p(u, v)$ and not selecting $e = (u, v) \in E$ with probability $1 - p(u, v)$. The sample graph g is generated with probability:

$$\Pr[g \sim G] = \prod_{e \in E_g} p(e) \cdot \prod_{e \in E \setminus E_g} (1 - p(e)) \quad (2.1)$$

The live-edge propagation under the LT model is calculated with the probability of generating a sample graph g from G as follows:

$$\Pr[g \sim G] = \prod_{u \in V} p(u, g, G) \quad (2.2)$$

In which, $p(u, g, G)$ is the probability of choosing the edge corresponding to the vertex u :

$$p(u, g, G) = \begin{cases} w(u, v), & \text{if } (u, v) \in g \\ 1 - \sum_{u \in (u, v)} w(u, v), & \text{if } (u, v) \notin g \end{cases} \quad (2.3)$$

2.2 The approach of Influence Maximization

There are two main approaches to finding algorithms for IM problems: Approximation algorithms that guarantee the theoretical solution quality or Heuristic algorithms that do not guarantee the optimal ratio.

2.2.1 Greedy approach

Based on the increased monotony and submodularity, Kempe first proposed the greedy algorithm for the IM problem, which gives an approximate ratio of $1 - 1/e$. For the optimal problem of finding the greatest value, called S' is the set of solutions given by the greedy algorithm, S^* is the optimal solution, and the optimal ratio $p \in (0, 1]$, is defined as follows:

$$\frac{\sigma(S')}{\sigma(S^*)} \leq p \quad (2.4)$$

This ratio ensures that, in the worst case, the proposed algorithm gives a solution of p times the quality of the optimal solution. The greedy algorithm works in a simple step-by-step sequence.

$$\delta(S, u) = \sigma(S + \{u\}) - \sigma(S) \quad (2.5)$$

In each step, we select the vertex with the incremental influence reaching the maximum value until the number of vertices is k . Although this algorithm gives an approximation of $1 - 1/e$, it fails in practice because it is $\#P$ -hard to compute the objective function $\sigma(S)$.

Then, let \mathbb{R} be the number of Monte-Carlo simulations to estimate the objective function. The complexity in this case is $O(kn\mathbb{R}(m + n))$. To let the greedy algorithm achieve an approximate ratio of $1 - 1/e - \epsilon$, with ϵ being any given parameter, the complexity is $O(\epsilon^{-2}k^3n^2m \log n)$ [12]. This is a fairly large complexity and is difficult to apply to medium or large-sized networks.

Leskovec et al. [13] proposed the Lazy Greedy algorithm based on the submodular property of the objective function. Based on this property, the algorithm does not evaluate the vertices with low $\sigma(S, u)$ in the next step in the greedy algorithm. Experimental results show that this method gives nearly the same results as the greedy algorithm, but the running time is up to 700 times faster.

2.2.2 Heuristic method

The most serious difficulty in coming up with approximation algorithms for IM is the need for a good solution selection strategy and proof of the algorithm's approximation ratio. While it is possible to use simple, intuitive tools that also give good enough results in a short time. Based on the properties of each algorithm, the author divides them into groups as follows: (1) group of algorithms based on measure, (2) group of algorithms based on path, and (3) group of algorithms based on structure community architecture.

One of the simplest ideas in using heuristic information to select seed vertices is to select by measures in the input graph structure [14] [15]. Commonly used degrees are degree, closeness,

and betweenness centrality (BC). In the IM problem, the authors often use these algorithms as the base algorithm [16] [17] [18] [18]

Chen et al. based on the Maximum Influence Path to propose some algorithms for the IM problem on both LT and IC models [16] [17] [18]. Based on this idea, they construct directed acyclic graphs (DAGs), then approximate the effect of a vertex on the newly constructed graph. Two proposed algorithms for IC and LT models are PMIA and LDAG [17] [18]. Experimental results show that these algorithms give solution quality close to greedy algorithms but have better running time. Moreover, it can be applied to large networks of millions of vertices.

Goyal et al. [18] have shown that calculating the influence in LDAG can be solved by finding the edge vertices and then enumerating all single paths to it. They proposed the SIMPATH algorithm based on this idea, and experimental results show that SIMPATH is better than LDAG. Kim et al. [19] proposed another algorithm based on the independent influence curve named IPA (Independent Path Algorithm). This algorithm has the advantage of being parallelizable.

Path-based algorithms balance efficiency and running time and metric and greedy algorithms. However, the above algorithms also reveal some disadvantages, such as not guaranteeing the optimal ratio to the optimal solution. In empirical research, path-based algorithms give pretty bad results on some datasets compared to greedy algorithms. On the other hand, there are many paths in a graph, so these algorithms also consume a lot of memory to store them.

Another approach to the IM problem is to use the community property to select the seed set. The general idea of this approach is: vertices belonging to different communities have a low probability of influencing each other. Thus, instead of searching for seeds in a large network, we can search for seeds in individual communities and combine them. The typical studies for this method are [20] [21] [22]. In general, the algorithm group can be divided into three phases: (1) community detection, (2) generation of candidate nodes, and (3) finding seed nodes. Community-based influence maximization algorithms are often faster than traditional greedy algorithms. Furthermore, since it is generally assumed that the different communities are independent, these algorithms can support parallelism.

In addition to the above studies, some authors use metaheuristic techniques for IM problems, such as swarm optimization (PSO) [23], metallurgical simulation (SA) [24], and genetic algorithms [25] [26]. However, the limitation of these algorithms is the scalability of large networks because the computation time is relatively high due to the fact that the objective function calculation is not well solved.

2.2.3 Reverse influence sampling method

Borg et al. [2] provide another technique for approximating the problem of information propagation maximization. They developed a random sampling algorithm called Reverse Influence Sampling (RIS) using an inference approach based on the notion of Reverse Reachable set and its randomized variant. The foundation of this approach is cause-and-effect inference, which is used to determine who is the most likely cause of the node activated in the input graph by moving backward in the effect propagation.

Definition 1 (Random Reverse Reachable Set) *Let v denote a vertex in G and g denote a graph obtained by removing each edge e in G with a probability of $1 - p(e)$. Then, the reverse reachable set for v in g , denoted by $RR_g(v)$, is given by:*

$$RR_g(v) = \{u | \exists e_{u \rightarrow v} \in g\} \quad (2.6)$$

When a vertex u appears in the R_v , it has the chance of influencing v . In other words, the more random Reverse Reachable sets appear the vertex u , the more probability it is that it will influence propagation in the initial graph. RIS requires θ , the number of Random Reverse Reachable sets (\mathbb{R}), and then solves the Maximum Coverage problem on the \mathbb{R} set of Reverse Reachable sets to discover the set of k seed nodes. The RIS algorithm is a randomized algorithm with an approximation guarantee of $1 - 1/e - \epsilon$.

2.3 Related works

With the rise in popularity of social networks as a result of technological advancements, social networks have progressively become a common requirement for people, with one of the most pressing issues being the need to keep up with the latest news in a timely manner. Users can quickly contact each other on the social networking platform, building a massive information and communication network. Users of a social network construct communities with shared interests, habits, trends, or goals, resulting in the creation of a unified body of information that is disseminated within the community. Users will be reached more easily by communities with similar interests. OSNs are an extremely effective communication channel for quickly sharing information, promoting, and marketing to a large audience and quickly reaching the right fields. The fields of promotion and marketing are spread by taking advantage of the word-of-mouth effect, which has the advantage of connecting users easily.

Firstly, we introduce related works to the classical influence maximization problem and variants of the IM problem. As the first authors to publish on the IM problem [1], the authors proposed the combinatorial optimization problem and two classical models for the IM problem, the LT model and the IC model for the dissemination purposes of the IM problem affect.

Inheriting Kempe’s work, many variations on the IM problem have been introduced because of its important role in many practical applications, such as viral marketing [10, 27], profit maximization [28, 29, 30], social recommendations, healthcare, rumor control, etc. Kempe’s conclusion demonstrated that the IM problem is $\#NP$ -hard and that a greedy algorithm has an approximate scale of $1 - 1/\epsilon$ due to the objective function’s submodularity.

Due to the widespread use of IM in commerce, several efficient algorithms have been proposed to solve the problem in large-scale networks, including the approximation algorithm [10, 5, 2, 31], and heuristic methods without guarantee theory [3, 27, 32]. Borg et al. [2] introduce RIS, a reverse sampling method that lays the groundwork for developing linear-time algorithms with approximate solutions. The main idea behind Borg’s algorithm is that they proposed a sampling technique, known as the Reverse Reachable (RR) set, for estimating the number of affected nodes in random information propagation models, as well as an algorithmic framework for finding solutions on generated samples within theoretical bounds.

In reality, many issues have arisen in order to maximize the potential of social networks, as well as to find effective solutions for controlling and disseminating beneficial information based on demand. The aim of the IM problem is to search for seed users so that the information can reach as many people as possible and provide the best benefits according to demand [17, 33]. In addition, a number of other criteria have formed variations such as the lowest total cost, choosing the right number of k users, what is the best benefit, the threshold to spread influence.

Another problem that is widely studied and applied today is link prediction [34, 35]. Link prediction is the problem of predicting the existence of a link between two entities in the network. The applications of link prediction today such as predicting co-authors in citation networks, predicting economic fluctuations, predicting product marketing trends to users, etc., all these predictions are researched and given methods to optimize accuracy and efficiency.

In addition, the problem of detecting community structure in social networks has also received more attention recently in OSNs research [32, 36, 37, 38, 39]. Detecting the community structure in social networks helps to identify trends and groups of similar interests, allowing businesses to better target and reach potential customers who are more interested in their products.

Fake news has recently gained in popularity and caused significant damage in a variety of fields, including the economy, politics, and so on. With the advantage of quickly disseminating information, online social media platforms have become an ideal environment for the spread of fake news, which will affect user’s access to this fake news if not promptly prevented. The goal of the problem of preventing misinformation is to find the original source of news distribution, as well as solutions to stop information from spreading quickly [40, 41, 42, 43].

Previously, Leskovec et al. [44] published the CELF algorithm in their study on Cost-effective Outbreak Detection in Networks. To optimize the $\#NP$ -hard effect propagation

problem, CELF is based on optimization of the “lazy-forward” method, which is an algorithm to improve the performance of greedy and prioritizing algorithms for large networks. Soon after, Goyal et al. [45] proposed an improved algorithm CELF++ to increase the time cost by 35-55% compared to CELF.

Tang et al. [5] proposed an improved effect maximization algorithm for the state-of-the-art algorithm with the worst case, resulting in significantly better experimental performance. The IMM algorithm [46], which is based on martingales-based estimation techniques, produces accurate results at a low cost and supports a larger information propagation model than previous methods.

Popular methodologies like TIM+ or IMM take a long time to research in billion-scale social networks. Nguyen et al. [31] offer a stop-and-stare (SSA) technique and its dynamic D-SSA algorithm, demonstrating that the running time of this method is faster than that of other state-of-the-art IMM Methods. The memory overhead of this method is rather high, especially in billion-scale networks, which is a restriction.

We refer to the problem that is close to our study, the Influence Threshold (IT) problem, which searches for the seed set S with the smallest size such that the propagation of the effect information reached the threshold T . Goyal et al. [33] first studied this IT problem under IC model, by using the monotone submodular property of the influence function, and then proposed a greedy algorithm combined with a Monte Carlo simulation method to estimate influence spread. Tang et al. [4] proposed a near-optimal time complexity algorithm with novel heuristics to improve empirical efficiency. Borodin et al. [47] evidence that the original greedy approach should be upgraded and a natural model that is compatible with the greedy approach proposed.

Another project related to our issue, Nguyen and et al. launched a research problem called Cost-aware Targeted Viral Marketing (CTVM) [10], which studied how to find the most cost-effective user seed set. This CTVM problem proved to be an #NP-Hard problem, so they came up with an approximation algorithm called BCT to solve this problem in a billion-scale network. It is the first method to approach dense networks with the fastest processing time when compared to other state-of-the-art algorithms. The author of this paper considers the benefits of each node with the goal of determining the seed set within the budget limit that maximizes the total benefit. They use a new sampling method and $1 - 1/\sqrt{e} - e$ to estimate the total benefit. We also provide a problem-solving algorithm based on this sampling method; however, CTVM cannot just be directly tuned to solve our problem.

Soon after, Nguyen et al. [48] introduced a new sampling method, named SKIS. SKIS improves estimation quality while reducing processing time and memory costs, according to born with RIS and SKIM. Furthermore, using SKIS to execute an effect maximization search enhances the quality of the answer much more than using greedy algorithms. When

comparing the fastest DSSA algorithm based on RIS with the SKIS-based method, the SKIS-based approach is 10 times faster and decreases memory by 4 times.

Tip-Top algorithm [49] is proposed to solve large social networks such as Twitter, in which Tip-Top focuses on reducing the number of randomly generated samples as much as possible, providing a solution for CTVM. more exactly. Tip-Top provides the first (almost) exact solution to the microfinance problem with an approximate rate of $(1 - \epsilon)$, with experiments showing a slight improvement rate of up to 98% for microfinance solutions.

Moreover, the Influence Maximization problem based on the community structure has been concerned [50, 51, 52]. The INCIM algorithm [53] estimates each node’s propagation value as a combination of its local and global influences in order to measure each node’s influence in its community as well as the influence of each community in the input graph.

Beni et al. [54] propose TI-SC, a survey-based community discovery method. The TI-SC algorithm chooses influential nodes by examining the relationships between core nodes and the scoring ability of other nodes. The score is updated after selecting each seed node to reduce overlap in seed node selection.

Recently, Xuanhao Chen [32] proposed a community-based Influence Maximization model to study the influence maximization problem in LBSN, taking both the community structure and users’ space-time behavior into the user. Their work introduces two algorithms: one for detecting communities in LBSNs based on user mobility and another for determining the most influential individuals based on the community.

Secondly, we introduce related works to the variants of the IM problem based on group influence. Besides the problem of maximum influence based on the node, many authors/researchers have been interested in the problem of maximum influence on the group in recent years. As we all know, each user on a social network frequently joins a specific group that shares characteristics such as similar interests, locations, or interests in specific topics. J. Zhu [55] proposed a framework for selecting k seed users that combined the benefits of activated groups with the propagation costs of influence in order to maximize the expected return. They used the IC model to train an information diffusion model. Furthermore, they expressed their description as an optimization problem, proving that it is #NP-hard and that the objective function is neither submodular nor supermodular.

Yuting Zhong et al. [56] recently proposed the Maximizing Group Coverage algorithm, which greedily chooses the best node based on evaluating node contributions to groups, ensuring success in estimating the maximum number of activated groups. The experimental results show that the MGC algorithm outperforms the base algorithm, Maximum Coverage, in terms of the number of activated group averages.

Recently, many works have proposed an approach to the influence maximization problem with k -submodular. More specifically, Li et al. [57] propose a new problem, named Keyword-Based Targeted Influence Maximization (KB-TIM), to find a seed set that maximizes the

expected influence on users relevant to a given advertisement. They introduce a sampling technique based on a weighted reverse influence set and achieve an approximation ratio of $(1 - 1/\epsilon - \epsilon)$. However, this method adopts online sampling, so it cannot respond to real-time processing requirements.

In particular, Chen et al. [58] have studied online topic-aware influence maximization (TIM). They find k seeds from a social network such that the topic-aware influence spread of the k seeds is maximized with a theoretical guarantee. Their work utilizes a Maximum Influence Arborescence (MIA) model to approximate the computation of influence spread. Specifically, they proposed a best-effort algorithm with a $(1 - 1/\epsilon)$ -approximation ratio, which estimates an upper bound of the topic-aware influence of each user and utilizes the bound to prune large numbers of users with a slight influence that devise effective techniques to estimate tighter upper bounds. They then propose a faster topic sample-based algorithm with $\epsilon \cdot (1 - 1/e)$ approximation ratio for any $\epsilon \in (0, 1]$, which materializes the influence spread of some topic-distribution samples the materialized information to avoid computing the actual influence of users with minor influences.

Huber and Kolmogorov [59] first introduced the k -submodular function problem, which naturally generalizes submodular and bi-submodular functions as exceptional cases $k = 1$ and $k = 2$, respectively. Their work also proves a Min-Max-Theorem for k -submodular functions and gives a greedy algorithm in polynomial time. Note that, when $k = 1$ this problem of k -submodular is call submodularity. And $k = 2$, this notation is same as bisubmodularity. On this issue, Tang et al. [60] introduced on k -submodular functions in the case with $k = 2$, named bisubmodularity maximization, which gives richer value-of-information and constant-factor approximation algorithms in two applications, such as sensor placement and feature selection. For bisubmodularity with $k = 2$, there are two types of k -submodular function optimization which consist of minimizing [61] [62] [63] [64] and maximization problem [60] [65] [66] [67]. For instance, Thapper et al. [68] used a polynomial time algorithm in the k -submodular function minimizing valued constraint satisfaction problems. However, the problem of k -submodular function maximization needs to be considered more challenging because this kind of problem is $\#NP$ -hard.

Moreover, a wide range of maximizing k -submodular functions has been introduced under more constraints in real-world applications. For k -submodular function with size constraints, Ohsaka et al. [69] proposed approximations algorithms for maximizing monotone k -submodular functions with size constraint under two different cases, i.e, total size constraint and individual size constraint. For the first case, they used to a simple greedy algorithm and random sampling technique to output $\frac{1}{2}$ -approximation algorithms with complexity in $O(kn \log(B) \log(\frac{B}{\delta}))$ for the total size constraint. For the second case, they also used to greedy algorithm to show $\frac{1}{3}$ -approximation algorithms with complexity in $O(knB)$ with $B = \sum_{i=1}^k B_i$ for the individual size constraint. However, their work only considers general monotone for

maximizing k -submodular functions problems under size constraints and does not consider the non-monotone case for maximizing k -submodular functions in optimization problems. Thereafter, Shinsaku Sakaue [70] proposed a greedy algorithm to achieve a $\frac{1}{2}$ -approximate ratio for non negative monotone k -submodular maximization with a matroid constraint. Later, the authors in [71] proposed a multiobjective evolutionary optimization approach for the problem of k -submodular function maximization subject to total size constraint. Their work achieved a $\frac{1}{2}$ -approximation ratio guarantee in polynomial time for the general case which reaches the asymptotically tight bound nearly optimal solution. Recently, many works [72] [73] have considered k -submodular function maximization under noise because the cost to find function f may be expensive and cause some errors. The author [73] proposed two novel streaming algorithms, briefly named DSTREAM and RSTREAM, which have an approximation ratio of $O((1-\epsilon)-2\epsilon B)$ when f is monotone and $O((1-\epsilon)-3\epsilon B)$ when f is non-monotone, respectively. Lately, Zheng et al. [74] presents the problem of approximately k -submodular function maximization subject to size constraint by a simple greedy algorithm with approximation guarantees for different types of size constraints.

For k -submodular function with unconstraint setting, the authors Ward and Zivny [75] considered the problem of maximizing bisubmodular and k -submodular functions in the value oracle model which provided $1/3$ -approximation ratio by a greedy algorithm. Subsequently, Iwata et al. [76] introduced an improved the approximation ratio which achieve $\frac{1}{2}$ -approximation algorithm for maximizing non-monotone case and $\frac{k}{(2k-1)}$ for maximizing monotone k -submodular function case, respectively. And later, the author Hiroki Ohsima [77] improved the approximation ratio to $\frac{k}{(2k-1)}$ -approximation algorithm for monotone k -submodular functions and $\frac{k}{(3k-2)}$ -approximation for non-monotone. Basically, their algorithm based on a variety of greedy algorithm and a different probability distribution.

For k -submodular function with matroid constraint, Chkrabarti and Kale [78] proposed the problem of finding a maximum matching in a graph given including submodular maximization on hypermatchings and intersection of matroids. Their algorithm provided 7.75 -approximation solution for one-pass semi-streaming and $(3+\epsilon)$ -approximation ratio for multi-pass semi-streaming algorithm. Moreover, Sakaue [70] presented a greedy algorithm outputs a $\frac{1}{2}$ -approximate solution for nonnegative monotone k -submodular maximization with a matroid constraint which is improved the $\frac{1}{2}$ -approximation ratio by Iwata et al. [76]. In recent times, the authors in [72] introduced the problem of maximizing monotone submodular functions subject to matroid constraints under framework of differential privacy. Their work proposed two streaming algorithms which has an asymptotically tight approximation ratio.

For k -submodular function with knapsack constraint, [79] considered the problem of maximizing a monotone submodular function subject to a knapsack constraint with streaming setting. They proposed a $(0.363 - \epsilon)$ -approximation algorithm for a single pass through the data and $(0.4 - \epsilon)$ -approximation algorithm for a constant number of passes through the data.

Similarity, authors in [80] presented a new framework for k -submodular maximization problems. By using multilinear extension technique, they proposed $\frac{1}{2}$ -approximation algorithm for the total size constraint and knapsack constraint. Newly, [81] proposed the problem of a non-negative monotone k -submodular function maximization. Their algorithm provided a deterministic $(\frac{1}{2} - \frac{1}{2}e)$ -approximation ratio which is improved by [82] and gave $O(n^4 k^3)$ query complexity.

For k -submodular function with cardinality constraint, Badanidiyuru et al. [83] proposed an efficient streaming algorithm for this issue. Their algorithm provided a constant factor $(\frac{1}{2} - \epsilon)$ -approximation ratio guarantee to the optimal solution and requires only a single pass. By using streaming algorithm approach, the authors in [84] developed an algorithm for the streaming submodular maximization with cardinality constraint under two kinds of noises and has an approximation ratio goes to $\frac{2}{k}$ if $\epsilon \rightarrow 0$ for both multiplicative and additive noises. Recently, the authors Alina Ene and Huy L.Nguyen introduced a new streaming algorithm for maximizing a monotone k -submodular function subject to a per-coordinate cardinality constraint [85]. Their approximation ratio has been improved at least $\frac{1}{4}$ and running time of this proposed algorithm is optimal.

In the reality, there are many application of the problem of k -submodular function maximization are related to budget constraint. The authors in [86] first proposed the problem of maximizing k -submodular function under budget constraint with feasible for k type of topics which is a $\frac{1}{5}(1 - \frac{1}{e})$ -approximation solution with query complexity $O(kn^2)$. The work of authors [87] gave a streaming algorithm with an approximation guarantee of 0.3178 that holds regardless of the minimum budget. The recent work [88] extends the algorithm of (Nguyen and Thai, 2020) to the maximizing k -submodular functions under budget settings by using cost and limit budget of each element in social network. Their work proposed two single pass streaming algorithms with approximation guarantees which provided the approximation ratio of $O(kn/\epsilon)$ query complexity for the special case and $O(kn/\epsilon \log n)$ query complexity for the general case.

Finally, we also propose related works to the maximizing k -submodular influence maximization problem for the k topic and variants of the IM problem. In addition, Ohsaka et al. [69] proposed k -submodular maximization with a size constraint. Their work considers a k -submodular function as a submodular function that consists of k disjoint subsets of the domain. Their work introduces the characterization of k -submodular functions with the condition that the two elements Orthant Submodular and pairwise monotone are satisfied. They give constant-factor approximation algorithms, which use a greedy algorithm with lazy evaluations, to maximize monotone k -submodular functions subject to total size and individual size constraints. The experimental results show that their algorithm runs almost linear and achieves approximately $\frac{1}{2}$ for the total size and $\frac{1}{3}$ for the individual size constraint. In another study, Iwata et al. [76] show that for monotone k -submodular functions with a polynomial-

time $\frac{k}{(2k-1)}$ -approximation algorithm while for any $\epsilon > 0$ a $(\frac{(k+1)}{2k} + \epsilon)$ -approximation algorithm for maximizing monotone k -submodular functions, but there is no require constraint.

In the other study, Nguyen et al. [73] propose two novel streaming algorithms to maximize a noisy k -submodular function subject to size constraints. Their work introduced a k -submodular function with size constraints that still ensures approximation ratios, memory, and query complexity. In particular, their first algorithm is a streaming algorithm taking only one single pass over a vertices set, which works in greedy by putting e into a set i that guarantees that the ϵ -estimate of f is maximized and large enough in comparison with the optimal solution. On the other hand, their second algorithm is also a streaming algorithm that works randomly selected with a probability proportional to its upper bound on the marginal gain.

However, the study of maximizing k -submodular function with size constraint has not been much investigated in the literature and there are more challenges to resolve for the community in the future.

Moreover, the amount of data increases rapidly in the case of online applications in social networks. In some practical cases, the data has increased very quickly that the memory computer cannot store an amount of data in time. Reducing data storage memory and providing guarantee solutions need to be considered. The streaming algorithm is one of the efficient methods used to solve submodular and k -submodular function maximization. To our best knowledge, Ashwinkumar et al. [83] are the first to introduce an efficient streaming algorithm with a constant coefficient $(\frac{1}{2} - \epsilon)$ - approximation guarantee. Similarly, the authors in [73] [84] [89] also introduced the streaming algorithm for submodular function maximization subject to various constraints, i.e., under noise models. Motivated by the above-mentioned results, we develop an efficient streaming algorithm for the k -submodular cover problem with a bi-criteria approximation ratio. Our algorithm desires only a single pass or a few passes over all the data in real-world datasets. It provides theoretical guarantee solutions regarding a number of query complexity, memory usage, and approximation ratio.

2.4 Discussion

In this chapter, we present an overview of the background of IM problem such as Information diffusion models, the approach of IM, and sampling technique. We also introduce relate works for this work in many fields. Based on the background of IM, this dissertation focuses on improving IM algorithms, which guarantee solutions with theoretical bounds in a large social network. These issues will be discussed in the next chapter.

Chapter 3

Multi-Threshold benefit for Influence Maximization

This chapter begins with the definition and motivation of the problem presented in the first section. The second section then shows the sampling technique and the main algorithm, named ESSM. The core of this section is a novel algorithmic framework that can use a solution to a smaller threshold to find a larger threshold. The last section discusses the result of the experiment of the proposed algorithm.

3.1 Motivation

In recent years, there has been a rapid development of the global economy thanks to the contribution of the OSNs, based on the provision of a powerful platform for communication and information dissemination in the field of marketing, media, and advertising, particularly in social networks with billions of users. The strong underpinnings of problems of social influences in OSNs are information diffusion models. The authors in [1] first introduced LT and IC classic models to formulated the IM problem, which aims to select k nodes that may impact the largest number of users a social network. This work has inspired many studies on social influence [4] [5] [10] [12], viral marketing [3] [17] [90] [91] [92], misinformation detection [93] [40], rumors control [94] [95] [96].

In the context of viral marketing for product promotion, hosts (companies) often devise a marketing campaign including the distribution of product samples to selected users and expect that they persuade their friends, friends of friends, etc. The number of people who have been impacted reaches a certain level. IT was inspired by this phenomenon and a slew of research backed it up; it looks for a node set with the smallest size possible so that the number of impacted nodes reaches or surpasses a predetermined threshold γ [97, 90, 98]. The value of γ can determine the scale of of the viral marketing. However, in some realistic

scenarios, there is a distinct cost to persuade a user who promotes a sample product [10, 28]. Besides, each influenced user often offers a different benefit when one is influenced after the marketing process. Customers with significant financial resources, for example, will be able to purchase more things than others. As a result, the existing algorithms for IT problem may offer an inaccurate solution of a marketing purpose. Moreover, the marketing strategies are often adjusted since the market can vary in a short time. Consequently, a particular solution for a benefit is insufficient to be the overall effective solution. This can be overcome by finding solutions for multiple thresholds and selecting the best one that suits their budget and current market.

For instance, assume that a company wants to come up with a strategy that can influence customers on an OSN. Nonetheless, or due to budget fluctuations or the instability of the market, they may consider strategies of spreading with the different number of influenced customers such as 1000, 2000, 3000, 5000, etc. In this case, the company wants to find solutions, where the benefit function of each is above the corresponding threshold and then that company can select a solution with a reasonable cost so as to execute its marketing plan well.

Our goal in this study is to develop an answer to a novel MBT problem, which is expressed as follows. For a social network $G = (V, E)$ given a set of k benefit thresholds $T = \{T_1, T_2, \dots, T_k\}$, each user u has a distinct cost price $c(u) > 0$. The issue is to seek for the various seed sets $\{S_1, S_2, \dots, S_k\}$, in which each S_i has the cheapest total cost $c(S_i)$ by a result of each seed set's earned benefit S_i , characterized by $\mathbb{B}(S_i)$, and is at least T_i for $i = 1 \dots, k$. There are two main challenges for solving MBT problem. First ones are to find MBT as #NP-Hard and to calculate the benefit function #P-Hard. Secondly, finding numerous seed sets for multiple thresholds needs more time and memory than other information propagation challenges, as well as the IT problem. It is necessary to run the existing algorithms for a single threshold k times to prove it is costly and, hence, not applicable to large networks. To overcome the challenges, in this thesis, we propose a highly efficient algorithm to solve the problem. This not only guarantees a solution but also produces good results in practice. This work revised and extended the our work [99] by providing all the proofs more detail and experiment evaluation.

3.2 Problem definition

We formally introduce our studied the MBT problem, as follows:

Definition 2 (MBT) *Given a graph $G = (V, E)$ under the IC model and the set of benefit thresholds $T = \{T_1, T_2, \dots, T_k\}$. For each $T_i \in T$, the problem is required to find $S_i \in V$ with smallest cost $c(S_i)$ so that $\mathbb{B}(S_i) \geq T_i$.*

In the case when $b(u) = 1, \forall u \in V$, the benefit function $\mathbb{B}(\cdot)$ becomes the influence spread function [1]. Ref. [17] showed that it was $\#P$ -hard to compute the number of influence nodes (influence spread function) exactly, so calculating $\mathbb{B}(\cdot)$ was also $\#P$ -hard. Besides, the IT problem [90, 45, 98], a special case of MBT problem with $b(u) = c(u) = 1, \forall u \in V$ and $k = 1$, is NP-hard, which implies that MBT is also $\#NP$ -hard.

3.3 Multi-threshold for Influence Maximization problem

In this section, the Efficient Sampling for Selecting Multiple seed sets (ESSM), an efficient algorithm for MBT problem with theoretical guarantee, is introduced. Our novel technique is to develop a method that combines two following ideas: (1) finds the candidate seed set for each threshold via the benefit sampling; (2) uses the seed set with a smaller threshold for finding the seed sets with bigger ones, which can improve the running time as well as memory usage. Moreover, the sampling technique with martingale theory is in use to estimate the benefit function effectively.

3.3.1 Benefit Sampling

We first recap the concept of *Benefit Sample* (BS) in [10] to estimate the $\mathbb{B}(\cdot)$.

Definition 3 (Benefit Sample) *A BS is generated from $G = (V, E)$ under the IC model by following steps: (1) Choose a source node u with probability $\frac{b(u)}{\Gamma}$, (2) create a sample graph g from G , and (3) return R_j as the set of nodes that can reach node u in g .*

The Algorithm 1 in [10] can be used to generate a BS for IC model.

Algorithm 1: An algorithm for generating a BS under the IC model

Input: Graph $G = (V, E)$ under IC model
Output: A BS set R_j

- 1: Choose a source node u with probability $\frac{b(u)}{\Gamma}$
- 2: Initialize a queue $Q = \{u\}$ and $R_j = \{u\}$
- 3: **while** Q is not empty **do**
- 4: $v \leftarrow Q.pop()$
- 5: **for** $u \in N_{in}(v) \setminus (R_j \cup Q)$ **do**
- 6: With probability $p(u, v)$ do: $Q.push(u)$, $R_j \leftarrow R_j \cup \{u\}$;
- 7: **end for**
- 8: **end while**
- 9: **return** R_j

Given \mathcal{R} is a collection of BSes, a seed set S , we define a random variable $X_j(S)$ as follows:

$$X_j(S) = \begin{cases} 1, & \text{If } R_j \cap S \neq \emptyset \\ 0, & \text{Otherwise} \end{cases} \quad (3.1)$$

We can estimate the benefit function $\mathbb{B}(S)$ by the following Lemma in [10].

Lemma 1 (Lemma 2, [10]) *For any set of nodes $S \subseteq V$, we have: $\mathbb{B}(S) = \Gamma \cdot \mathbb{E}[X_j(S)]$*

The function $\mathbb{B}(\cdot)$ is *monotone* and *submodular* [10], i.e., for any $S \subseteq T \subseteq V$, and $v \notin T$, we have

$$\mathbb{B}(T) \geq \mathbb{B}(S) \quad (3.2)$$

$$\mathbb{B}(S + \{v\}) - \mathbb{B}(S) \geq \mathbb{B}(T + \{v\}) - \mathbb{B}(T) \quad (3.3)$$

We can calculate an estimation $\hat{\mathbb{B}}(S)$ of $\mathbb{B}(S)$ via a collection \mathcal{R} of BSes as follows:

$$\hat{\mathbb{B}}(S) = \frac{\Gamma}{|\mathcal{R}|} \sum_{R_j \in \mathcal{R}} X_j(S) \quad (3.4)$$

It can be seen that $X_j(S) \in [0, 1]$. We define a random variable $Y_i = \sum_{j=1}^i (X_j(S) - \mu)$, $\forall i \geq 1$, where $\mu = \mathbb{E}[X_j]$ and a sequence random variables Y_1, Y_2, \dots, Y_k , we have

$$\mathbb{E}[Y_i | Y_1, \dots, Y_{j-1}] = \mathbb{E}[Y_{i-1}] + \mathbb{E}[Y_i(S) - \mu] = \mathbb{E}[Y_{i-1}]$$

Therefore, Y_1, Y_2, \dots, Y_k are a form of martingale [100]. Thus, we have the following Lemma [100].

Lemma 2 ([100]) *Given a collection \mathcal{R} with $T = |\mathcal{R}|$ and $\lambda > 0$, we have*

$$\Pr \left[\sum_{j=1}^T X_j(S) - T \cdot \mu \geq \lambda \right] \leq \exp \left\{ -\frac{\lambda^2}{2\lambda\frac{2}{3} + \mu T} \right\} \quad (3.5)$$

$$\Pr \left[\sum_{j=1}^T X_j(S) - T \cdot \mu \leq -\lambda \right] \leq \exp \left\{ -\frac{\lambda^2}{2\mu T} \right\} \quad (3.6)$$

Let $\lambda = \epsilon T \mu$ in Lemma 2, we obtain

$$\Pr[\hat{\mathbb{B}}(S) \geq (1 + \epsilon)\mathbb{B}(S)] \leq \exp \left\{ -\frac{\epsilon^2 \mu T}{2 + \frac{2}{3}\epsilon} \right\} \quad (3.7)$$

$$\Pr[\hat{\mathbb{B}}(S) \leq (1 - \epsilon)\mathbb{B}(S)] \leq \exp \left\{ -\frac{\epsilon^2 \mu T}{2} \right\} \quad (3.8)$$

If the number of BSs is at least $T \geq (2 + \frac{2}{3})\frac{1}{\mu} \frac{1}{\epsilon^2} \ln(\frac{1}{\delta})$ for $\delta \in (0, 1)$, $\hat{\mathbb{B}}_{\mathcal{R}}(S)$ is an (ϵ, δ) -approximation of $\mathbb{B}(S)$, i.e.,

$$\Pr[(1 - \epsilon)\mathbb{B}(S) \leq \hat{\mathbb{B}}(S) \leq (1 + \epsilon)\mathbb{B}(S)] \geq 1 - \delta \quad (3.9)$$

The characteristics of the martingale sequence play an important role in devising our algorithm in the next subsection.

3.3.2 ESSM Algorithm

Our proposed algorithm is now described. On a high level, our algorithm combines two methods: (1) We provide a (δ, ϵ) -approximation of the benefit function via martingale theory. (2) In each iteration, we propose the algorithmic framework that finds some candidate seed sets for a threshold and then choose the final seed set, which guarantees the solution quality by checking static evidence. (3) We reuse the seed set for smaller threshold for finding the seed sets with the larger threshold. Our proposed algorithm is presented in Algorithm 2.

At the beginning of the algorithm, it generates collection \mathcal{R}_0 that contains $\frac{(2+\frac{2}{3})\Gamma}{\epsilon^2(T_i - \epsilon T_i)}(\ln n + \ln(1/\delta))$ BSs by using Algorithm 1 and initiates a seed set S_1 as empty.

At each iteration i of **first loop** (line 3–18), it finds the seed set with respect to threshold T_i . Denote $f(S_i) = \min(\hat{\mathbb{B}}(S_i), T_i - \epsilon T_i - \epsilon)$. At each iteration of the **second loop** (line 7–18), the algorithm finds a seed S_i , by iteratively selecting a node u with maximum marginal of the estimation function f as per its cost, i.e., $\frac{(f(S_i \cup \{u\}) - f(S_i))}{c(v)}$ and (2) checking the condition of the number of samples (line 12). If the number of samples is sufficient to give an (δ, ϵ) -approximation (by Lemma 3), the algorithm moves into next iterations and keeps current seed set S_i ; otherwise, the algorithm generates more samples (line 13) so that the number of samples is $N(i, j)$ and adds them into R_i . In this case, the seed set S_i is suitable for new collection R_i . The second loop terminates when it satisfies the condition $\hat{\mathbb{B}}(S_i) \geq T_i - \epsilon T_i - \epsilon$. Next, the algorithm reuses the current samples and seed set to find the seed set for larger threshold (lines 4–5) by using similar steps with previous iteration.

The theoretical bounds of the algorithm are now analyzed. Firstly, the satisfactory number of BSes is provided to estimate $\mathbb{B}(\cdot)$ is shown in Lemma 3.

Algorithm 2: ESSM algorithm

Input: A graph $G = (V, E)$, $T = \{T_1, \dots, T_k\}$, $\epsilon, \delta \in (0, 1)$

Output: S_1, S_2, \dots, S_k

```
1: Generate  $\mathcal{R}_0$  containing  $\frac{(2+\frac{2}{3})\Gamma}{\epsilon^2(T_i-\epsilon T_i)}(\ln n + \ln(1/\delta))$  BSs by using Algorithm 1
2:  $S_0 \leftarrow \emptyset$ 
3: for  $i = 1$  to  $k$  do
4:    $\mathcal{R}_i \leftarrow \mathcal{R}_{i-1}$ 
5:    $S_i \leftarrow S_{i-1}$ 
6:   Calculate  $\hat{\mathbb{B}}(S_i)$  by eq. (3.4)
7:   while  $\hat{\mathbb{B}}(S_i) < T_i - \epsilon T_i - \epsilon$  do
8:      $u \leftarrow \arg \max_{v \in V \setminus S_i} \frac{\min(\hat{\mathbb{B}}(S_i \cup v), T_i - \epsilon T_i - \epsilon) - \hat{\mathbb{B}}(S_i)}{c(v)}$ 
9:      $S_i \leftarrow S_i \cup \{u\}$ 
10:     $j \leftarrow |S_i|$ 
11:     $N(i, j) \leftarrow \frac{(2+\frac{2}{3})\Gamma}{\epsilon^2(T_i-\epsilon T_i)} \ln(\binom{n}{j}/\delta)$ 
12:    if  $|\mathcal{R}_i| < N(i, j)$  then
13:      Generate more  $N(i, j) - |\mathcal{R}_i|$  BSs and add them into  $\mathcal{R}_i$ 
14:       $N \leftarrow N(i, j)$ 
15:       $S_i \leftarrow \emptyset$ 
16:    end if
17:  end while
18: end for
19: return  $S_1, S_2, \dots, S_k$ 
```

Lemma 3 If $|\mathcal{R}| \geq \frac{(2+\frac{2}{3})\Gamma}{\epsilon^2(T_i-\epsilon T_i)}(\ln n + \ln \frac{1}{\delta})$ then $\Pr[\hat{\mathbb{B}}(S_i^*) \geq T_i - T_i\epsilon] \geq 1 - \delta$

Proof Denote $\mu = \mathbb{B}(S_i^*)/\Gamma$, $\hat{\mu} = \hat{\mathbb{B}}(S_i^*)/\Gamma$, we have

$$\begin{aligned} \Pr[\hat{\mathbb{B}}(S_i^*) \leq T_i - T_i\epsilon] &\leq \Pr[\hat{\mathbb{B}}(S_i^*) \leq (1 - \epsilon)\mathbb{B}(S_i^*)] \\ &= \Pr[\hat{\mu} \leq (1 - \epsilon)\mu] \quad (\text{By applying (3.8)}) \\ &\leq \exp\left(\frac{-\epsilon^2|\mathcal{R}|\mu}{2}\right) \\ &\leq \exp\left(\frac{-\epsilon^2|\mathcal{R}|\hat{\mu}}{2(1 - \epsilon)}\right) \quad (\text{Due to } \mu \geq \hat{\mu}/(1 - \epsilon)) \\ &\leq \exp\left(-\frac{(2 + \frac{2}{3})\hat{\mathbb{B}}(S_i^*)}{2(1 - \epsilon)(T_i - \epsilon T_i)} \ln \frac{1}{\delta}\right) \leq \delta \end{aligned}$$

which implies the proof. ■

The theoretical guarantee of Algorithm 2 is stated as follows.

Theorem 1 For any inputs $\epsilon, \delta \in (0, 1)$, the Algorithm 2 returns a set of seed sets $S = \{S_1, S_2, \dots, S_k\}$ satisfying

$$(a) \Pr[c(S_i) \leq (1 + \ln \frac{T_i - \epsilon T_i}{\epsilon})c(S_i^*)] \geq 1 - \delta/n.$$

$$(b) \Pr(\mathbb{B}(S_i) \geq T_i \cdot \frac{1-\epsilon}{1+\epsilon} - \epsilon) \geq 1 - \delta.$$

Proof At any i -th iterator of the first loop (line 3 to 19) in Algorithm 2, denote $S_i = S_i^t = \{s_i^1, s_i^2, \dots, s_i^t\}$ as the solution of algorithm with respect to the threshold T_i , and $P_i = \{v_1^i, v_2^i, \dots, v_l^i\}$ as a set of nodes with minimum cost satisfying $\hat{\mathbb{B}}(P_i) \geq T_i - \epsilon T_i$ and $C_i = c(P_i)$. Due to the checking condition in line 12, the number of BSeS at the end of iteration i obtains at least

$$N_{min}^i = \frac{(2 + \frac{2}{3}\epsilon)\Gamma}{\epsilon^2(T_i - \epsilon T_i)} \ln\left(\binom{n}{|S_i|}/\delta\right) \quad (3.10)$$

and obtains at most,

$$N_{max}^i = \max_{j:1 \dots |S_i|} \frac{(2 + \frac{2}{3}\epsilon)\Gamma}{\epsilon^2(T_i - \epsilon T_i)} \ln\left(\binom{n}{j}/\delta\right) \quad (3.11)$$

Prove (a) As $\hat{\mathbb{B}}(\cdot)$ is submodular, we have

$$\begin{aligned} T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^{t-1}) &\leq \hat{\mathbb{B}}(P_i) - \hat{\mathbb{B}}(S_i^{t-1}) \\ &\leq \hat{\mathbb{B}}(P_i \cup S_i^{t-1}) - \hat{\mathbb{B}}(S_i^{t-1}) \\ &\leq \sum_{v \in P_i \setminus S_i^{t-1}} (\hat{\mathbb{B}}(S_i^{t-1} \cup \{v\}) - \hat{\mathbb{B}}(S_i^{t-1})) \\ &\leq \frac{C_i}{c(S_i^{t-1})} \sum_{v \in P_i \setminus S_i^{t-1}} (\hat{\mathbb{B}}(S_i^{t-1} \cup \{v\}) - \hat{\mathbb{B}}(S_i^{t-1})) \end{aligned}$$

For any positive numbers a_1, \dots, a_l and b_1, \dots, b_l . According to [101], we have

$$\min_{i=1 \dots l} \frac{a_i}{b_i} \leq \frac{\sum_{i=1}^l a_i}{\sum_{i=1}^l b_i} \leq \max_{i=1 \dots l} \frac{a_i}{b_i} \quad (3.12)$$

Applying the above inequality, we obtain

$$T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^t) \leq \frac{C_i}{c(s_i^t)} (\hat{\mathbb{B}}(S_i^t) - \hat{\mathbb{B}}(S_i^{t-1})) \quad (3.13)$$

$$\leq (1 - \frac{c(s_i^t)}{C_i}) (T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^{t-1})) \quad (3.14)$$

$$\leq e^{-\frac{c(s_i^t)}{C_i}} (T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^{t-1})) \quad (3.15)$$

The (3.15) condition must satisfy $x + 1 \leq e^x$, for any $x > 0$. Therefore,

$$T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^t) \leq e^{-\frac{1}{C_i} \sum_{j=1}^t c(s_j^t)} (T_i - \epsilon T_i) \quad (3.16)$$

$$= e^{-\frac{1}{C_i} c(S_i^t)} (T_i - \epsilon T_i) \quad (3.17)$$

By the definition of S_i^t and because S_i satisfies the condition in line 7, we have $\hat{\mathbb{B}}(S_i^{t-1}) < T_i - \epsilon T_i - \epsilon$ and $\hat{\mathbb{B}}(S_i^t) \geq T_i - \epsilon T_i - \epsilon$. Combining with (3.17), we have

$$\begin{aligned} (T_i - \epsilon T_i) e^{-\frac{1}{C_i} c(S_i^{t-1})} &\geq T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^{t-1}) \\ &> T_i - \epsilon T_i - (T_i - \epsilon T_i - \epsilon) = \epsilon \end{aligned}$$

implying that $c(S_i^{t-1}) < C_i \ln \frac{T_i - \epsilon T_i}{\epsilon}$. On the other hand, from (3.15), we obtain

$$c(s_i^t) \leq C_i \ln \frac{T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^{t-1})}{T_i - \epsilon T_i - \hat{\mathbb{B}}(S_i^t)} \leq 1 \quad (3.18)$$

Thus, $c(S_i^t) = c(S_i^{t-1}) + c(s_i^t) \leq C_i(1 + \ln(\frac{T_i - \epsilon T_i}{\epsilon}))$, where S_i is the candidate solution for threshold T_i . After i -th iteration of the first loop, $|\mathcal{R}_i| = N(i, j) = \frac{(2+\frac{2}{3})\Gamma}{\epsilon^2(T_i - \epsilon T_i)} \ln(\binom{n}{j}/\delta)$. By applying Lemma 3, after iterator i , we have $\Pr[\mathbb{B}(S_i^*) \geq T_i - \epsilon T_i] \geq 1 - \delta/\binom{n}{j}$. Combining with the definition of P_i , the following events happen with a probability of at least $1 - \delta/\binom{n}{j} \geq 1 - \delta/n$:

$$c(S_i) \leq C_i(1 + \ln(\frac{T_i - \epsilon T_i}{\epsilon})) \quad (3.19)$$

$$\leq c(S_i^*)(1 + \ln(\frac{T_i - \epsilon T_i}{\epsilon})) \quad (3.20)$$

Prove (b) The i -th iteration of the first loop ends when $\hat{\mathbb{B}}(S_i) \geq T_i - T_i\epsilon - \epsilon$, we obtain

$$\begin{aligned} \Pr\left(\mathbb{B}(S_i) \leq T_i \frac{1-\epsilon}{1+\epsilon} - \epsilon\right) &\leq \Pr\left(\mathbb{B}(S_i) \leq \frac{T_i - T_i\epsilon - \epsilon}{1+\epsilon}\right) \\ &\leq \Pr\left(\mathbb{B}(S_i) \leq \frac{\hat{\mathbb{B}}(S_i)}{1+\epsilon}\right) \\ &\leq e^{\frac{-\epsilon^2 |\mathcal{R}_i| \hat{\mathbb{B}}(S_i)}{2\Gamma(1+\epsilon)}} \quad (\text{By applying (3.8)}) \\ &\leq e^{\frac{-\ln(\binom{n}{j}/\delta)}{1+\epsilon}} \\ &\leq 1 - \delta/\binom{n}{j} \end{aligned}$$

Since $|S_i| = j$ there are at most $\binom{n}{j}$ possible solutions S_i . By applying the union

bound of the probability of events, we have $\Pr\left(\forall S_i, \mathbb{B}(S_i) \leq T_i \cdot \frac{1-\epsilon}{1+\epsilon} - \epsilon\right) \leq \delta$. Hence, $\Pr\left(\mathbb{B}(S_i) \geq T_i \cdot \frac{1-\epsilon}{1+\epsilon} - \epsilon\right) \geq 1 - \delta$. The proof is completed. ■

Theorem 2 (Number of required BSes) *For any $\epsilon, \delta \in (0, 1)$, the sample complexity of ESSM is $O(\epsilon^{-2}n \ln(\binom{n}{i_{max}}/\delta))$, where $i_{max} = \arg \max_{i=1 \dots |S_k|} \ln(\binom{n}{i})$.*

Proof The number of BSes for finding seed set S_i is at most N_{max}^i . The algorithm reuses the set of BSes for current seed set for next iteration, so the number of BSes generated by the algorithm is at most N_{max}^k . On the other hand, $\Gamma = \sum_{u \in V} b(u) \leq b_{max}n = O(n)$. Therefore, the number of samples used in the algorithm is

$$\frac{(2 + \frac{2}{3}\epsilon)\Gamma}{\epsilon^2(T_1 - \epsilon T_1)} \ln\left(\binom{n}{i_{max}}/\delta\right) = O(\epsilon^{-2}n \ln(\binom{n}{i_{max}}/\delta))$$

which completes the proof. ■

Denote $M, (M \leq n)$ is the expected running time for generating one BS, and j_{max} is the largest number of iterations of selecting a seed set. The time complexity of the algorithm is $O(\epsilon^{-2}nkj_{max}M \ln(\binom{n}{N_{max}^k}/\delta))$.

3.4 Experiment

For computing the transmission probability in IC model, the conventional computation as in [1, 4, 5, 10] is followed and the transmission probability is calculated as $p(u, v) = \frac{1}{|N_{in}(v)|}$. We set $c(u) = \frac{n \cdot N_{out}(u)}{\sum_{v \in V} N_{out}(v)}$ and randomly choose 20% of nodes in each network and set the benefit to 1, the rest assign to 0 as in [10]. Finally, $\epsilon = 0.1$ and $\delta = 1/n$ are set as a default setting [10, 4, 5] in all the experiments. We utilize a Linux computer with $2 \times$ Intel(R) Xeon(R) CPU E5-2630 v4 processors running at 2.20 GHz and used 64 GB DDR4 RAM performing at 2400 MHz. Our algorithms are developed in C/C++ using the g++11 compiler.

For a comprehensive experiment, six networks are selected for information propagation problems [1, 10, 12, 31, 4, 5] of different sizes. The description of used datasets is presented in Table 3.1.

- Gnutella [102] represents Gnutella peer-to-peer file sharing network in August 2002. In this network, 20,777 edges among 6301 nodes show connections among hosts in the Gnutella network topology.
- Email-Enron [103] network covers all the email communication within a dataset of around half a million emails. These originally public data were posted on the web, by the Federal Energy Regulatory Commission during its investigation. Nodes of the network are email addresses and if an address \mathbf{i} has sent at least one email to address

j , the graph contains an undirected edge. Note that non-Enron email addresses act as sinks and sources in the network as their communication with the Enron email addresses is only under observation. The Enron email data were originally released by William Cohen at CMU.

- Net-Hept [104] and Net-Phy [12] are collaborative networks from the “high-energy physics theory” section and “physics” section, in which the nodes represent the authors and undirected edges represent papers written by the same authors.
- Amazon [105] was collected in 2 March 2003 by crawling the Amazon website. It is based on customers who bought an item and also bought features of the Amazon website. If a product i is frequently copurchased with product j , the graph contains a directed edge from i to j .
- DBLP computer science bibliography [106] provides a comprehensive list of research papers in computer science. If two authors publish at least one publication together, they establish a co-authorship network.

Since IT [33] and CTVM [10] are the problems most closely related to MBT problem, ESSM is compared with their algorithms with some modifications in our experiment. In addition, the DEGREE algorithm, a popular baseline algorithm for information propagation problems [1, 4, 17, 12], is in use. Compared algorithms are listed below.

- BCT is an algorithm for CTVM problem [10]. BCT is used by comparison due to the similarity between the BCT and CTVM problem by considering the costs and benefits of the nodes. However, due to the differences between MBT and CTVM, BCT is adapted with some modifications as follows: For each threshold T_i , we use a binary search on the cost from range $[0, \sum_u c(u)]$ until the reached benefit function falls in $[T_i(1 - \epsilon), T_i]$, where $\epsilon = 0.1$ and returns the seed set with minimum cost.
- IT is a greedy algorithm for the Influence Threshold problem in [33]. In order to adapt IT algorithm for MBT problem, the Monte Carlo simulation is used to estimate benefit function with 10,000 time simulations as in [1, 12].
- DEGREE is one of common baseline algorithms for influence problem [10, 27, 1], which select the highest degree of nodes until the benefit of the selection set exceeds thresholds.

The ESSM algorithm was experimented and compared other algorithms BCT, DEGREE, and IT on the datasets Net-Hept 15K nodes and Net-Phy 37K nodes, and the results are shown below (**Figure 3.1**).

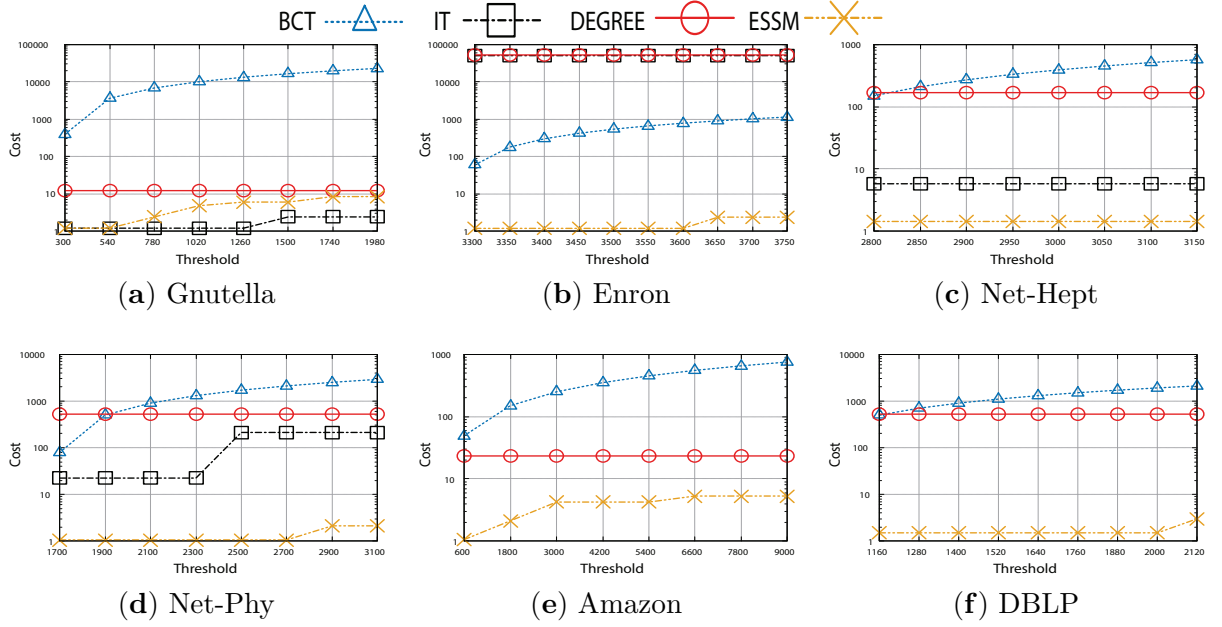
Table 3.1: MBT experimental datasets

Dataset	#Nodes	#Edges	Avg. Degree	Source
Gnutella	6301	20,777	3.3	[102]
Enron	36,692	183,831	5.0	[103]
Net-Hept	15,233	58,891	5.5	[104]
Net-Phy	37,154	231,584	13.4	[12]
Amazon	262,111	1,234,877	9.4	[105]
DBLP	317,080	1,049,866	6.6	[106]

Figure 3.1 showed the costs of seed sets returned by algorithms in which the smaller one was better. Our algorithm ESSM outperformed other algorithms by a large gap in most datasets except the Gnutella network. Particularly, ESSM returned the seed sets whose costs are 1875 to 116,000 times less than that of other algorithms. The results also confirmed that our framework algorithm was more efficient than the others. The IT algorithm only produced good results on the Gnutella dataset and produced worse results than ESSM did on the rest. However, it delivered better results than the rest algorithms did, because the algorithm always finds important seed nodes with low and rational cost as our algorithm do. With large datasets (Amazon and DBLP), IT did not finish within the time limit. This showed that the Monte Carlo method was not suitable for large networks due to its high complexity. DEGREE algorithm selected the highest out-degree of nodes to prioritize as seed nodes, so the highest degree value affected the cost of computing formula, leading to considerable increase in cost, even when the variety of found seed nodes were small. Especially in the Email-Enron dataset, at the first threshold T_i , where a seed node was loaded with the highest out-degree, the DEGREE algorithm resulted in the high cost value, even higher than that of the BCT algorithm; although, BCT was also based on the use of BS samples but produced worse results because it used binary search, which could give much larger results than the optimal solution.

The running times of algorithms were demonstrated in Figure 3.2. ESSM was significantly faster than the others on all datasets. ESSM algorithm was 6900 to 127,710 times faster and 39 to 2120 times faster than IT and BCT, respectively. The running time of IT was the longest and could not finish within time limit for Amazon and DBLP networks. This was caused by the long time IT spent on accessing Monte Carlo simulation to estimate the benefit function. The running times of algorithms are shown in Figure 3.2. ESSM was significantly faster than the others on all datasets. ESSM algorithm is 6900 to 127,710 times faster and 39 to 2120 times faster than IT and BCT, respectively. The running time of IT was the longest and it could not finish within time limit for Amazon and DBLP networks. This resulted from IT spending a long time on calling Monte Carlo simulation to estimate benefit function. BCT was significantly faster than IT even though it used many loops for binary search for the reason that the BCT used BS samples to estimate the benefit function instead of Monte

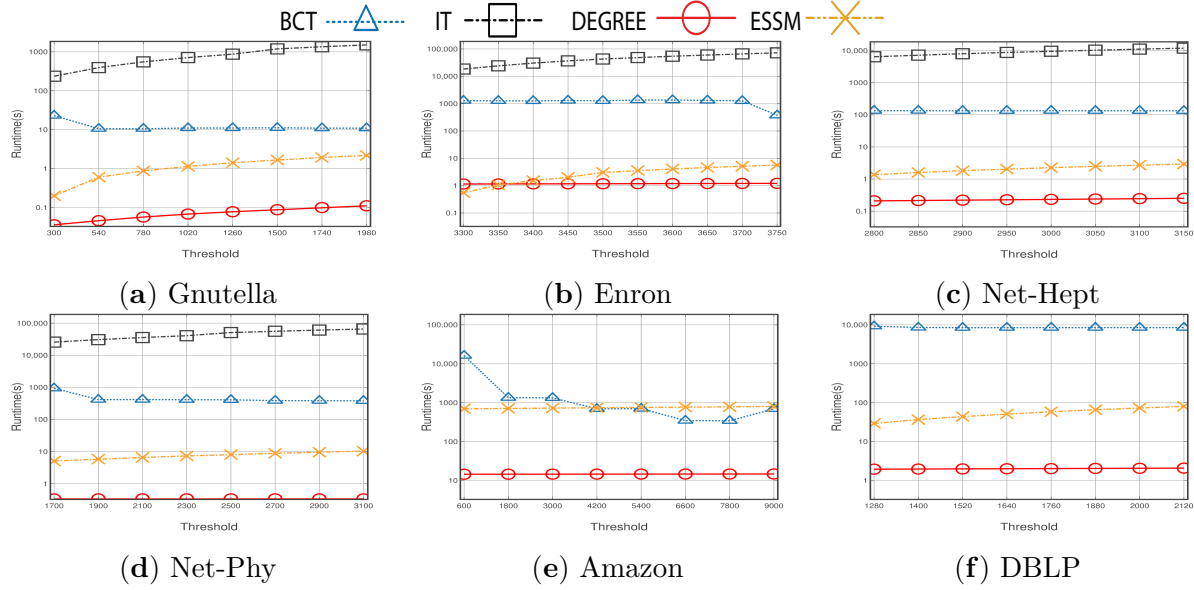
Figure 3.1: Comparison about the Costs of seed sets between ESSM and other algorithms within threshold T_i from 300 to 9,000



Carlo simulation method. However, BCT was significantly slower than our algorithm because it did not have a mechanism for reusing the seed set in finding other seed sets with a larger benefit threshold. The larger number of vertices of the datasets, the more time it took BCT to find a solution. The above results were consistent with our assessment that the seed selection strategy in the reuse of solution could shorten the running time of the algorithm. The above results were consistent with our assessment that the seed selection strategy with the reuse seed sets in our algorithm could shorten the time to find the solution. DEGREE algorithm was also based on the use of a Monte-Carlo-like IT algorithm. Nevertheless, choosing seed nodes was easily dependent on the existing seed set without predicting the next seed nodes. As a consequence, DEGREE ran for a few seconds and was 4 to 54 times faster than our algorithm.

The memory usage of algorithms are illustrated in Table 3.2. The memory of our ESSM algorithm was not the lowest in small and medium datasets, depending on the characteristics of the data, but the difference was not fairly significant. In the remaining medium and large datasets, the ESSM algorithm clearly offered its advantages with a reduction in memory usage of more than 20,000 times compared with the BCT algorithm in the DBLP dataset. The ESSM algorithm will be more likely to be used on larger datasets while the BCT and IT algorithms will be less likely. The BCT algorithm does not inherit the sample set across multiple thresholds, such as regenerating independent time-consumption and memory usage for sample sets at each threshold. With a lower threshold, the formula for calculation requires the large number of samples. Whereas the threshold increases, the total required sample

Figure 3.2: Comparison of Running time between ESSM and other algorithms within threshold T_i from 300 to 9,000



set decreases. As a result, the memory usage must decrease and the threshold value must increase. Moreover, the BCT's sampling algorithm does not guarantee the consistency of the number of samples at a certain threshold, leading to an unusual variation in memory usage among these closing thresholds T_i , which was clearly displayed in small datasets using the close thresholds in the experiments as Gnutella, Net-Hept. During the experiment, the IT algorithm always consumed the highest running time among the algorithms, caused by the use of the classical Monte Carlo sampling algorithm, which consumed the memory usage as well as the run-times. Two large datasets as Amazon and DBLP could not experiment with the IT algorithm partly because during the sampling process, the algorithm overloaded the memory usage. This exhibited the disadvantage of IT algorithm compared with other algorithms. On the contrary, IT used less memory than BCT and ESSM did in some cases because of its no need of storing BS samples such as the other two mentioned algorithms. Finally, similar to IT, DEGREE used the least amount of memory because of its simplicity with no inheritance in building solutions.

In comparison to the preceding solutions' optimized inheritance loops, the ESSM method is expected to have a Running duration nearly 100 to 20000 times lower than the BCT algorithm, which is at more optimal intervals than the AT algorithm. In terms of total cost, the ESSM algorithm gives a much better solution than the other algorithms.

3.5 Discussion

In this chapter, we study an efficient algorithm for Multiple Benefit Threshold, named MBT problem. We also introduce the sampling technique, which can find larger solutions based on small solutions. In addition, we point out good performance by our proposed algorithm compared to the others in extensive experiments with social networks.

Table 3.2: Memory usage of compared algorithms ESSM and other algorithms within threshold T_i from 300 to 9,000(MB)

(a)

Dataset	Threshold	Algorithm			
		BCT	IT	DEGREE	ESSM
Gnutella	$T_i = 300$	0.758	0.77	0.855	1.02
	$T_i = 540$	0.805	0.75	0.852	1.02
	$T_i = 780$	0.805	0.758	0.719	1.02
	$T_i = 1020$	0.758	0.789	0.75	1.02
	$T_i = 1260$	0.758	0.789	0.723	1.02
	$T_i = 1500$	0.809	0.816	0.723	1.02
	$T_i = 1740$	0.824	0.855	0.785	1.02
	$T_i = 1980$	0.824	0.813	0.746	1.02
Email-Enron	$T_i = 3300$	4859.98	1.051	0.746	0.813
	$T_i = 3350$	4874.96	1.051	0.855	0.809
	$T_i = 3400$	4841.89	1.051	0.77	0.715
	$T_i = 3450$	4863.27	1.051	0.75	0.855
	$T_i = 3500$	4839.59	2.328	0.809	0.75
	$T_i = 3550$	4856.99	2.582	0.711	0.816
	$T_i = 3600$	4858.67	2.582	0.746	0.7
	$T_i = 3650$	4835.6	2.582	0.855	0.715
Net-Hept	$T_i = 2800$	0.723	0.711	0.711	0.77
	$T_i = 2850$	0.723	0.742	0.855	0.77
	$T_i = 2900$	0.723	0.75	0.754	0.77
	$T_i = 2950$	0.77	0.75	0.855	0.77
	$T_i = 3000$	0.805	0.77	0.754	0.77
	$T_i = 3050$	0.746	0.809	0.809	0.77
	$T_i = 3100$	0.75	0.715	0.75	0.77
	$T_i = 3150$	0.75	0.754	0.809	0.77

(b)

Dataset	Threshold	Algorithm			
		BCT	IT	DEGREE	ESSM
Net-Phy	$T_i = 1700$	2,800.25	0.805	20.66	1.117
	$T_i = 1900$	1,444.21	0.723	20.66	1.117
	$T_i = 2100$	1,446.43	0.82	20.66	1.117
	$T_i = 2300$	1,442.56	0.82	20.66	1.117
	$T_i = 2500$	1,434.55	0.867	20.66	1.117
	$T_i = 2700$	1,429.17	0.75	20.66	1.117
	$T_i = 2900$	1,426.53	0.758	20.66	1.117
	$T_i = 3100$	1,437.99	0.793	20.66	1.117
Amazon	$T_i = 600$	0.195	N/A	0.723	12.453
	$T_i = 1800$	0.742	N/A	0.789	12.453
	$T_i = 3000$	0.742	N/A	0.809	12.512
	$T_i = 4200$	0.746	N/A	0.719	12.512
	$T_i = 5400$	0.715	N/A	0.813	12.512
	$T_i = 6600$	0.715	N/A	0.746	12.512
	$T_i = 7800$	0.805	N/A	0.758	12.512
	$T_i = 9000$	0.715	N/A	0.742	12.512
DBLP	$T_i = 1280$	26,316.8	N/A	0.711	19.121
	$T_i = 1400$	41,369.6	N/A	0.715	19.227
	$T_i = 1520$	26,009.6	N/A	0.813	19.227
	$T_i = 1640$	24,883.2	N/A	0.816	19.227
	$T_i = 1760$	24,883.2	N/A	0.719	19.227
	$T_i = 1880$	24,883.2	N/A	0.711	19.227
	$T_i = 2000$	24,883.2	N/A	0.711	19.227
	$T_i = 2120$	24,883.2	N/A	0.754	19.227

Chapter 4

Groups Influence Maximization in Large Network

This chapter starts with the definition and motivation of the Influence Maximization of groups or communities problem presented in the first section. The second section then shows two main algorithms, named Threshold Benefit for Groups Influence (TGI problem) and Groups Influence Maximization with Minimum Cost (GIM problem). Then, problem definitions show in the next section. Next, the main of this section is a novel algorithmic framework and group sampling technique for Group Influence Maximization under constraint setting such as benefit and cost. Finally, the last section discusses the result of the experiment of the two proposed algorithms.

4.1 Motivation

4.1.1 Threshold Benefit for Groups Influence in OSNs

IM is a popular concern in recent times, especially in social networks with millions of users such as Youtube, Facebook, Twitter, or the other OSNs like LinkedIn, Tumblr, etc. Users participating in social networks can connect with others easily, not only they are not only exchange and share information, but also update information quickly. Companies, organizations, and businesses have used to OSNs as a communication channel to advertise products, such as new products, online shopping, online training, spreading opinions or supply of human resources, etc. OSNs has become an important platform in the field of marketing and advertising by leveraging the effectiveness of word-of-mouth. The primary aim of the IM problem is to discover the seed set of K users that the number of users in the OSNs will be affected the most. Each user in the social network has two states: active and inactive, with the activation process switching the user's stable based on the influence of the seed set nodes.

The method inherits from Kempe’s work [1], many variations on the IM problem have been introduced because of its important role in many practical applications, such as viral marketing [17] [10], profit/revenue maximization [28] [29], social recommendation, health-care, rumor control, etc. However, the majority of the aforementioned existing works only focus solely on maximizing influence based on individual user nodes, ignoring the diffusion of influence to groups of individuals and communities. It can be shown that a small number of users who play a significant role in a group or community impact every user’s behavior or choice in that group or community. Hence, the problem of maximum influence on a group or society can produce more beneficial results than the problem of maximum influence on an individual. As a result of the above practical significance, there have been several recent studies on groups influence by choosing a seed set with at least k users, such as groups influence [52], competitive groups influence, groups influence via network embedding [50]. In this article, we propose a method for optimizing the influence of groups instead of individuals in OSNs: assuming there is a social network under the information propagation model with $G = (V, E)$, where the set of groups C_i is the target group. To evaluate the role and importance of each node in the propagation process, we provide each node a cost $c(u)$ to pay for activating a node in the network, a benefit $b(u)$ if that node has influenced. Then, a fixed threshold t_i is assigned to each group C_i and for each node in group C_i , a fixed score $s(u)$ is assigned. As a result, our contents are presented in our work as follows:

- Firstly, we present widely known information propagation models such as IC and LT. Our algorithm is effectively implemented for the IC model in this thesis, but it can be easily extended to other similar propagation models.
- Secondly, we propose a groups influence maximization algorithm that has more advantages than the influence of each node in the network, with a threshold t_i for each group.
- Thirdly, we present a group-based sampling (GBS) technique inherited from the RIS sampling method to estimate the epsilon group effect function.
- Finally, we conduct extensive experimentation of our proposed algorithm on real data social networks. The results show that our algorithm is efficient, providing quality solutions compared to other methods in terms of running time, memory usage, and total seed set cost.

4.1.2 Groups Influence with Minimum Cost in OSNs

Information diffusion in OSNs has been a hot research topic recently due to its tremendous commercial value. By leveraging the “word of mouth” effect, companies and organizations have used social networks as effective communication to product promotion, spread opinion and renovation, persuade voters, etc.

In a seminal work [1] published almost twenty years ago, Kempe *et al.* [1] introduced the IM problem, which aimed to find a set of k users (called *seed set*) in a social network

to initiate a propagation process that could possibly influence to the largest number of users under some predefined propagation model. Since then, this problem and its notable variants have demonstrated their significant role in various real-world problems, not only in viral marketing [10, 57], but also in other fields such as politics epidemics control [107, 94, 108, 43], social network monitoring [95, 109], recommendation system [110].

In many realistic scenarios, the user’s decision and behavior tend to be dependent on her group and most of important decisions or works, which would affect many individuals, are done by a group of key persons. Therefore, creating an impact on groups or communities is able to bring more benefits than that on individuals and deserves special consideration.

A significant example is the US President election where any candidate who gets an absolute majority of electoral votes (not the popular votes) will be chosen as the winner of the presidency. In reality, she often focuses on swing States to win in the end. Each State is allocated a fixed number of electoral votes that can be owned by a candidate if she wins the most popular votes in the State. To do this, her election campaigns might leverage social networks in order to persuade the most voters in the State to vote for her. As the budget of the campaign is limited, the candidate would not be able to convince all of the voters in the State, and she should target on most influenced voters in the State.

Motivated by the aforementioned examples, recent studies have been carried out on a general version of IM, whose objective is to maximize the number of influenced groups of users instead of the number of influenced individuals, by choosing some seed sets of at most k users (see, e.g., [111, 112, 55, 113]). Also, one can consider a dual problem of this problem by asking for the minimum number of seed nodes to influence a given number of groups. In line with this research, we investigate in this thesis a slightly general problem named Groups Influence with Minimum cost (GIM), which aims to find a seed set with minimum cost to influence all the target groups in the network.

Different from existing works, we consider the role of each user in a group by assigning a score to her, and each group admits a threshold representing how difficult it is to influence a group. Specifically, a group is influenced if the total score of influenced members reaches its threshold. One can easily see that GIM subsumes IM and its dual version as special cases and thus, it is $\#NP$ -hard to solve, not only by the combinatorial structure of the problem, but also by the $\#P$ -hardness of the calculation of the group influence function (denoted by $\sigma(\cdot)$). Another challenge is that $\sigma(\cdot)$ is neither a submodular nor supermodular, implying that the classical greedy algorithms when being applied to GIM may not result in any approximation guarantee.

In this work, we address all above challenges and our contributions can be summarized as follows. Assume that $G = (V, E)$ is a social network, $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$ is a set of target groups, each group C_i has a threshold t_i , and each node u has a cost $c(u)$ and a score $b(u)$.

4.2 Problem definition

4.2.1 Threshold Benefit for Groups Influence in OSNs

Next, we investigate Threshold Benefit for Groups Influence problem (TGI) in OSNs as follows:

Given a social network $G = (V, E)$ under the IC model and a collection of K disjoint groups $C = \{C_1, C_2, \dots, C_K\}$ (called target groups), where $C_i \subseteq V, C_i \cap C_j = \emptyset$, for every pair of nodes (i, j) with $i \neq j$.

Definition 4 (TGI problem) *An instance of TGI is given by (G, C, T) , where $G = (V, E)$ is a social network under the IC model, and C is a collection of disjoint target groups $\{C_1, C_2, \dots, C_K\}$, $C_i \cap C_j = \emptyset$. The objective is to find a seed set $S \subseteq V$ of minimum total cost that the benefit function is at least T , i.e., $S = \arg \min_{S' \subseteq V, \sigma(S') \geq T} c(S')$.*

To determine a group is influenced or not, we extend the influence group model in [111] by scoring each node in the group based on the fact that each user has a different role in their group. Thus, each node $u \in V$ has a *cost* $c(u)$ and a *score* $s(u)$. The weight $c(u)$ measures the cost or the price of the node u that has to pay if u is chosen as a seed node. The node score $s(u) > 0$ metrics the role of node u in the group $C(u)$. Each group C_i assigns a threshold t_i ($t_i > 0$), which reflects the minimum total score that we must reach if we want to influence group C_i , and a benefit value b_i is when C_i was influenced. We say that the group C_i is influenced if the total score of the influenced nodes in C_i is at least t_i .

We define a *cost function* $c : 2^V \rightarrow \mathbb{R}_+$ for Definition 4. For a given seed set $S \subseteq V$, the total cost of S is $c(S) = \sum_{u \in S} c(u)$. Therefore, a *groups influence function* $\sigma : 2^V \rightarrow \mathbb{R}_+$ denotes as follows:

Denote the total score of nodes which influence by S in C_i that

$$\sigma_i(S) = \sum_{v \in C_i} \mathbb{I}(S, v) s(v) \quad (4.1)$$

Then, $\sigma(S)$ is (expected) the number of groups in \mathcal{C} are influenced by the seed set S when the diffusion process ends, that is,

$$\sigma(S) = \sum_{C_i \in \mathcal{C}: \sigma_i(S) \geq t_i} b_i \quad (4.2)$$

In the special case where each group C_i has only one node the groups influence function $\sigma(\cdot)$ above becomes the influence spread function $\mathbb{I}(\cdot)$ of the IM problem. As a consequence, computing $\sigma(\cdot)$ is #P-hard. On the other hand, one can easily verify that the function $\sigma(\cdot)$ is neither submodular nor supermodular. The function $\sigma(\cdot)$ is submodular if for every pair of

subsets $A, B \subseteq V$ it holds that $\sigma(A) + \sigma(B) \geq \sigma(A \cup B) + \sigma(A \cap B)$. If the inequality holds in the reverse direction, we call $\sigma(\cdot)$ a supermodular function.

We first introduce the Group Benefit Sample (GBS) concept, by modifying Reachable Reverse (RR) sample, to estimate $\sigma_i(S)$

Definition 5 (GBS sample) *Given a graph G and group C_i , a GBS sample is generated by the following steps:*

1. Randomly select a node u with probability $\frac{s(u)}{\sum_{u \in C_i} s(u)}$ (call u a source node), where $\gamma_i = \sum_{u \in C_i} s(u)$
2. Generate a sample graph g according to the live-edge model under IC model.
3. Return a node set R_i that is reachable from u in g . We call u a source node.

Denote a random variable $X_i(S)$ as follows

$$X_i(S) = \begin{cases} 1, & \text{if } S \cap R_i = \emptyset \\ 0, & \text{otherwise} \end{cases} \quad (4.3)$$

We have the following lemma

Lemma 4 *Given a set node S , we have $\sigma_i(S) = \gamma_i \cdot \mathbb{E}[X_g(S)]$*

Given a set of GBSes \mathcal{R} , we have an estimation of $\hat{\sigma}_i(S)$ of $\sigma_i(S)$ as follows

$$\hat{\sigma}_i(S) = \frac{\gamma_i}{|\mathcal{R}|} \sum_{R_i \in \mathcal{R}} X_i(S) \quad (4.4)$$

We introduce an estimation of $\sigma(S)$

$$\hat{\sigma}(S) = \sum_{i \in [K]} \hat{\sigma}_i(S) \quad (4.5)$$

Threshold Benefit for Groups Influence in Online Social Networks (TGI problem): we studied a novel TGI problem defined as follows a given a social network G , a set of K target group $C = C_1, C_2, \dots, C_K$ and a threshold $T > 0$, TGI asks us to find a seed set S in G with the minimum cost so that the benefit gained from the influence of the groups in C is at least T . We propose a group influence maximization algorithm that has then advantages the influence of each node in the network with a threshold t_i for each group. Furthermore, we also present a group-based sampling (GBS) technique inherited from the RIS sampling method to estimate the epsilon group effect function.

4.2.2 Groups Influence with Minimum Cost in OSNs

Third, we now formally define the problem *Groups Influence with minimal Cost* (GIM).

Definition 6 (GIM problem) *An instance of GIM is given by (G, C) , where $G = (V, E)$ is a social network under the IC model, and C is a collection of disjoint target groups $\{C_1, C_2, \dots, C_K\}$, $C_i \cap C_j = \emptyset$. The objective is to find a seed set $S \subseteq V$ of minimum total cost that influences all groups in C , i.e., the problem asks to find*

$$S = \arg \min_{S' \subseteq V, \sigma(S')=K} c(S')$$

We call an algorithm a (γ, σ) -**bicriteria approximation** for GIM problem if it returns a solution S satisfying $\sigma(S) \geq \gamma \cdot$ and $c(S) \leq \sigma \cdot \text{OPT}$, for $\gamma, \sigma > 0$.

The most of current research focuses only on IM based on individual users on OSNs. However, we have recently studied the problem of the maximum spread of influence based on groups and communities. In fact, if that one or several users in the network can archive a large spread of influence based on their popularity, then, for a group or community where the users joined in have the same interest issues, specifying one or several factors that have a great impact on the behavior or choice of trends of the others is a good option. The issue of concern in advertising today is that product information or advertisement will reach the right potential customers, thereby reducing unnecessary advertising costs and increasing the chances of generating beneficial results. By search for group influence, not only quickly approaching when narrowing the search space, but also more accurately when the users are potential candidates in the community. Besides, approaching dense networks, which is typical of the reality OSNs such as Facebook, Twitter, etc., there is a high possibility of overlapping communities, and users can belong to multiple communities. The group influence problem for IM can be applied to spread information between groups and communities, in which this IM variant focuses on finding seed users who have the most influence in their communities and spreading that influence to other communities. We are now ready to introduce two problems as follows.

Groups Influence with Minimum Cost in Social Network (GIM problem): we studied a novel GIM problem which aims to find a seed set with smallest cost that can influence all target groups, where each user is associated with a cost and a group is influenced if the total score of the influenced users belong to the group is at least a certain threshold. To address this challenge, we propose a bi-criteria polynomial-time approximation algorithm with high certainty.

4.3 Proposed algorithms

4.3.1 Threshold Benefit for Groups Influence in OSNs

Among this subsection, we present our efficient algorithm for the TGI problem with input parameters, require a social network graph $G = (V, E)$, an influence threshold T , and a set of discrete groups C ($C_i \in C$, $C_i \subseteq V$), in which each C_i has a specific influence threshold t_i . The output of the algorithm is a set of seeds S provided that its influence estimate exceeds the influence threshold T so that resulting in the lowest total cost. We cover two search solutions in this algorithm.

Algorithm 3: Generating a GBS for Group C_i

Input: Graph $G = (V, E)$, C_i , γ_i

Output: A GBS R_j

- 1: Pick a source node $u \in C_i$ with probability $\frac{s(u)}{\gamma_i}$
 - 2: Initialize a queue $Q = \{u\}$ and $R_j = u$
 - 3: **while** Q is not empty **do**
 - 4: $v \leftarrow Q.pop()$
 - 5: **for** $x \in N_{in}(v) \setminus (R_j \cup Q)$ **do**
 - 6: With probability $p(x, v)$: $Q.push(x)$ and $R_j \leftarrow R_j \cup \{u\}$
 - 7: **end for**
 - 8: **end while**
 - 9: **return** R_j
-

We follow the method in [31] that uses the martingale theory for generating a sufficient number of samples to make a good approximation of the objective function. If the number of samples is at least $T \geq (2 + \frac{2}{3}\epsilon) \frac{1}{\mu} \frac{1}{\epsilon^2} \ln(\frac{1}{\delta})$ for $\delta \in (0, 1)$, $\hat{\sigma}_i(S)$ is an (ϵ, δ) -approximation of $\sigma(S)$, i.e.,

$$\Pr[(1 - \epsilon)\sigma_i(S) \leq \hat{\sigma}_i(S) \leq (1 + \epsilon)\sigma_i(S)] \geq 1 - \delta \quad (4.6)$$

In the first solution, we first search for the seed set S_i ($S_i \subseteq V$) that has the best effect on each group of C_i , so that the influence estimate of S_i exceeds t_i (as determined by the formula (4.4)), which is considered to have affected the entire C_i group, with the search condition mentioned in line 8 of the algorithm 4. In this process, we use the *GBS* sample generation model (mentioned in algorithm 3) to estimate the effect for each target group C_i , with the number of samples to be calculated as N_i (mentioned in line 3 algorithm 4), we assign these parameters $\epsilon = 0.1$ and $\delta = 1/n$ as a default setting. After joining the sets S_i together, we consider the seed set S to perform non-optimal seed elimination. In here, we are searching for seeds that, if the influence estimate after eliminating them (by formula (4.5)) still exceeds the threshold T , should be discarded, with the highest cost seeds being preferred, in order to

Algorithm 4: RIS-based Heuristic Algorithm

Input: Graph $G = (V, E)$, set of group $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$, ϵ, δ

Output: A seed set S

```
1:  $U \leftarrow V$ 
2:  $S_i \leftarrow \emptyset, \forall i \in [K]$ 
3:  $N_i = (2 + \frac{2}{3}\epsilon)|C_i|^{\frac{1}{\epsilon^2}} \ln(\frac{1}{\delta})$ 
   /* Find the first candidate solution  $S_1$  */
4:  $S_1 \leftarrow \emptyset$ 
5: for  $i = 1$  to  $K$  do
6:   Generate a set of  $\mathcal{R}_i$  containing  $N_i$  GBSs for group  $C_i$  by using Algorithm 3
7:   while  $\hat{\sigma}_i(S_i) < t_i$  do
8:      $u \leftarrow \arg \max_{v \in U \setminus S_i} \frac{\hat{\sigma}_i(S_i \cup \{v\}) - \hat{\sigma}_i(S_i)}{c(v)}$ 
9:      $S_i \leftarrow S_i \cup \{u\}$ 
10:     $U \leftarrow U \setminus \{u\}$ 
11:   end while
12: end for
13:  $S_0 \leftarrow \bigcup S_i$ 
14:  $S_1 = S_0$ 
15: for  $i = 1$  to  $K$  do
16:   while  $S_1 \neq \emptyset$  do
17:      $u \leftarrow \arg \max_{v \in S_1} c(v)$ 
18:     if  $\hat{\sigma}(S_1 \setminus \{u\}) \geq T$  then
19:        $S_1 \leftarrow S_1 \setminus \{u\}$ 
20:     end if
21:      $S_1 \leftarrow S_1 \setminus \{u\}$ 
22:   end while
23: end for
   /* Find the second candidate solution  $S_2$  */
24:  $S_2 \leftarrow \emptyset$ 
25: for  $i = 1$  to  $K$  do
26:   Generate a set of  $\mathcal{R}_i$  contains  $N'_i$  GBSs  $R_i$  by using Algorithm 3
27: end for
28: while  $\hat{\sigma}(S_2) < T$  do
29:    $u \leftarrow \arg \max_{v \in V \setminus S_2} \frac{\hat{\sigma}(S_2 \cup \{v\}) - \hat{\sigma}(S_2)}{c(v)}$ 
30:    $S_2 \leftarrow S_2 \cup \{u\}$ 
31: end while
32:  $S \leftarrow \arg \min_{S' \in \{S_1, S_2\}} c(S')$ 
33: return  $S$ 
```

obtain the seed set with the lowest total cost.

Besides, the second solution ensures IM search by the conventional method, allowing both solutions to be compared to find the best seed set. First, as solution 1, we generate *GBS* samples with the amount of N_i for the target groups C_i . Then, we perform a sequential

search for seeds in the vertex set V of the G graph until we reach threshold T then stopped, so that the influence of this seed set is the largest, satisfying the search condition in line 29 of algorithm 4. Finally, we compare the two solutions and choose the seed set with the lowest total cost.

4.3.2 Groups Influence with Minimum Cost in OSNs

In this section, we first introduce the concept of *Group Reverse Reachable (GRR)* sample, based on extending existing reverse reachable [2], and Reverse Influenceable Community (RIC) [111] samples, to estimate the influence group function $\sigma(\cdot)$.

Definition 7 (GRR sample) *Given an instance of GIM problem (G, C) , a GRR sample is generated by the following four steps:*

1. Randomly select a group C_i with probability $\frac{1}{K}$ (call C_i a source group).
2. Generate a sample graph g according to the live-edge model under IC model.
3. For each node $u \in C_i$, return a node set $R_g(u)$ that is reachable from u in g . We call u a source node.
4. Return a GRR sample $R_g = \{R_g(u) | \forall u \in C_i\}$, we also refer to $C(R_g)$ as the source group of R_g and $t(R_g)$ as the threshold of $C(R_g)$.

Our GRR sample is a nature extended version of the RR sample [1] by combining RR samples with the source node belongs to the source group. Moreover, our GRR sample is also an extended version of Reverse Influence Community (RIC) [111] which uses to estimate the group influence with the score of each node is equal to 1. The main differences between ours and RIC are: (1) the definition of GRR sample specifically determines whether or not a group is influenced via the total score of influenced nodes but this is not well defined in the RIC even when the score is equal to 1, (2) storing the reachable influence set for each node in GRR sample can help us exploit some important properties that are used for analyzing approximation ratio of proposed algorithms in the next sections.

For a set $S \subseteq V$ and a GRR sample R_g , and for $R_g(u) \in R_g$, if $R_g(u) \cap S \neq \emptyset$, we say that S covers node u , define:

$$\text{cover-score}(S, R_g(u)) = b(u) \cdot \min\{|S \cap R_g(u)|, 1\} \quad (4.7)$$

is score of source node u covered by S in R_g , we denote the following random variable:

$$X_g(S) = \begin{cases} 1, & \text{if } \sum_{R_g(u) \in R_g} \text{cover-score}(S, R_g(u)) \geq t_i \\ 0, & \text{otherwise} \end{cases} \quad (4.8)$$

The variable $X_g(S)$ indicates that the total score of nodes covered by S is greater than the threshold t_i or not? When $X_g(S) = 1$, C_i is influenced by S in the sample graph g . It's also said that a sample R_g is influenced by S . The probability of generating a sample R_g is:

$$\Pr[R_g] = \frac{1}{K} \sum_{g \sim G: reach(u, g) = R_g(u), \forall u \in C(R_g)} \Pr[g \sim G] \quad (4.9)$$

where $reach(u, g)$ is the set of nodes that can reach to u in g . We now show that we can estimate the value of $\sigma(S)$ by the expectation of $X_g(S)$ which is a key property of GRR sample that helps us devise the algorithms with theoretical bounds.

Lemma 5 *For any set $S \subseteq V$, we have $\sigma(S, C) = K \cdot \mathbb{E}[X_g(S)]$ where the expectation is taken over the randomness of g .*

Proof We have

$$\sigma(S) = \sum_{C_i \in C} \sum_{g \sim G} \Pr[g \sim G] X_g(S, C_i) \quad (4.10)$$

$$= K \sum_{C_i \in C} \frac{1}{K} \sum_{g \sim G} \Pr[g \sim G] X_g(S, C_i) \quad (4.11)$$

$$= \sum_{C_i \in C} \sum_{g \sim G} \Pr[C_i \text{ is a source group}] \Pr[g \sim G] X_g(S, C_i) \quad (4.12)$$

$$= K \cdot \mathbb{E}[X_g(S)], \quad (4.13)$$

where $X_g(S, C_i)$ is the variable $X_g(S)$ with source group C_i . The eq. (4.10) is due to the definition of $\sigma(S)$, and the eq. (4.12) follows from the fact that the probability of selecting the source node is $1/K$. ■

We introduce Algorithm 5 to generate a GRR sample. It first randomly chooses a source group C_i in \mathcal{C} (line 1). For each source node u in C_i , it generates a reverse reachable (RR) set $R_g(u)$ similar to that in the RIS model [4, 31, 5, 114].

Specifically, for each node u , the algorithm maintains a queue Q containing nodes that can be reached from u on a random sample graph. To do this, it first adds u into Q and selects the last node v in Q (line 4). It then randomly visits and adds its incoming neighbors into Q and $R_g(u)$ with probabilities equal to their edge probabilities. If incoming neighbors were visited the algorithm adds them into $R_g(u)$. This process continues until Q becomes empty.

From Lemma 5, we have an estimation of group influence function over a collection of GRR sets \mathcal{R} is:

$$\hat{\sigma}(S) = \frac{K}{|\mathcal{R}|} \cdot \sum_{R_g \in \mathcal{R}} X_g(S) \quad (4.14)$$

Algorithm 5: Generating a GRR sample

Input: Social network $G = (V, E)$, set of groups $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$

Output: A GRR sample R_g

Randomly pick a source group C_i among \mathcal{C}

```
1: for each node  $u \in C_i$  do
2:   Initialize a queue  $Q = \{u\}$  and  $R_g(u) = \{u\}$ 
3:   while  $Q$  is not empty do
4:      $v \leftarrow Q.pop()$ 
5:     for each node  $u \in N_{in}(v) \setminus (R_j \cup Q)$  do
6:       if  $(u, v)$  was visited then
7:          $Q.push(u)$ 
8:          $R_j \leftarrow R_j \cup \{u\}$ 
9:       else
10:        With probability  $p(u, v)$ :
11:          mark  $(u, v)$  is visited
12:           $Q.push(u)$ 
13:           $R_j \leftarrow R_j \cup \{u\}$ 
14:        end if
15:      end for
16:    end while
17:  end for
18: return  $R_g = \{R_g(u) | u \in C_i\}$ 
```

We observe that $X_g(S) \in [0, 1]$. Let random variable $M_i = \sum_{j=1}^i (X_g(S) - \mu_X)$, $\forall i \geq 1$, where $\mu = \mathbb{E}[X_g(S)]$. For a sequence of random variables M_1, M_2, \dots we have:

$$\mathbb{E}[M_i | M_1, \dots, M_{j-1}] = \mathbb{E}[M_{i-1}] + \mathbb{E}[X_i(A) - \mu] \quad (4.15)$$

$$= \mathbb{E}[M_{i-1}] \quad (4.16)$$

Therefore, M_1, M_2, \dots is a form of the martingale [100]. We utilize the following Lemma, which is trivially derived from the martingale theory in [100]:

Lemma 6 ([100]) *Given a set of MRR samples \mathcal{R} with $T = |\mathcal{R}|$ and $\lambda > 0$, we have:*

$$\Pr \left[\sum_{j=1}^T X_j(S) - T \cdot \mu \geq \lambda \right] \leq e^{-\frac{\lambda^2}{\lambda^2/3 + 2\mu T}} \quad (4.17)$$

$$\Pr \left[\sum_{j=1}^T Z_j(S) - T \cdot \mu \leq -\lambda \right] \leq e^{-\frac{\lambda^2}{2\mu T}} \quad (4.18)$$

In the Lemma 6, by replacing $\lambda = \epsilon T \mu$ with note that $\sigma(S) = K \mu$, we have:

$$\Pr[\hat{\sigma}(S) \geq (1 + \epsilon)\sigma(S)] \leq e^{-\frac{\epsilon^2 \mu T}{2 + \frac{2}{3}\epsilon}} \quad (4.19)$$

$$\Pr[\hat{\sigma}(S) \leq (1 - \epsilon)\sigma(S)] \leq e^{-\frac{\epsilon^2 \mu T}{2}} \quad (4.20)$$

Therefore, if the number of samples is at least $T \geq (2 + \frac{2}{3})\frac{1}{\mu} \frac{1}{\epsilon^2} \ln(\frac{1}{\delta})$ for $\delta \in (0, 1)$, $\hat{\sigma}_{\mathcal{R}}(S)$ is an (ϵ, δ) -approximation of $\sigma(S)$, i.e.,

$$\Pr[(1 - \epsilon)\sigma(S) \leq \hat{\sigma}_{\mathcal{R}}(S) \leq (1 + \epsilon)\sigma(S)] \geq 1 - \delta \quad (4.21)$$

In the following part, we are going to use this observation for devising algorithm that guarantees the estimation of group influence function of the solution. We introduce two of our proposed algorithms for GIM problem. From the analysis in Section 4.3.2, we can use $\hat{\sigma}(S)$ to closely estimate $\sigma(S)$ if the number of samples $|\mathcal{R}|$ is sufficiently large. Therefore, instead of solving GIM directly, we find the solution of the following problem:

Definition 8 (Samples Influence with Minimal Cost (SIM) problem) *Given a set of GRR samples \mathcal{R} . The problem asks to find a seed set $S \subseteq V$ with minimal total cost so that $\hat{\sigma}(S) = K$, i.e., find $S = \arg \min_{S' \subseteq V: \hat{\sigma}(S')=K} c(S')$.*

The idea behind of our algorithms is that we propose algorithms for solving SIM problem and use them as a core in our framework, which creates multiple candidate solutions and selects a final solution. We prove the approximation guarantees by utilizing martingale theory [100].

4.3.2.1 A bi-criteria approximation algorithm

We first propose the *Modified Greedy* (MoGreedy), a $(1 + \ln(|\mathcal{R}|t_{max}))\frac{b_{max}}{b_{min}}$ -approximation algorithm for SIM and then use it as the core of our bicriteria approximation algorithm.

An approximation algorithm for SIM.

First of all, it is not hard to see that SIM problem also is NP-hard and $\hat{\sigma}_{\mathcal{R}}(\cdot)$ is non submodular and non supermodular. Therefore, similar to GIM, it does not admit a naive greedy algorithm with any approximation ratio. We handle the challenges via introducing a lower bound function F of $\hat{\sigma}_{\mathcal{R}}(\cdot)$ and exploiting its properties.

Define $f(S, R_g)$ be the total score of all source nodes in R_g which are influenced by set S in sample graph g :

$$f(S, R_g) = \sum_{u \in C(R_g)} \text{cover-score}(S, R_g(u)) \quad (4.22)$$

We can see that $f(S, R_g)$ is a non negative and monotonic set function respect to $S \subseteq V$. Denote $\Delta_T f(S, R_g) = f(S \cup T, R_g) - f(S, R_g)$. We discover an important property of $f(\cdot, R_g)$ in the following.

Lemma 7 *For all $S \subseteq T \subseteq V$ and $v \notin T$, we have:*

$$\Delta_v f(S, R_g) \geq \frac{b_{\min}}{b_{\max}} \cdot \Delta_v f(T, R_g) \quad (4.23)$$

where $b_{\min} = \min_{v \in V} b(v)$, $b_{\max} = \max_{v \in V} b(v)$.

Define a set function $g(S, R_g) = \min \left\{ 1, \frac{f(S, R_g)}{t(R_g)} \right\}$, we also have following Lemma.

Lemma 8 *For all $S \subseteq T \subseteq V$ and $v \notin T$, we have:*

$$\Delta_v g(S, R_g) \geq \frac{b_{\min}}{b_{\max}} \cdot \Delta_v g(T, R_g) \quad (4.24)$$

Proof We consider following cases:

Case 1. $t(R_g) \geq f(S \cup \{v\}, R_g) \geq f(S, R_g)$, we have:

$$\begin{aligned} \Delta_v g(S, R_g) &= \\ &= \min \left\{ 1, \frac{f(S \cup \{v\}, R_g)}{t(R_g)} \right\} - \min \left\{ 1, \frac{f(S, R_g)}{t(R_g)} \right\} \\ &= \frac{f(S \cup \{v\}, R_g) - f(S, R_g)}{t(R_g)} \end{aligned}$$

In this case, we consider following three sub-cases:

- If $t(R_g) \geq f(T \cup \{v\}, R_g) \geq f(T, R_g)$ then $g(T \cup \{v\}, R_g) = \frac{f(T \cup \{v\}, R_g)}{t(R_g)}$, and $g(T, R_g) = \frac{f(T, R_g)}{t(R_g)}$.
- If $f(T \cup \{v\}, R_g) \geq t(R_g) \geq f(T, R_g)$, then $g(T \cup \{v\}, R_g) = 1$, and $g(T, R_g) = \frac{f(T, R_g)}{t(R_g)}$.
- If $f(T \cup \{v\}, R_g) \geq f(T, R_g) \geq t(R_g)$, $g(T \cup \{u\}, R_g) = g(T, R_g) = 1$.

In three cases above, we also have:

$$\begin{aligned} \Delta_v g(T, R_g) &= g(T \cup \{v\}, R_g) - g(T, R_g) \\ &\leq \frac{f(T \cup \{v\}, R_g) - f(T, R_g)}{t(R_g)} \\ &\leq \frac{b_{\max}}{b_{\min}} \frac{f(S \cup \{v\}, R_g) - f(S, R_g)}{t(R_g)} \quad (\text{Lemma 7}) \end{aligned}$$

Case 2. $f(S \cup \{v\}, R_g) \geq t(R_g) \geq f(S, R_g)$. In this case, we have $f(T \cup \{v\}, R_g) \geq f(S \cup \{v\}, R_g) \geq t(R_g)$, and $g(S, R_g) = \frac{f(S, R_g)}{t(R_g)} \leq g(T, R_g) \leq 1$. Therefore $g(T \cup \{v\}, R_g) = g(S \cup \{v\}, R_g) = 1$, and

$$\begin{aligned} \Delta_v g(S, R_g) &= 1 - g(S, R_g) = g(T \cup \{v\}, R_g) - g(S, R_g) \\ &\geq g(T \cup \{v\}, R_g) - g(T, R_g) \\ &\geq \frac{b_{\min}}{b_{\max}} \cdot (g(T \cup \{v\}, R_g) - g(T, R_g)) \end{aligned}$$

Case 3. $f(S \cup \{v\}, R_g) \geq f(S, R_g) \geq t(R_g)$. We obtain

$$\begin{aligned} \min \left\{ 1, \frac{f(S \cup \{v\}, R_g)}{t(R_g)} \right\} &= \min \left\{ 1, \frac{f(S, R_g)}{t(R_g)} \right\} = 1 \\ \min \left\{ 1, \frac{f(T \cup \{v\}, R_g)}{t(R_g)} \right\} &= \min \left\{ 1, \frac{f(T, R_g)}{t(R_g)} \right\} = 1 \end{aligned}$$

Therefore, $\Delta_v g(S, R_g) = \frac{b_{\min}}{b_{\max}} \Delta_v g(T, R_g) = 0$. The proof is proved. \blacksquare

In order to influence all samples in \mathcal{R} , it's necessary to find S such that $g(S, R_g) = 1, \forall R_g \in \mathcal{R}$. Therefore, we find S with minimal total cost such that

$$F(S, \mathcal{R}) = \frac{K}{T} \sum_{R_g \in \mathcal{R}} g(S, R_g) = \hat{\sigma}(S) = K \quad (4.25)$$

Proof It is easy to see that $\Delta_u F'(S) \geq \frac{b_{\min}}{b_{\max}} \cdot \Delta_u F'(T)$, for $S \subseteq T \subseteq V$ and $u \in V \setminus T$. Denote $S' = S^0 \setminus S_i = \{s'_1, s'_2, \dots, s'_t\}$, $S'_j = \{s'_1, s'_2, \dots, s'_j\}, j \leq t$, and $S'_0 = \emptyset$, we have:

$$\begin{aligned} K - F'(S_i) &= F(S^0) - F'(S_i) \leq F'(S^0 \cup S_i) - F'(S_i) \\ &= F'(S_i \cup S') - F'(S_i) \\ &= \sum_{j=1}^t (F'(S_i \cup S'_j) - F'(S_i \cup S'_{j-1})) \\ &\leq \sum_{j=1}^t \frac{b_{\max}}{b_{\min}} (F'(S_i \cup s'_j) - F'(S_i)) \\ &\leq \frac{b_{\max}}{b_{\min}} \text{opt} \cdot \frac{1}{c(S')} \sum_{j=1}^t (F'(S_i \cup s'_j) - F'(S_i)) \\ &\quad (\text{due to } c(S') \leq \text{opt}) \\ &= \frac{b_{\max}}{b_{\min}} \text{opt} \cdot \frac{\sum_{j=1}^t (F'(S_i \cup s'_j) - F'(S_i))}{\sum_{j=1}^t c(s'_j)} \end{aligned}$$

For any positive numbers a_1, \dots, a_l and b_1, \dots, b_l . According to [101], we have:

$$\min_{i=1\dots l} \frac{a_i}{b_i} \leq \frac{\sum_{i=1}^l a_i}{\sum_{i=1}^l b_i} \leq \max_{i=1\dots l} \frac{a_i}{b_i} \quad (4.26)$$

Apply above inequality, we obtain:

$$\begin{aligned} K - F'(S_i) &\leq \frac{b_{max}}{b_{min}} \text{opt} \cdot \max_{s'_j \in S'} \frac{F'(S_i \cup s'_j) - F'(S_i)}{c(s'_j)} \\ &\leq \frac{b_{max}}{b_{min}} \text{opt} \cdot \frac{F'(S_i \cup s_{i+1}) - F'(S_i)}{c(s'_j)} \\ &\leq \frac{b_{max}}{b_{min}} \text{opt} \cdot \frac{F'(S_{i+1}) - F'(S_i)}{c(s_{i+1})} \end{aligned}$$

By rearranging the terms we complete the proof. ■

Theorem 3 *Algorithm 6 provides a $\frac{b_{max}}{b_{min}}(1 + \ln(|\mathcal{R}|t_{max}))$ -approximation solution for SIM problem.*

Proof Denote by $S_i = \{s_1, s_2, \dots, s_i\}$ the solution of algorithm after iterations i of main loop. From Lemma 9, we have:

$$\begin{aligned} K - F'(S_{i+1}) &\leq \left(1 - \frac{b_{min}}{b_{max}} \frac{c(s_{i+1})}{\text{opt}}\right) (K - F'(S_i)) \\ &\leq e^{-\frac{b_{min}}{b_{max}} \frac{c(s_{i+1})}{\text{opt}}} \cdot (K - F'(S_i)) \\ &\leq e^{-\frac{b_{min}}{b_{max}} \frac{\sum_{j=1}^{i+1} c(s_{i+1})}{\text{opt}}} \cdot K \end{aligned}$$

It follows that

$$\begin{aligned} K - F'(S_{l-1}) &\leq e^{-\frac{b_{min}}{b_{max}} \frac{\sum_{j=1}^{l-1} c(s_j)}{\text{opt}}} \cdot K = e^{-\frac{b_{min}}{b_{max}} \frac{c(S_{l-1})}{\text{opt}}} \cdot K \\ \implies c(S_{l-1}) &\leq \frac{b_{max}}{b_{min}} \text{opt} \cdot \ln \frac{K}{K - F'(S_{l-1})} \\ &\leq \frac{b_{max}}{b_{min}} \text{opt} \cdot \ln(|\mathcal{R}|t_{max}) \end{aligned}$$

The last inequality is due to $K - F'(S_{l-1}) \geq K - F(S_{l-1}) \geq \frac{K}{|\mathcal{R}|} \frac{1}{t_{max}}$. Also, from Lemma 9 we imply that $c(s_l) \leq \frac{b_{max}}{b_{min}} \text{opt}$. Therefore:

$$c(S) = c(S_{l-1}) + c(s_l) \leq (1 + \ln(|\mathcal{R}|t_{max})) \frac{b_{max}}{b_{min}} \text{opt}$$

We complete the Proof. ■

Theorem 4 (Complexity of MoGreedy) *At each iteration, MoGreedy scans at most n nodes and calculates the marginal gain value of F' . Therefore, it takes $O(|S|n)$ time complexity.*

Bi-criteria approximation algorithm for GIM

Base on above theoretical analysis, we propose Modified Greedy (MoGreedy) (Algorithm 6) algorithm which utilizes the above characteristic of $F(\cdot, \mathcal{R})$. The general idea is that: we iteratively add a node v into the current solution S , which maximizes the marginal gain per its cost $\Delta_v(S)/c(v) = \frac{\min\{K, F(S \cup \{v\}, \mathcal{R})\} - F(S, \mathcal{R})}{c(v)}$ until the value of $F(S)$ achieves K .

Algorithm 6: MoGreedy(\mathcal{R}, \mathcal{C})

Input: A set of GRR samples \mathcal{R} , set of groups $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$

Output: Seed set S

```

1:  $S \leftarrow \emptyset$ 
2: while  $F(S, \mathcal{R}) < K$  do
3:    $v_{max} \leftarrow \arg \max_{v \in V \setminus S} \frac{\min\{K, F(S \cup \{v\}, \mathcal{R})\} - F(S, \mathcal{R})}{c(v)}$ 
4:    $S \leftarrow S \cup \{v_{max}\}$ 
5: end while
6: return  $S$ ;
```

Denote by $S_i = \{s_1, s_2, \dots, s_i\}$ the solution after i iterations in Algorithm 6, $S^0 = \{s_1^0, s_2^0, \dots, s_k^0\}$ is an optimal solution of SIM problem and let $\text{opt} = c(S^0)$, we obtain the following Lemma.

Lemma 9 *For each iteration i in the MoGreedy algorithm, we have:*

$$K - F'(S_i) \leq \text{opt} \cdot \frac{b_{max}}{b_{min}} \cdot \frac{F'(S_{i+1}) - F'(S_i)}{c(s_{i+1})} \quad (4.27)$$

where $F'(S) = \min\{K, F(S, \mathcal{R})\}$

We now present Groups Influence Approximation (GIA) algorithm, a $(1 - \epsilon, O(\ln K + \ln \ln n))$ -bi criteria approximation algorithm w.h.p for GIM problem. Our algorithm is inspired by the idea of Stop-and-Stare framework for IM problem [31], which devises a stopping condition to check the quality of candidate solutions.

Due to the different between GIM and IM, we have to give another stopping condition to check the candidate solutions and establish the number of required samples that ensure the theoretical bounds of the final solution

GIA algorithm operates in multiple iterations and finds a candidate solution at each iteration by leveraging MoGreedy algorithm and checks the quality of these solutions based on

Algorithm 7: GIA algorithm

Input: Graph $G = (V, E)$, set of groups $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$, $\epsilon, \delta \in (0, 1)$.

Output: A seed set S

```
1:  $N_{max} = (2 + \frac{2}{3}\epsilon) \frac{K}{\epsilon^2} \ln \left( \frac{2 \binom{n}{k_{max}}}{\delta} \right)$ ,  
2:  $N_1 \leftarrow (2 + \frac{2}{3}\epsilon) \frac{1}{\epsilon^2} \ln(\frac{1}{\delta})$   
3:  $i_{max} \leftarrow \lceil \log_2(N_{max}/N_1) \rceil$ ,  $\delta_1 \leftarrow \frac{\delta}{2^{i_{max}}}$   
4: Generate set of  $N_1$  samples  $\mathcal{R}_1$   
5: for  $i = 1$  to  $i_{max}$  do  
6:    $S_i \leftarrow \text{MoGreedy}(\mathcal{R}_i, \mathcal{C})$   
7:   Calculate  $F_l(S, \mathcal{R}_i, \epsilon, \delta_1)$  by Lemma 10  
8:   if  $F_l(S, \mathcal{R}_i, \epsilon, \delta_1) \geq K - \epsilon K$  or  $i = i_{max}$  then  
9:     break  
10:  else  
11:    Double size of  $\mathcal{R}_i$  by generating  $|\mathcal{R}_i|$  samples and adding them into  $\mathcal{R}_i$   
12:     $\mathcal{R}_{i+1} \leftarrow \mathcal{R}_i$   
13:  end if  
14: end for  
15: return  $S$ 
```

static evidences. Denote $k_{max} = \arg \max_{k=1 \dots n} \binom{n}{k}$, the algorithm needs at most

$$N_{max} = (2 + \frac{2}{3}\epsilon) \frac{K}{\epsilon^2} \ln(2 \binom{n}{k_{max}} / \delta)$$

samples and operates in at most $i_{max} = \lceil \log_2(N_{max}/N_1) \rceil$ iterations, where $N_1 = (2 + \frac{2}{3}\epsilon) \frac{1}{\epsilon^2} \ln(\frac{n}{\delta})$. We then show that N_{max} is the number of samples required that can ensure the approximation ratio by Theorem 5.

At iteration i , the algorithm generates a set of $(2 + \frac{2}{3}\epsilon) \frac{1}{\epsilon^2} \ln(\frac{1}{\delta}) 2^{i-1}$ samples \mathcal{R}_i and finds a candidate solution S_i by utilizing MoGreedy algorithm (line 5). We devise an stopping condition and check the quality of S_i in line 7. Note that, we do not reuse the stopping condition in [31], which is used in a recent work [111].

Our stopping condition is based on a lower bound of function $F_l(S, \mathcal{R}, \epsilon, \delta)$ of f , defined in Lemma 10.

We show that F_l gives a lower bound value of f w.h.p in Lemma 10. The algorithm then checks the termination condition in line 7. This condition also helps us prove the approximation ratio more succinctly than Stop and Stare. If the condition is true, the algorithm returns S_i as a final solution. Otherwise, it doubles size of \mathcal{R}_i and moves to the next iteration. The details of the algorithm described in Algorithm 7.

Theoretical Analysis. The approximation analysis is based on martingale theory [100]. Apply Lemma 6, we can show that $F_l(S, \mathcal{R}, \epsilon, \delta)$ is an lower bound function of $\sigma(S)$ with high

probability.

Lemma 10 *Given $\epsilon, \delta \in (0, 1)$ and a any set $S \subseteq V$ and a set of samples \mathcal{R} . Denote $c = \ln(1/\delta)$, $T = |\mathcal{R}|$ and $F_l(S, \mathcal{R}, \epsilon, \delta) = \min\{\hat{\sigma}_{\mathcal{R}}(S) - \frac{Kc}{3T}, \hat{\sigma}_{\mathcal{R}}(S) + \frac{K}{T}(\frac{2c}{3} - \sqrt{\frac{4c^2}{9} + 2Tc\frac{\hat{\sigma}_{\mathcal{R}}(S)}{K}})\}$. We have $\Pr[\sigma(S) \geq F_l(S, \mathcal{R}, \epsilon, \delta)] \geq 1 - \delta$*

Proof Denote $\mu = \frac{\sigma(S)}{K}$, $\hat{\mu} = \frac{\hat{\sigma}(S)}{K}$ and $c = \ln(1/\delta)$. Apply (??) in Lemma 6 with $\lambda = \frac{c}{3} + \sqrt{\frac{c^2}{9} + 2c\mu T}$, we have:

$$\Pr \left[\sum_{j=1}^T X_j(S) - T \cdot \mu \geq \lambda \right] \leq \delta \quad (4.28)$$

Therefore, the following event happen with probability at least $1 - \delta$:

$$\sum_{j=1}^T X_j(S) - T \cdot \mu \leq \lambda \quad (4.29)$$

$$\iff T\hat{\mu} - T\mu - \frac{c}{3} \leq \sqrt{\frac{c^2}{9} + 2c\mu T} \quad (4.30)$$

Solve the above inequality for μ , we have:

$$\mu \geq \min \left\{ \hat{\mu} - \frac{c}{3T}, \hat{\mu} + \frac{1}{T} \left(\frac{2c}{3} - \sqrt{\frac{4c^2}{9} + 2Tc\hat{\mu}} \right) \right\} \quad (4.31)$$

Replace $\mu = \frac{\sigma(S)}{K}$, $\hat{\mu} = \frac{\hat{\sigma}(S)}{K}$ into above inequality we obtain the proof. ■

Lemma 11 shows an interesting property of the optimal solution of GIM problem, which helps us find a connection between the our solution and the optimal solution.

Lemma 11 *For any set of GRR samples \mathcal{R} , we have: $\hat{\sigma}_{\mathcal{R}}(S^*) = K$*

Proof We prove this Lemma by contradiction. Assume that there is exist a set of GRR sample \mathcal{R} that $\hat{\sigma}_{\mathcal{R}}(S^*) < K$, then there is exist a set $\mathcal{R}_1 \subseteq \mathcal{R}$ so that $\sum_{R_g \in \mathcal{R}_1} X_g(S^*) = 0$. Denote Ω is the space of GRR samples with probability of generating a sample defined in eq.

(4.9), we have:

$$\begin{aligned}
\sigma(S^*) &= K\mathbb{E}[X_g(S^*)] = K \sum_{R_g \in \Omega} \Pr[R_g] X_g(S^*) \\
&= K \sum_{R_g \in \mathcal{R}_1} \Pr[R_g] X_g(S^*) + K \sum_{R_g \in \Omega \setminus \mathcal{R}_1} \Pr[R_g] X_g(S^*) \\
&= K \sum_{R_g \in \Omega \setminus \mathcal{R}_1} \Pr[R_g] X_g(S^*) \\
&= K \sum_{R_g \in \Omega \setminus \mathcal{R}_1} \Pr[R_g] < K
\end{aligned}$$

This contracts with the fact that S^* is an optimal solution of GIM problem. Therefore, $\hat{\sigma}_{\mathcal{R}}(S^*) = K$. ■

We formally claim the performance ratio of GIA algorithm in Theorem 5.

Theorem 5 *For any input parameters $\epsilon, \delta \in (0, 1)$, GIA algorithm returns a solution S satisfying $\Pr[\sigma(S) \geq K - \epsilon K] \geq 1 - \delta$ and $c(S) \leq \frac{b_{max}}{b_{min}}(1 + \ln((2 + \frac{2}{3}\epsilon)\epsilon^{-2})) + \ln K + \ln(nt_{max} \ln(n/\delta))$ OPT.*

Proof Denote $\mu = \frac{\sigma(S)}{K}, \hat{\mu} = \frac{\hat{\sigma}(S)}{K} = 1$ and $c = \ln(\binom{n}{k_{max}}/\delta)$. In Algorithm ??, we consider following bad events $B_i : \sigma(S_i) < K - \epsilon K$, for each iteration $i = 1, \dots, i_{max}$. We consider two following cases:

Case 1. If the algorithm terminates at some iterations $i = 1, \dots, i_{max} - 1$, apply Lemma ?? we have:

$$\begin{aligned}
\Pr(B_i) &= \Pr[\sigma(S_i) < K - K\epsilon] \\
&\leq \Pr[\sigma(S_i) < F_l(S_i, \mathcal{R}_i, \epsilon, \delta_1)] \leq \delta_1.
\end{aligned}$$

Case 2. If the algorithm stops at iteration i_{max} , applying Lemma ?? with a notice that $T = |\mathcal{R}| = (2 + \frac{2}{3}\epsilon)\frac{K}{\epsilon^2} \ln\left(2\frac{\binom{n}{k_{max}}}{\delta}\right) \geq 2c/\epsilon^2$ and $\hat{\mu} = 1$, the following event happens with a

probability of at least: $1 - \frac{\delta}{2\binom{n}{k_{max}}}$:

$$\begin{aligned}
\mu &\geq \min \left\{ \hat{\mu} - \frac{c}{3T}, \hat{\mu} + \frac{1}{T} \left(\frac{2c}{3} - \sqrt{\frac{4c^2}{9} + 2Tc\hat{\mu}} \right) \right\} \\
&= \min \left\{ 1 - \frac{c}{3T}, 1 + \frac{1}{T} \left(\frac{2c}{3} - \left(\frac{2c}{3} + \sqrt{2Tc} \right) \right) \right\} \\
&\quad (\text{Since } a^2 + b^2 \leq (a+b)^2, a, b > 0) \\
&\geq \min \left\{ 1 - \frac{\epsilon^2}{6}, 1 - \sqrt{\frac{2c}{T}} \right\} \\
&\geq \min \left\{ 1 - \frac{\epsilon^2}{6}, 1 - \epsilon \right\} \\
&\geq 1 - \epsilon
\end{aligned}$$

Hence, $\Pr[B_{i_{max}}] = \Pr[\mu < 1 - \epsilon] \leq \frac{\delta}{2\binom{n}{k_{max}}}$. Assume that $|S| = k$, there are at most $\binom{n}{k}$ possible solution, so we have:

$$\Pr[\forall S_{i_{max}} : B_{i_{max}}] \leq \binom{n}{k} \frac{\delta}{2\binom{n}{k_{max}}} \leq \frac{\delta}{2}$$

By the union bound of the probabilities, none of the events B_i , $i = 1, \dots, i_{max}$ happens with a probability at least $1 - (i_{max}\delta_1 + \frac{\delta}{2}) \geq 1 - \delta$, so we have:

$$\Pr[\sigma(S) \geq K - \epsilon K] \geq 1 - \delta$$

Denote $S_i^0 = \arg \min_{S: \sigma_{\mathcal{R}_i}(S)=K} c(S)$ and $\text{opt}_i = c(S_i^0)$, where $\sigma_{\mathcal{R}_i}(S)$ is an estimation of $\sigma(S)$ over \mathcal{R}_i . From Lemma 11, we have $\sigma_{\mathcal{R}_i}(S^*) = K$, therefore $\text{opt}_i \leq c(S^*)$. From Theorem 5, we have:

$$\begin{aligned}
c(S_i) &\leq \frac{b_{max}}{b_{min}} \cdot (1 + \ln(N_i t_{max})) \text{opt}_i \\
&\leq \frac{b_{max}}{b_{min}} \cdot (1 + \ln(N_{max} t_{max})) \text{opt}_i \\
&\leq \frac{b_{max}}{b_{min}} \left(1 + \ln \left(\left(2 + \frac{2}{3}\epsilon \right) \epsilon^{-2} \right) + \ln K + \right. \\
&\quad \left. + \ln(n t_{max} \ln(n/\delta)) \right) \text{OPT}
\end{aligned}$$

which completes the proof. ■

Theorem 6 (Complexity) GIA algorithm has

$$O\left((n \ln n + \ln(\frac{1}{\delta})\epsilon^{-2})|C|\eta + n^2\right) \log n$$

time complexity, where $\rho = |\bigcup_i C_i|$ and η is the expectation of influence spread of a node.

Proof The algorithm spends its running time on generating GRR samples and MoGreedy algorithm. Let $\mathbb{I}(S, v)$ denote the probability that a node set S influences v , and $\mathbb{I}(S)$ is influence spread of node set S , we obtain:

$$\begin{aligned} \mathbb{E}[|R_g|] &= \frac{1}{K} \sum_{C_i \in \mathcal{C}} \sum_{v \in C_i} \sum_{u \in V} \mathbb{I}(\{u\}, v) \\ &= \frac{1}{K} \sum_{v \in C} \sum_{u \in V} \mathbb{I}(\{u\}, v) \\ &= \frac{|C|}{K} \frac{1}{|C|} \sum_{u \in C} \sum_{v \in V} \mathbb{I}(\{u\}, v) \\ &= \frac{|C|}{K} \frac{1}{|C|} \sum_{u \in C} \mathbb{I}(\{u\}) \\ &= \frac{|C|}{K} \eta \end{aligned}$$

Therefore generating samples takes $O(N_{max} \frac{|C|\eta}{K})$ and the running time at any iteration i is at most:

$$\begin{aligned} &(k_{max} \ln n + \ln(\frac{1}{\delta})\epsilon^{-2})|C|\eta + |S_i|n \\ &= O\left((n \ln n + \ln(\frac{1}{\delta})\epsilon^{-2})|C|\eta + n^2\right) \end{aligned}$$

In addition,

$$\begin{aligned} i_{max} &= O(\log \frac{N_{max}}{N_1}) = O(\log(Kn \log n)) \\ &= O(\log K + \log n + \log \log n) \\ &= O(\log n) \quad (\text{Since } K \leq n) \end{aligned}$$

This implies the time complexity of the algorithm. ■

4.3.2.2 Exact Groups Influence Algorithm

We further propose Exact Groups Influence (EGI) algorithm, an (almost) exact solution for with high probability for GIM by using integer programming for solving SIM problem instead of MoGreedy algorithm and reusing the algorithmic framework of Algorithm 7.

Given a set of samples \mathcal{R} , we formulate the integer linear programming for solving SIM problem for an instance $(\mathcal{R}, \mathcal{C})$ of SIM problem, denoted by $\text{IP}(\mathcal{R}, \mathcal{C})$, as follows:

$$\min: \sum_{v \in V} x_v c(v) \quad (4.32)$$

$$\text{s.t: } \sum_{u \in R_g} \min \left\{ \sum_{v \in R_g(u)} x_v, 1 \right\} b(u) \geq t(R_g), \quad \forall R_g \in \mathcal{R} \quad (4.33)$$

$$x_v \in \{0, 1\}, \quad \forall v \in V \quad (4.34)$$

where

$$x_v = \begin{cases} 1, & \text{if } v \text{ is selected in the solution } S \\ 0, & \text{otherwise} \end{cases} \quad (4.35)$$

The objective of the IP is to select a seed set with minimal total cost. The constraints (4.33), (4.34) ensure all target groups be influenced by S .

Algorithm 8: EGI algorithm

Input: Graph $G = (V, E)$, set of K target groups $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$, $\epsilon, \delta \in (0, 1)$.

Output: Seed set S

```

1:  $N_{max} = (2 + \frac{2}{3}\epsilon) \frac{K}{\epsilon^2} \ln \left( \frac{2 \binom{n}{k_{max}}}{\delta} \right)$ ,  $N_1 \leftarrow \frac{1}{\epsilon^2} \ln(\frac{1}{\delta})$ 
2:  $i_{max} \leftarrow \lceil \log_2(N_{max}/N_1) \rceil$ ,  $\delta_1 \leftarrow \frac{\delta}{2(i_{max}-1)}$ 
3: Generate set of  $N_1$  samples  $\mathcal{R}_1$ 
4: for  $i = 1$  to  $i_{max}$  do
5:    $S_i \leftarrow$  a solution by solving  $\text{IP}(\mathcal{R}_i, \mathcal{C})$ .
6:   Calculate  $F_l(S_i, \mathcal{R}_i, \epsilon, \delta_1)$  by Lemma 10
7:   Double size of  $\mathcal{R}_i$  by generating  $|\mathcal{R}_i|$  samples and add them into  $\mathcal{R}_i$ 
8:    $\mathcal{R}_{i+1} \leftarrow \mathcal{R}_i$ 
9:   if  $F_l(S_i, \mathcal{R}_{i+1}, \epsilon, \delta_1) \geq (1 - \epsilon)K$  or  $i = i_{max}$  then
10:    return  $S_i$ 
11:   end if
12: end for
13: return  $S_i$ 
```

The details of EGI is presented in Algorithm 8. We only replace MoGreedy in GIA by solving

$\text{IP}(\mathcal{R}_i, \mathcal{C})$ (line 7), the rest of this algorithm is the same as EGI. By very similar reasoning with that in Theorem 5, we can also prove the approximation ratio of EGI as follows.

Theorem 7 *For any $\epsilon, \delta \in (0, 1)$, the Algorithm 8 returns a solution S satisfying $\sigma(S) \geq K - \epsilon K$ and $c(S) \leq \text{OPT}$ with probability at least $1 - \delta$.*

4.4 Experiment

In this section, we conduct our algorithm with many real datasets in large social networks. To provide a comprehensive performance experiment, we separate the experiment into two parts: one is for the TGI problem, and another is for the GIM problem.

4.4.1 Threshold Benefit for Groups Influence in OSNs

4.4.1.1 Experimental Settings

Datasets. In the experiment with the TGI algorithm, we experiment with 6 datasets for the problem of information propagation with different sizes. The datasets are described in Table 4.1 below. In which, the small and medium datasets calculated by the TGI algorithm are divided into several groups with a maximum of $K = 100$ groups (excluding discrete vertices), each group has from 1 to 10 vertices.

Table 4.1: TGI experimental datasets

Dataset	#Nodes	#Edges	Avg.Deg	Directed	#K	Source
Email-Eu-core	1,005	20,777	3.3	Directed	30	[115]
Gnutella	6,301	20,777	3.3	Directed	100	[102]
Wiki-vote	6,301	20,777	3.3	Directed	100	[116]
Net-Hept	15,233	58,891	5.5	Undirected	100	[104]
Net-Phy	37,154	231,584	13.4	Undirected	100	[12]
Email-Enron	36,692	183,831	5.0	Undirected	100	[103]

Parameters setting. All experiments are propagated under the IC information propagation model with the given edge probability $p(u) = p(u, v) = 1/|N_{in}(v)|$.

In the formula to calculate the number of GBS samples (line number 3 in algorithm 2), the parameters $\epsilon = 0.1$ and $\delta = 1/n$ are the default assignment values. And the Degree algorithm with the number of Monte Carlo iterations $t = 100,000$ times. The entire generating of GBS live-edge samples and the Monte Carlo propagation simulation iteration is time-optimized up to 4 threads.

For the TGI algorithm that aims to propagate the effect across groups, $\forall u \in C_i$ the score of u is assigned $s(u) = 1$, and the threshold of group C_i $t_i = \sum_{u \in C_i} s(u)/2$ for $i = 1 \dots K$. Each

node has its cost $c(u) = 1/|N_{out}(v)|$ with the support $(0, 1]$ and its benefit assigned $b(u) = 1$ if has entry degree, and vice versa.

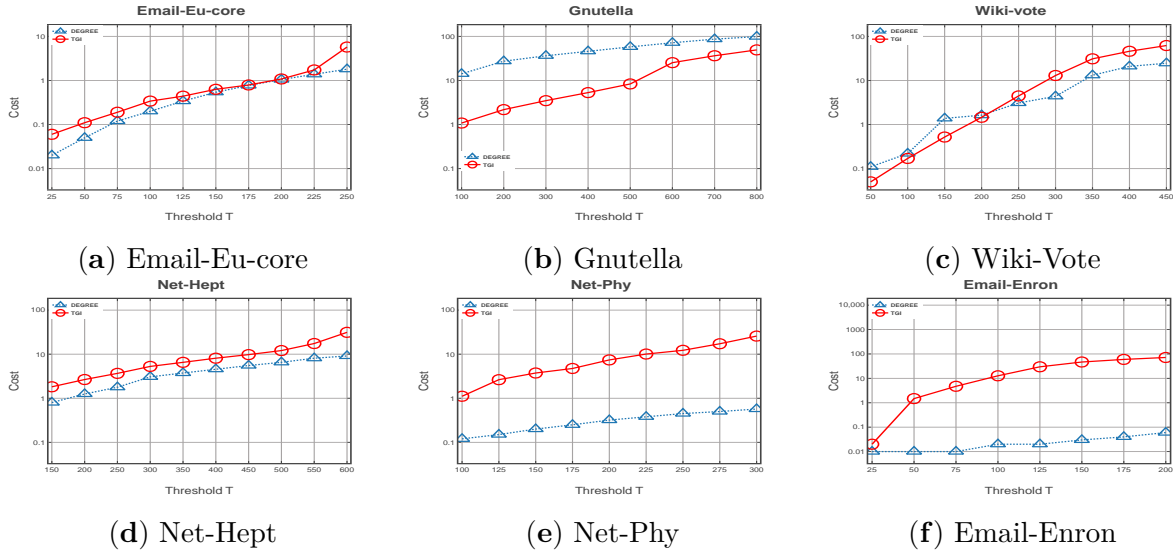
We experimented all dataset with a Linux machine with a 2 x Intel(R) Xeon(R) CPU E5-2630 v4 @2.20GHz, 64GB RAM DDR4 @ 2400MHz. The TGI algorithm and the High Degree algorithm are written in Python 3 language.

4.4.1.2 Experiment results

In the experimental comparison between algorithms, we give the advantages and disadvantages of the TGI algorithm.

Comparison of the total cost for seed set. Regarding the seed set cost comparison, the datasets are compared in (Fig. 4.1), we evaluate the High Degree algorithm better in most of the experiments. The total cost at High Degree thresholds is more optimal as the thresholds are larger. To explain this reason, we notice that there is an inconsistency between the given number of K groups and the size of the dataset, as well as the magnitude of the cluster peaks belonging to the group. On this point, we will improve in future experiments with more stable group quality.

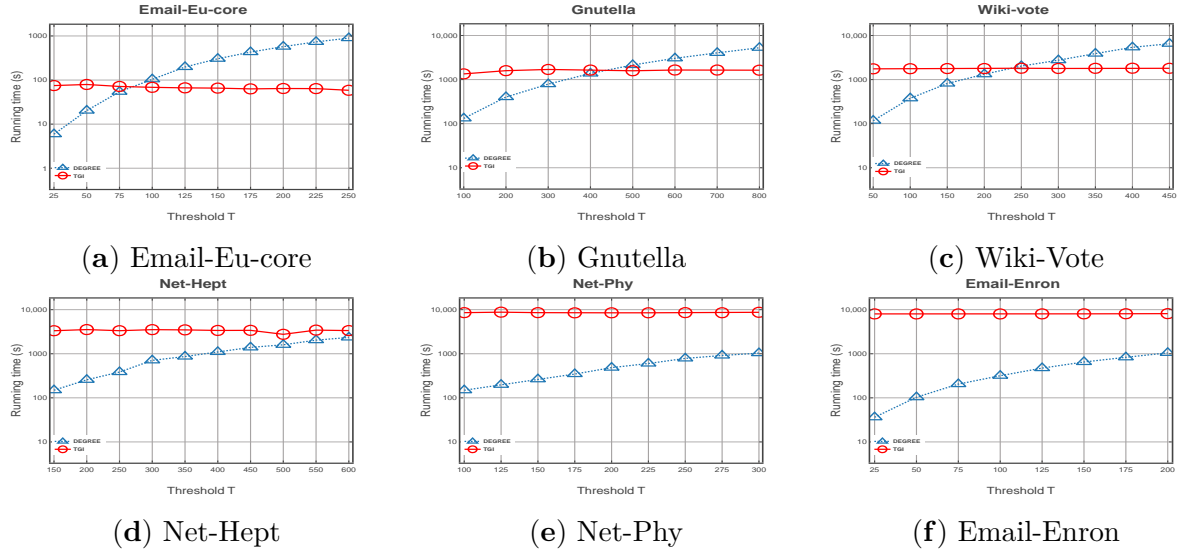
Figure 4.1: Total cost compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800



Comparison of running time. Regarding the running time comparison (Fig. 4.2), the running time of TGI will be better than the High Degree algorithm if the threshold T is predicted in the range that best fits the dataset. In general, the High Degree algorithm is a simple heuristic algorithm, using Monte Carlo simulation to propagate information, the algorithm will stop when the benefit estimate exceeds the threshold T , so when comparing

the time running, High Degree really dominated. We have ensured that the Monte Carlo computation time of High Degree is roughly equivalent to the generation time of the proposed GBS samples with an iteration of 1000 times. However, the TGI algorithm uses the method of removing unnecessary seeds to ensure that the propagation estimate is closest to the threshold T , in order to obtain the lowest cost. Since then, the propagation estimation value of TGI not only stops when it exceeds T but also considers the search and elimination time, the time that takes up almost the majority of the time in the experiment with the TGI algorithm. We observe that this is inevitable when traversing the thresholds T to find the best seed set.

Figure 4.2: Running time compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800



Comparison of memory usage. In terms of memory usage comparison, when comparing the two algorithms, the memory usage for the TGI algorithm is higher than High Degree but not significantly (Table. 4.2). However, both algorithms have a rather high cost of memory usage.

4.4.2 Groups Influence with Minimum Cost in OSNs

4.4.3 Experimental Settings

Dataset. We use public OSN datasets in the experiments, which are shown in Table 4.3. These data sets are widely used in the related work [dataset, 111].

Parameters setting. All experiments are under the IC model with edge probabilities set to $p(u, v) = 1/|N_{in}(v)|$. This weight setting is adopted from prior works [1, 4, 5, 31, 114, 10, 111]. We set parameters $\epsilon = 0.1$, $\delta = 1/n$ and the limited time is 6 hours. For the purpose of providing a comprehensive experiment, we divide the experiment in following two cases.

Table 4.2: Memory usage compared between TGI and DEGREE algorithms within threshold T_i from 25 to 800

(a)				(b)			
Dataset	Threshold	Algorithm		Dataset	Threshold	Algorithm	
		DEGREE	TGI			DEGREE	TGI
EmailEu-core	$T_i = 25$	359,940	430,740	Net-Hept	$T_i = 150$	387,632	488,976
	$T_i = 50$	360,040	435,748		$T_i = 200$	396,464	486,160
	$T_i = 75$	360,296	430,228		$T_i = 250$	394,032	476,380
	$T_i = 100$	360,296	428,948		$T_i = 300$	398,480	488,272
	$T_i = 125$	360,312	431,508		$T_i = 350$	398,540	486,160
	$T_i = 150$	360,632	437,540		$T_i = 400$	398,920	453,192
	$T_i = 175$	360,924	430,228		$T_i = 450$	398,924	459,920
	$T_i = 200$	361,032	430,228		$T_i = 500$	398,968	457,872
	$T_i = 225$	360,568	428,692		$T_i = 550$	394,596	484,112
	$T_i = 250$	361,364	434,292		$T_i = 600$	399,320	473,052
Gnutella	$T_i = 100$	371,736	476,388	Net-Phy	$T_i = 100$	455,364	545,824
	$T_i = 200$	371,172	487,524		$T_i = 125$	460,308	556,680
	$T_i = 300$	371,572	494,948		$T_i = 150$	463,116	556,000
	$T_i = 400$	371,784	487,268		$T_i = 175$	467,540	543,264
	$T_i = 500$	372,400	507,620		$T_i = 200$	469,600	544,032
	$T_i = 600$	372,904	487,332		$T_i = 225$	472,404	553,444
	$T_i = 700$	374,668	507,940		$T_i = 250$	475,424	549,412
	$T_i = 800$	375,164	469,348		$T_i = 275$	479,536	528,048
Wiki-Vote	$T_i = 50$	387,644	470,492	Email-Enron	$T_i = 300$	481,404	528,576
	$T_i = 100$	388,884	464,484		$T_i = 25$	454,092	558,012
	$T_i = 150$	389,016	456,416		$T_i = 50$	445,024	572,864
	$T_i = 200$	389,104	485,468		$T_i = 75$	459,980	533,628
	$T_i = 250$	389,292	475,776		$T_i = 100$	465,308	539,912
	$T_i = 300$	389,400	487,928		$T_i = 125$	458,180	571,372
	$T_i = 350$	388,644	456,424		$T_i = 150$	469,012	547,356
	$T_i = 400$	388,772	480,944		$T_i = 175$	463,532	573,584
	$T_i = 450$	390,248	461,668		$T_i = 200$	476,108	543,588

- **Case 1. Uniform Cost (UC).** In this case, $s(u) = 1, \forall u \in U$, and the thresholds $t_i = \sum_{u \in C_i} s(u)/2$ for $i = 1 \dots K$ according to the setting in [111] and the cost $c(u) = 1, \forall u \in V$.
- **Case 2. General Cost (GC).** Each node has its cost calculated under Normalized Linear model with the support $(0, 1]$ according to recent works [120, 121, 49] and $s(u) = 1, \forall u \in V$, and the thresholds $t_i = \sum_{u \in C_i} s(u)/2$ for $i = 1 \dots K$ according to the setting in [111].

Algorithms compared. To our knowledge, there is no existing algorithm can be adopted to solve the GIM problem directly. Therefore, we compare our GIA and EGI algorithms with the state-of-the-art algorithms for the closest problem: Influence Maximization at Community level (IMC) [111]. Also, we adapt High Degree, a common baseline algorithm for related problem on information diffusion [1, 27, 12]. These algorithms are described in detail as fol-

Table 4.3: GIM experimental datasets

Dataset	#Nodes	#Edges	Avg.Deg	Directed	Source
Facebook	747	60.05K	81	Directed	[117]
Wiki	7.1K	103.6K	15	Directed	[116]
Epinions	76K	508.8K	7	Directed	[118]
DBLP	317K	1.05M	4	Directed	[106]
Pokec	1.6M	30.6M	20	Directed	[119]

lows.

- **UBG** (Upper Bound Greedy) [111]. This is the best performance algorithm for the Influence Maximization at Community level (IMC) problem, which finds a set seed of k nodes that can influence to the largest number of groups while GIM problem requires to find the set of nodes with minimal cost that can influence all target groups. Therefore, we adapt UBG algorithm with some modifications as follows. We first initialize an empty candidate solution S . We then sequentially use UBG with k from 1 to n to find the best influence node then add it into S until the estimation $\hat{\sigma}(S)$ in each algorithm is at least $(1 - \epsilon)K$.
- **MAF** [111]. This is also an algorithm for IMC problem. We also modify it as the workflow in UBG to adapt for GIM problem.
- **High Degree (HD)**. We repetitively select a node with highest degree until the current solution influence all target groups.

For all above algorithms, we use the Monte-Carlo method in [122] to obtain an (ϵ, δ) -approximation for estimating influence group function. We implement GIA in C++ using CPLEX to solve the IP.

For each algorithm, we run 5 times to get the average results.

4.4.4 Experiment results

We first compare the performance of algorithms under UC case (Figure 4.3, 4.4 and 4.5). For the solution quality, measured by the size of seed set, GIA and EGI outperform other algorithms by a large gap. Specifically, they are up to 2.5 times better than UBG and MAF. EGI provides the best solution. It returns the solution with total cost up to 1.2 times lower than that of GIA.

Although UBG can give better results than MAF and HD in general but it does not give any approximation ratio for GIM problem. The selection of a fixed-size seed set in each

Figure 4.3: Size of seed set returned by GIA, EGI and other algorithms under the UC setting

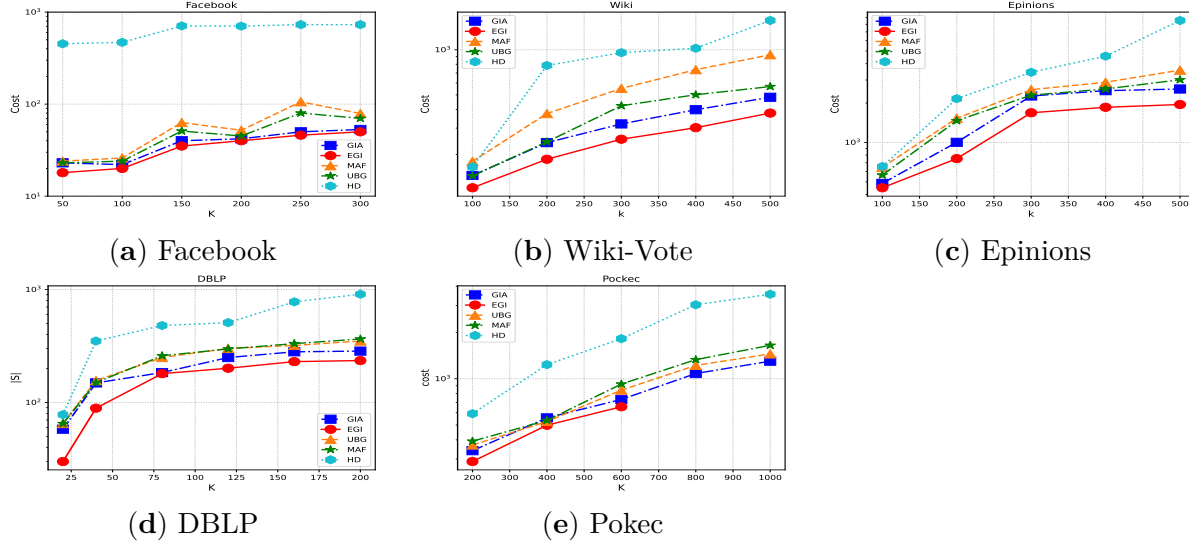
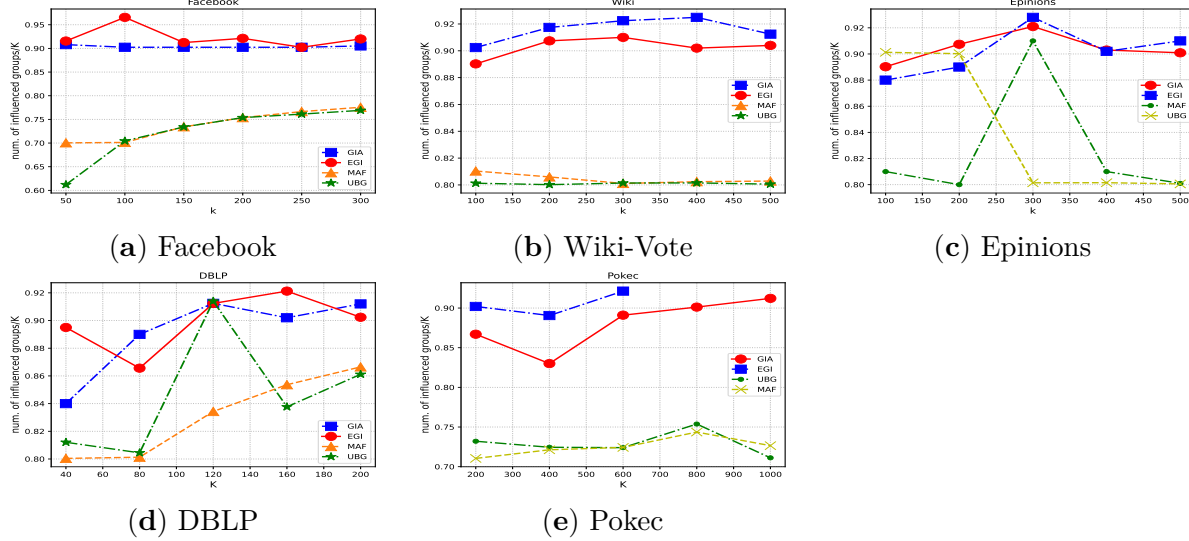
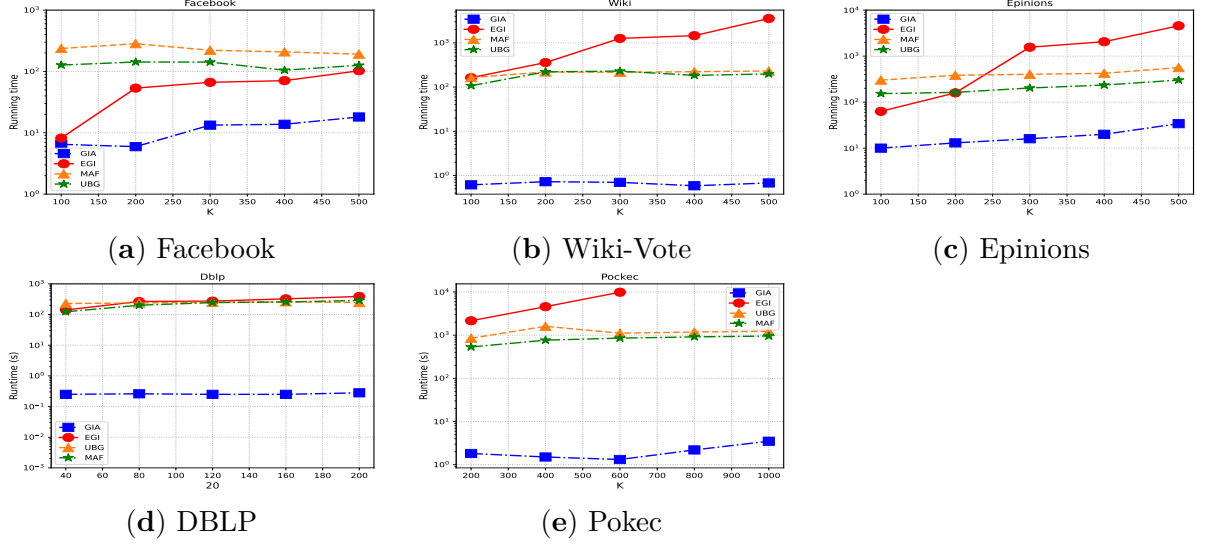


Figure 4.4: Ratio of number of influenced groups over K of GIA, EGI and other algorithms under the UC setting



iteration of binary search makes UBG to select many seed nodes unnecessary. The same happens with MAF. HD returns the poor results since it is a simple heuristic and only consider degree of nodes instead of influencing to groups. We further report the ratio of number of influenced groups over K of algorithms in Fig. 4.4. It can be seen that GIA EGI can output ratios that are above $(1 - \epsilon)$ in most cases and outperforms MAF and UBG. This is due to: (1) our algorithms always make sure all GRRs influenced and (2) stopping conditions in our algorithms ensure that $\sigma(S) \geq (1 - \epsilon)K$ with high certain. These results confirm that the

Figure 4.5: Running time of GIA, EGI and other algorithms under the UC setting



proposed algorithm is more efficient than the other algorithms. They not only ensure the approximation guarantee but also return the smaller-size set of nodes in practice.

Fig 4.5 shows the running time of algorithms. We do not report the running time of HD because it is a simple heuristic algorithm and can finish within some seconds. GIA is the fastest algorithm. It is up to approximately 840 and 646 times faster than MAF and UBG, respectively. This is because the mechanisms of MAF and UBG consist of many iterations to find the seed set that can reach to the terminal condition. In contrast, GIA follows the mechanism of our framework which can find the final solution after a few loops and it finishes with the largest network within only a few seconds. EGI has the longest running time since it uses IP solver to find the candidate solution instead of modified greedy. Although EGI gives the best quality of the solutions, it cannot be completed for the Pokec network when K is large within limited time. Interestingly, when K increases, the runtime of our algorithms does not decrease in some cases. This could be explained by: the larger the value of K , the earlier the terminal condition is satisfied, making our algorithms can finish within less iterations.

We next conduct the experiments under GC case. The results are shown in Figure 4.6, 4.7 and 4.8. We first compare the quality solution, measured by the total cost of seed set. Similar to the previous case, our algorithms outperform the others in terms of solution quality and EGI also provides the best solution. The reason is that MAF and UBG only consider the candidate seed sets with fixed-size and they do not consider the cost of nodes. Again, our algorithms give ratios that are above $(1 - \epsilon)$ in most cases. MAF and UBG give lower and unstable ratios. We also show the running time of algorithms in Figure 4.8. GIA is the fastest algorithm in all cases and EGI has the longest running time. These results are consistent with the previous case.

Figure 4.6: Size of seed set returned by GIA, EGI and other algorithms under the GC setting

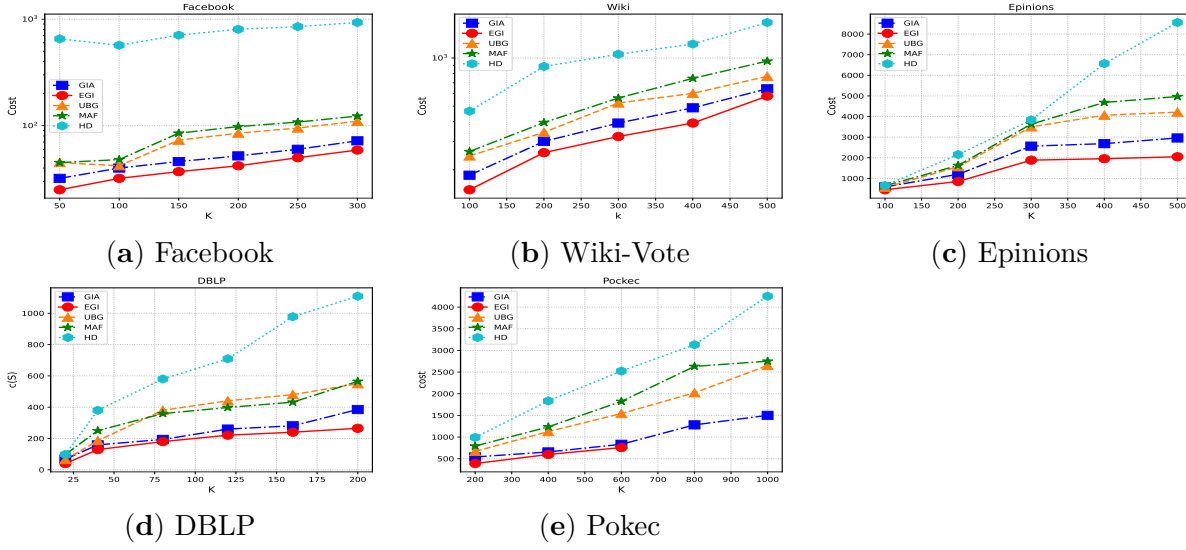
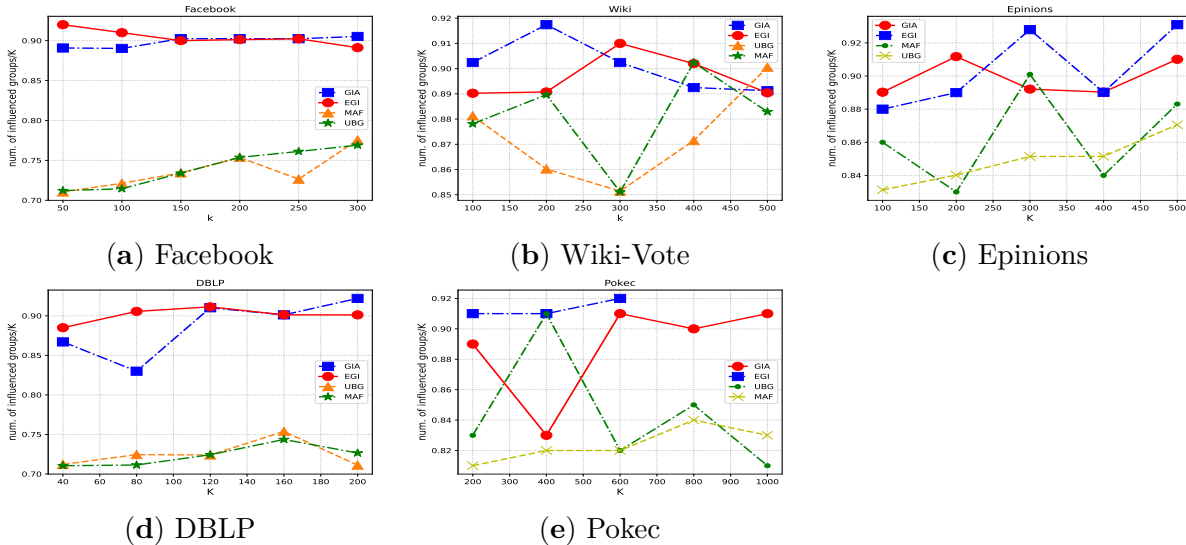


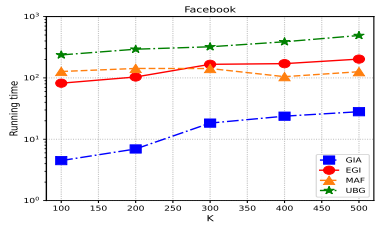
Figure 4.7: Ratio of number of influenced groups over K of GIA, EGI and other algorithms under the GC setting



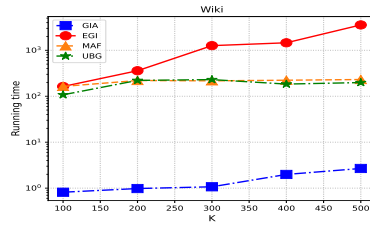
4.5 Discussion

In this chapter, we introduce two efficient algorithms for Groups Influence Maximization, named TGI problem and GIM problem. Next, we also present a framework algorithmic for find good candidate solutions with provable guarantee and a nove group reachable reverse sample technique. Finally, we extensive experiments to conduct on some real datasets in large network.

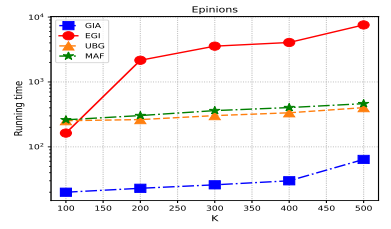
Figure 4.8: Running time of GIA, EGI and other algorithms under the GC setting



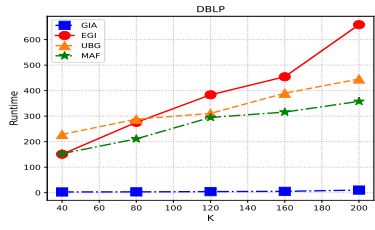
(a) Facebook



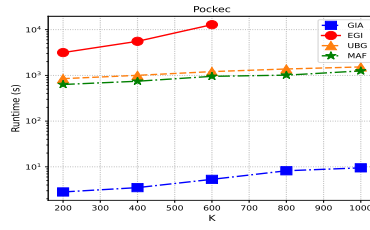
(b) Wiki-Vote



(c) Epinions



(d) DBLP



(e) Pokec

Chapter 5

Influence Maximization with k-topic in Social Network

This chapter proposes an overview of Submodular Maximization and k -Submodular Maximization problem in the first section. The second section then shows the problem definition and the main algorithm as the k -submodular cover, named k SC. Next, the last section discusses the result of the experiment of the proposed algorithm on real-world datasets.

5.1 Introduction

Submodular Maximization problems have recently received attention in many computer sciences and economics, such as machine learning, game theory, and combinatorial optimization. The Submodular Maximization problem can be defined as follows [123]:

Given a utility function $f : 2^V \mapsto \mathbb{R}^+$ that measures the quality of a given subset $S \subseteq V$ where $V = \{e_1, \dots, e_m\}$ is the ground set, f is the monotone submodular if for any $S \subseteq T \subseteq V$, $f(S) \geq f(T)$ and for any $x \notin T$, we have:

$$f(S \cup \{x\}) - f(S) \leq f(T \cup \{x\}) - f(T) \quad (5.1)$$

Basically, k -submodular function is a natural generalization of submodular function to k dimension. The k -submodular optimization problems have attracted a lot of attention because of their important role various domains such as influence maximization [69, 124, 125, 73], sensor placement [69, 124, 125], feature selection [60] and information coverage maximization [125]. Given a finite ground set V and an integer k , we define $[k] = \{1, 2, \dots, k\}$ and $(k+1)^V = \{(X_1, X_2, \dots, X_k) | X_i \subseteq V, \forall i \in [k], X_i \cap X_j = \emptyset, \forall i \neq j\}$ be a family of k disjoint sets. A function $f : (k+1)^V \mapsto \mathbb{R}_+$ is k -submodular iff for any $\mathbf{x} = (X_1, X_2, \dots, X_k)$

and $\mathbf{y} = (Y_1, Y_2, \dots, Y_k) \in (k+1)^V$, we have:

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \sqcap \mathbf{y}) + f(\mathbf{x} \sqcup \mathbf{y}) \quad (5.2)$$

where

$$\mathbf{x} \sqcap \mathbf{y} = (X_1 \cap Y_1, \dots, X_k \cap Y_k)$$

and

$$\mathbf{x} \sqcup \mathbf{y} = \left(X_1 \cup Y_1 \setminus \left(\bigcup_{i \neq 1} X_i \cup Y_i \right), \dots, X_k \cup Y_k \setminus \left(\bigcup_{i \neq k} X_i \cup Y_i \right) \right)$$

The problem of maximizing a k -submodular function has recently pad a lot of attentions due to its application in various domains. The problem is NP-hard in general. Singh *et al.* first studied on maximizing a k -submodular function with $k = 2$, i.e, bisubmodular [60]. For the problem of maximizing an unconstrained k -submodular function, Ward *et al.* [75] proposed a deterministic greedy algorithm with an approximation ratio of $1/3$. The author in [76] improved the approximation ratio to $\frac{k}{2k-1}$ in expectation by introducing a probability distribution to select any larger marginal element that has a higher probability. [77] derandomized the algorithm in [76] while remain the approximation ratio. However, the number of queries of their algorithm increased to $O(n^2 k^2)$.

The size constraint or cardinality constraint is an important constraint in studying the k -submodular function maximization problem. Under this constraint, Oshaka et al. [69] first proposed a greedy approach that returns an approximation ratio of $1/2$ and this approach can give an approximation ratio of $1/2$ for the matroid constraint [70]. The authors in [125] proposed multi-objective evolutionary algorithms that provided $1/2$ -approximation ratio under the size constraint but took $O(kn \log^2 B)$ queries in expectation. Recently, Nguyen [124] *et al.* proposed efficient streaming algorithms with performance guarantees.

Besides the size constraint, the problem of k -submodular maximization has studied under richer constraints, such as knapsack, budget, matroid, etc. Under a matroid constraint, the work [70] showed a Greedy algorithm can provide approximation ratio of $1/2$. [124] proved that the differentially private continuous Greedy method can provide the same approximation ratio. Recently, Tang *et al.* [81] first investigated the problem under knapsack constraint and devised a $(1/2 - 1/(2e))$ -approximation algorithm within $O(n^4 k^3)$ queries inspired by the Greedy algorithm in [82]. Pham *et al.* [88] considered the problem under the budget constraint, a general of knapsack constraint and proposed two deterministic and random streaming algorithms.

Although there have been many attempts to solve the problem of maximizing a k -submodular function under various constraints, however, the problem cannot reflect some practical applications where one needs to find the a set with smallest size so that the objective function

Table 5.1: Table of the usually used notations

Notation	Description
G	a graph.
n	the number of nodes in the graph G .
V	the set of nodes in the graph G . $ V = n$.
2^V	the subset family of V .
E	the set of edges in the graph G .
m	the number of edges in the graph. $ E = m$.
S	a seed set S
f, h	the submodular functions
$f(S)$	the number of influenced nodes/users by the seed set S
T	the threshold of $f(S)$
S_i	S at the i -th step where $i = 0, 1, 2, \dots$
v	an arbitrary node in V .
u	a neighbor node of v in V .
w	the set of weighted values of all edges in the graph G .
$p(u, v)$	the weighted value of the edge (u, v)
k	an upper bounded of $ S $.
g	a sample graph in G .
r	the number of sample graph generated from the original graph $G = (V, E)$.
$G_i = (V, E_i)$	the i -th sample graph generated from the original graph $G = (V, E)$, where $i = 1, 2, \dots, r$

greater than or equal to a certain threshold. Let's consider the following application:

Influence Threshold with k topics. Given a social network under an information diffusion model and k topics. Each user has a cost to start the influence under a topic which manifests how hard it is to initially influence to a respective person. Given a threshold T , we consider the problem of finding a set of users (seed set) with minimal size in which each user initially adopts a topic so that the expected number influenced users (who are eventually activated by at least one topic) is at least threshold T . In this application, the expected number of influenced users (objective) function is k -submodular where each user corresponds to each element in the set V [69, 124, 73].

Furthermore, streaming algorithms for submodular maximization have been extensively studied in many prior works because they produce solutions quickly and avoid excessive memory storage. In particular, the amount of data increases rapidly in the case of the online applications in social networks. In some practical applications, the data has increased very quickly that the memory computer cannot store an amount of data in time. Reducing data storage memory and providing guarantee solutions need to be considered. The streaming algorithm is one efficient approach to solving the submodular and k -submodular function maximization

problem. To our best knowledge, Badanidiyuru et al. [83] is the first to introduce an efficient streaming algorithm with a constant coefficient $(1/2 - \epsilon)$ -approximation guarantee. In addition, the author in [126] [127] [128] also investigated the submodular maximization problem in scenarios with many kinds of constraints by streaming algorithms. After that, Lan Nguyen et al. [73] also proposed two novel streaming algorithms for k -submodular maximization named DStream and RStream that used $O(\frac{nk}{\gamma} \log(\frac{(1+\epsilon)(1+\gamma)}{1-\epsilon} BM))$ and $O(\frac{nk\eta}{\gamma} \log(\frac{((1+\epsilon)^2 + 4B\epsilon)(1+\gamma)}{(1-\epsilon)^2}))$ query complexity when f was monotone and f was non-monotone. Similarly, [84] [89] also introduced the streaming algorithm for submodular function maximization under noise models. These methods, however, are only applied to the k -submodular function maximization issue; they are not suitable to the k -submodular cover problem. Due to the distinctions between the submodularity and k -submodularity settings, it cannot be directly applied to our k SC problems. Motivated by the abovementioned results, we develop an efficient streaming algorithm for the k -submodular cover problem with a bi-criteria approximation ratio. Our algorithm desires only a single pass or a few passes over all the data in real datasets. It provides theoretical guarantee solutions regarding several query complexity, memory usage, and approximation ratio setting.

5.2 Problem definition

Motivated by that observation, in this work, we study a novel problem named **k -submodular cover (k SC)**, defined as follows:

Definition 9 (k SC problem) *Given a finite set V , a monotone k -submodular function $f : (k+1)^V \mapsto \mathbb{R}_+$, and a threshold $T \leq \max_{\mathbf{x} \in (k+1)^V} f(\mathbf{x})$. The problem asks to find a solution $\mathbf{s} = (S_1, S_2, \dots, S_k)$ with the size $\text{supp}(\mathbf{s}) = \sum_{i=1}^k |S_i|$ is minimal so that the function $f(\mathbf{s}) \geq T$.*

We use following notations throughout the Ph.D thesis as the follows:

Given a finite set V and an integer k , we denote $[k]$ as the set $\{1, 2, \dots, k\}$. For $\mathbf{x} = (X_1, X_2, \dots, X_k)$, $\mathbf{y} = (Y_1, Y_2, \dots, Y_k) \in (k+1)^V$, we set if $e \in X_i$ then $\mathbf{x}(e) = i$ and i is called the position of e , otherwise $\mathbf{x}(e) = 0$. Adding an element $e \notin \text{supp}(\mathbf{x})$ into X_i can be represented by a tuple $\mathbf{x} \sqcup (e, i)$. When $X_i = \{e\}$, and $X_j = \emptyset, \forall j \neq i$, \mathbf{x} is denoted by (e, i) . For any $\mathbf{x}, \mathbf{y} \in (k+1)^V$, $\mathbf{x} \sqsubseteq \mathbf{y}$ iff $X_i \subseteq Y_i \forall i \in [k]$. Finally, we define $\text{supp}_i(\mathbf{x}) = X_i$, $\text{supp}(\mathbf{x}) = \cup_{i \in [k]} X_i$ and an empty k -set $\mathbf{0} = (\emptyset, \dots, \emptyset)$.

A function $f : (k+1)^V \rightarrow \mathbb{R}^+$ is monotone if for any $\mathbf{x} \sqsubseteq \mathbf{y}$, $f(\mathbf{x}) \leq f(\mathbf{y})$ and $f : (k+1)^V \rightarrow \mathbb{R}^+$ is k -submodular if for any \mathbf{x} and \mathbf{y} :

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \sqcap \mathbf{y}) + f(\mathbf{x} \sqcup \mathbf{y}) \quad (5.3)$$

where $\mathbf{x} \sqcap \mathbf{y} = (X_1 \cap Y_1, X_2 \cap Y_2, \dots, X_k \cap Y_k)$ and $\mathbf{x} \sqcup \mathbf{y} = (X_1 \cup Y_1 \setminus (\bigcup_{i \neq 1} X_i \cup Y_i), \dots, X_k \cup Y_k \setminus (\bigcup_{i \neq k} X_i \cup Y_i))$.

The function $f : (k+1)^V \mapsto \mathbb{R}_+$ is **k -submodular** iff for any $\mathbf{x} = (X_1, X_2, \dots, X_k)$ and $\mathbf{y} = (Y_1, Y_2, \dots, Y_k) \in (k+1)^V$, we have:

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \sqcap \mathbf{y}) + f(\mathbf{x} \sqcup \mathbf{y}) \quad (5.4)$$

where

$$\mathbf{x} \sqcap \mathbf{y} = (X_1 \cap Y_1, \dots, X_k \cap Y_k)$$

and

$$\mathbf{x} \sqcup \mathbf{y} = (Z_1, \dots, Z_k), \text{ where } Z_i = X_i \cup Y_i \setminus (\bigcup_{j \neq i} X_j \cup Y_j)$$

In this work, we consider f is *monotone*, i.e, for any $\mathbf{x} \in (k+1)^V, e \notin \text{supp}(\mathbf{x})$ and $i \in [k]$, we have:

$$\begin{aligned} \Delta_{(e,i)} f(\mathbf{x}) &= f(X_1, \dots, X_{i-1}, X_i \cup \{e\}, X_{i+1}, \dots, X_k) \\ &\quad - f(X_1, \dots, X_k) \geq 0 \end{aligned}$$

From Ohsaka [69], the k -submodularity of f implies the *orthant submodularity*, i.e,

$$\Delta_{(e,i)} f(\mathbf{x}) \geq \Delta_{(e,i)} f(\mathbf{y}) \quad (5.5)$$

for any $\mathbf{x}, \mathbf{y} \in (k+1)^V, e \notin \text{supp}(\mathbf{x}), \mathbf{x} \sqsubseteq \mathbf{y}$ and $i \in [k]$; and the *pairwise monotonicity*, i.e, for any $i, j \in [k], i \neq j$:

$$\Delta_{(e,i)} f(\mathbf{x}) + \Delta_{(e,j)} f(\mathbf{x}) \geq 0 \quad (5.6)$$

In this thesis, we assume that function f is normalized, $f(\emptyset) = 0$ and each element e was a positive cost $c_i(e)$ and we only consider $k \geq 2$ because if $k = 1$, the k -submodular function becomes the submodular function.

5.3 Proposed algorithm

Theorem 8 *Algorithm 9 is an $(1 + \log(1/\lambda), \frac{1-\lambda}{2})$ -bi criteria approximation algorithm, i.e, it returns a solution \mathbf{s} so that $\text{supp}(\mathbf{s}) \leq (1 + \log(1/\lambda)) \cdot \text{opt}$ and $f(\mathbf{s}) \geq T(1 - \lambda)/2$.*

Proof It is easy to see that there exist a positive $i \in [\Delta]$ so that $\frac{\text{opt}}{1+\epsilon} \leq v = (1 + \epsilon)^i \leq \text{opt}$. We consider following cases:

Algorithm 9: k -Threshold Greedy

Input: a k -submodular function $f : (k+1)^V \rightarrow \mathcal{R}_+$, an integer threshold T , accuracy parameters ϵ, λ

Output: A solution \mathbf{s}

```
1: for  $i = 0$  to  $\lceil \log(n)/\epsilon \rceil$  do
2:    $v \leftarrow (1 + \epsilon)^i$ 
3:    $\mathbf{s}_i \leftarrow \mathbf{0}$ 
4:    $j \leftarrow 0$ 
5:   while  $j \leq \log(1/\lambda)/\epsilon$  do
6:      $\theta \leftarrow \frac{1-\epsilon}{v}(T - 2f(\mathbf{s}_i))$ 
7:     for each node  $e \in V$  do
8:        $i_e \leftarrow \arg \max_{i \in [k]} \Delta_{(e, i_e)}(\mathbf{s}_i)$ 
9:       if  $\Delta_{(e, i_e)}f(\mathbf{s}_i) \geq \theta$  then
10:         $\mathbf{s}_i \leftarrow \mathbf{s}_i \sqcup (e, i_e)$ 
11:      end if
12:      if  $f(\mathbf{s}_i) \geq T(1 - \lambda)/2$  then
13:        break;
14:      end if
15:    end for
16:     $j \leftarrow j + 1$ 
17:  end while
18: end for
19: return  $\arg \min_{j \in [l], f(\mathbf{s}_j) \geq T(1-\lambda)/2} |\text{Supp}(\mathbf{s}_j)|$ 
```

Case 1: The main loop meets the condition in line 13 and it terminates before $j < \log(1/\lambda)\epsilon$. We have $f(\mathbf{s}) \geq (1 - \lambda)/2$

■

Lemma 12 *At the end of the outer loop of the algorithm 10, we have:*

$$T \leq 2f(\mathbf{s}_j) + \text{opt} \cdot \theta_j \quad (5.7)$$

Proof To prove the lemma, for the candidate solution \mathbf{s}_j we first define following notations:

- (e^l, i^l) as the l -th element added into \mathbf{s}_j .
- \mathbf{s}_j^l the solution when adding l elements.
- $\mathbf{o}^l = (\mathbf{o} \sqcup \mathbf{s}_j^l) \sqcup \mathbf{s}_j^l$.
- $\mathbf{o}^{l-1/2} = (\mathbf{o} \sqcup \mathbf{s}_j^l) \sqcup \mathbf{s}_j^{l-1}$.

Algorithm 10: Streaming Algorithm

Input: an evaluation oracle of a k -submodular function $f : (k+1)^V \mapsto \mathbb{R}^+$, threshold $T > 0$, ϵ .

Output: A solution \mathbf{s}

```

1:  $l \leftarrow \lceil \log(n)/\epsilon \rceil$ 
2:  $\mathbf{s}_1, \dots, \mathbf{s}_l \leftarrow \mathbf{0}$ 
3: for each node  $e \in V$  do
4:   for  $j = 0$  to  $l$  do
5:      $\theta_j \leftarrow \epsilon T / (1 + \epsilon)^j$ 
6:      $i_e \leftarrow \arg \max_{i \in [k]} \Delta_{(e,i)}(\mathbf{s}_j)$ 
7:     if  $\Delta_{(e,i_e)}(\mathbf{s}_j) \geq \theta_j$  then
8:        $\mathbf{s}_j \leftarrow \mathbf{s}_j \sqcup (e, i_e)$ 
9:     end if
10:    if  $\text{supp}(\mathbf{s}_j) = (1 - \epsilon)(1 + \epsilon)^j / (2\epsilon)$  then
11:      break;
12:    end if
13:  end for
14: end for
15:  $\mathbf{s} \leftarrow \arg \min_{\mathbf{s}_j, j \in [l], f(\mathbf{s}_j) \geq T(1-\epsilon)/2} \text{supp}(\mathbf{s}_j)$ 
16: return  $\mathbf{s}$ 

```

- $\mathbf{s}_j^{l-1/2}$: If $e^l \in \text{supp}(\mathbf{o})$, then $\mathbf{s}_j^{l-1/2} = \mathbf{s}_j^{l-1} \sqcup (e^l, \mathbf{o}(e^l))$. If $e^l \notin \text{supp}(\mathbf{o})$, $\mathbf{s}_j^{l-1/2} = \mathbf{s}_j^{l-1}$.
- $\mathbf{u}^t = \{(u_1, i_1), (u_2, i_2), \dots, (u_r, i_r)\}$: a set of elements that are in \mathbf{o}^t but not in \mathbf{s}_j^t , $r = |\text{supp}(\mathbf{u}^t)|$.
- $\mathbf{u}_q^t = \mathbf{s}_j \sqcup \{(u_1, i_1), (u_2, i_2), \dots, (u_q, i_q)\}, \forall 1 \leq q \leq r$ and $\mathbf{u}_0^t = \mathbf{s}_j$.

By the monotonicity of and the k -submodular property of f and note that $T \leq f(\mathbf{o}) = f(\mathbf{o}^0)$. For $t = |\text{supp}(\mathbf{s}_j)|$ we obtain:

$$\begin{aligned}
f(\mathbf{o}) - f(\mathbf{o}^t) &= \sum_{i=0}^t (f(\mathbf{o}^{i-1}) - f(\mathbf{o}^i)) \\
&\leq \sum_{l=0}^t (f(\mathbf{o}^{l-1}) - f(\mathbf{o}^{l-1/2})) \quad (\text{due to the monotonicity of } f) \\
&\leq \sum_{l=0}^t (f(\mathbf{s}_j^{l-1/2}) - f(\mathbf{s}_j^{l-1})) \quad (\text{due to the } k\text{-submodularity}) \\
&\leq \sum_{l=0}^t (f(\mathbf{s}_j^l) - f(\mathbf{s}_j^{l-1})) \quad (\text{due to the selection of algorithm}) \\
&\leq f(\mathbf{s}_j)
\end{aligned}$$

Therefore,

$$T - f(\mathbf{s}_j) \leq f(\mathbf{o}) - f(\mathbf{o}^t) + f(\mathbf{o}^t) - f(\mathbf{s}_j) \quad (5.8)$$

$$\leq f(\mathbf{s}_j) + \sum_{l=1}^{|supp(\mathbf{u}^t)|} (f(\mathbf{u}_l^t) - f(\mathbf{u}_{l-1}^t)) \quad (5.9)$$

$$\leq f(\mathbf{s}_j) + \sum_{l=1}^{|supp(\mathbf{u}^t)|} (f(\mathbf{s}_j \sqcup (u_l, i_l)) - f(\mathbf{s}_j)) \quad (5.10)$$

$$\leq f(\mathbf{s}_j) + \text{opt} \cdot \theta_j \quad (5.11)$$

which implies the proof. \blacksquare

Theorem 9 *Algorithm 10 is an $(\frac{(1-\epsilon)(1+\epsilon)}{2\epsilon}, \frac{1-\epsilon}{2})$ -bi criteria approximation algorithm, i.e, it returns a solution \mathbf{s} so that $supp(\mathbf{s}) \leq (1 - \epsilon^2)/(2\epsilon) \cdot \text{opt}$ and $f(\mathbf{s}) \geq T(1 - \epsilon)/2$.*

Proof Since there exist $\mathbf{x} \in V^{(k+1)}$ so that $f(\mathbf{x}) \geq T$, $\text{opt} \leq n$. Therefore, there exists an integer $j \in \{0, 1, \dots, \lceil \log(n)/\epsilon \rceil\}$ so that $\frac{v}{1+\epsilon} \leq \text{opt} \leq v$ where $v = (1 + \epsilon)^j$. The inner loop of the algorithm terminates when meets the condition in line 9, or scans over the ground set V one time. So we consider two following cases:

Case 1. If the terminal condition in line 9 is meet, we have $supp(\mathbf{s}_j) = \frac{(1-\epsilon)(1+\epsilon)^j}{2\epsilon} = \frac{1-\epsilon}{2\epsilon} \text{opt}$, then

$$f(\mathbf{s}_j) \geq supp(\mathbf{s}_j) \cdot \theta_j = \frac{(1 - \epsilon)(1 + \epsilon)^j}{2\epsilon} \frac{\epsilon T}{(1 + \epsilon)^j} = \frac{T(1 - \epsilon)}{2} \quad (5.12)$$

Case 2. If the terminal condition in line 9 is not meet, we have $supp(\mathbf{s}_j) < \frac{(1-\epsilon)(1+\epsilon)^j}{2\epsilon} \leq \frac{1-\epsilon}{2\epsilon} \text{opt}$. By Lemma 12, we have:

$$T \leq f(\mathbf{o}) \leq 2f(\mathbf{s}_j) + supp(\mathbf{o}) \cdot \theta_j \quad (5.13)$$

$$= 2f(\mathbf{s}_j) + v \cdot \frac{\epsilon T}{v} = 2f(\mathbf{s}_j) + \epsilon T \quad (5.14)$$

\blacksquare

which implies that $f(\mathbf{s}_j) \geq \frac{T(1-\epsilon)}{2}$. By the selection of the final solution, we have $f(\mathbf{s}) \geq \frac{T(1-\epsilon)}{2}$ and $supp(\mathbf{s}) \leq supp(\mathbf{s}_j) \leq \frac{1-\epsilon}{2\epsilon} \text{opt}$. The proof is completed.

5.4 Experiment

5.4.1 Datasets

We chose a few graph data sets suitable for the problem of maximizing influence in the experimental section, including small datasets such as Facebook, Wiki-Vote, and Epinions; medium and large datasets such as Stanford and DBLP. These datasets are frequently used to assess algorithmic results for cutting-edge algorithms. Furthermore, we studied the practical application of the algorithm based on the research experiment [joco], using the data set "Sensor" [cite paper sensor]. The data powders are described in turn below.

Table 5.2: kSC experimental datasets

Dataset	#Nodes	#Edges	Avg. Degree	Source
Facebook	4,039	88,234		[129]
Wiki-Vote	7,115	103,689	15	[116]
Epinions	75,879	508,837	7	[118]
Stanford	281,903	2,312,497		[103]
DBLP	655,312	4,156,343	6.1	[103]

5.4.2 Algorithms Compared

In our study, we run experiments on data sets using the algorithms mentioned in the previous sections. Furthermore, we employ two algorithms, the Greedy and the RandomChoose for experimental comparison, which describe by following:

- Greedy: While the threshold T is unaffected, continue searching (e, i_e) until the Delta function of solution s reaches its maximum.
- RandomChoose: If the threshold T is not reached, will search for a random vertex e outside the seed node set and select the seed set i_e so that (e, i_e) will obtain the function $\Delta(s)$ reaching max.

The results of each algorithm will be compared to one another and to the experimental results of [88] on the time, number of queries, number of seed nodes, and memory usage goals.

5.4.3 Parameters Setting

Information propagation under LT model with probability $p(u, v) = \frac{1}{|N_{in}(v)|}$ (weight). Moreover, the probability of vertex activation by the i -th topic is assigned to $\frac{i}{2k}$, where i belong to range $[1..K]$. We use the RIS propagation model under the LT model to generate sample

Algorithm 11: Greedy Algorithms

Input: an evaluation oracle of a k -submodular function $f : (k+1)^V \mapsto \mathbb{R}^+$, threshold $T > 0$.
Output: A solution \mathbf{s}

- 1: $\mathbf{s} \leftarrow \mathbf{0}$
- 2: **while** $f(\mathbf{s}) \leq T$ **do**
- 3: $(e, i_e) \leftarrow \arg \max_{i \in [k], e \in V \setminus \text{Supp}(\mathbf{s})} \Delta_{(e,i)}(\mathbf{s})$
- 4: $\mathbf{s} \leftarrow \mathbf{s} \sqcup (e, i_e)$
- 5: **end while**
- 6: **return** \mathbf{s}

Algorithm 12: RandomChoose Algorithms

Input: an evaluation oracle of a k -submodular function $f : (k+1)^V \mapsto \mathbb{R}^+$, threshold $T > 0$.
Output: A solution \mathbf{s}

- 1: $\mathbf{s} \leftarrow \mathbf{0}$
- 2: **while** $f(\mathbf{s}) \leq T$ **do**
- 3: $e \leftarrow \text{rand}_{e \in V \setminus \text{Supp}(\mathbf{s})}$
- 4: $(e, i_e) \leftarrow \arg \max_{i \in [k]} \Delta_{(e,i)}(\mathbf{s})$
- 5: $\mathbf{s} \leftarrow \mathbf{s} \sqcup (e, i_e)$
- 6: **end while**
- 7: **return** \mathbf{s}
- 8:

sets with a fixed number of $n = 1000$ for each topic in order to measure its topic influence. Epsilon $e = 0.1$, lambda $\lambda = 0.1, \dots$ are the default parameters. We measure the number of topics $K = 3$ for each threshold from 100 to 1000 in these experiments.

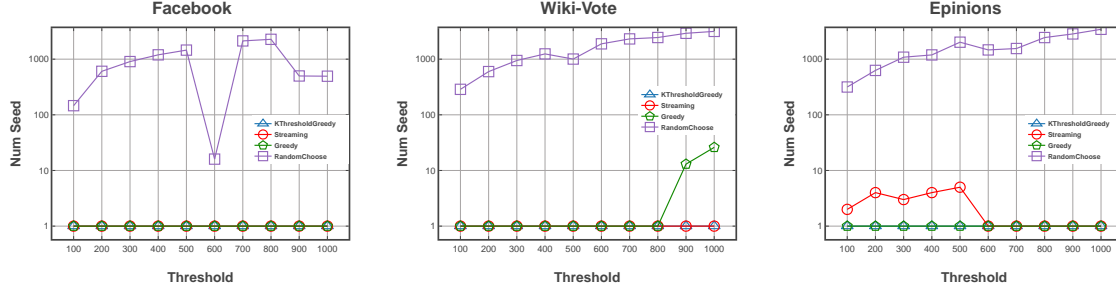
5.4.4 Experiment result

Figure 5.1(a) shows the number of seed nodes found when experimenting with the algorithms. We rate the RandomChoose algorithm 11 lower than the remaining methods with seed nodes that are 15 to 3000 times higher. When selecting a small number of very influential seed nodes that reach the threshold T , algorithms 9 and 10 are beneficial. Also, the Greedy method promotes the benefit of a regular increase in the number of seed nodes while keeping the number of seed nodes low for small data sets.

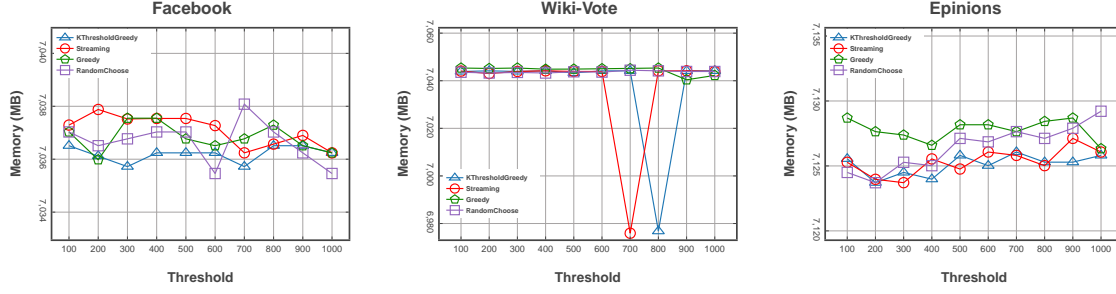
In general, the cost of memory usage differs minimally amongst algorithms. Only the difference in the size of the datasets influences the search strategy because resources are used to construct the RIS sample set size and to search (Figure 5.1(b)).

We compare the total running time of the algorithms (Figure 5.1(c)) and decide excepted that algorithms 9 and 10 take longer than algorithms 11 and 12 because it is related to the

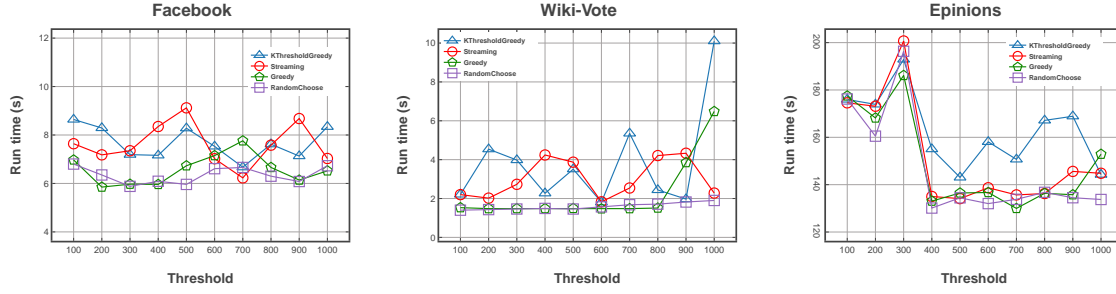
(a) Num of seed nodes comparison



(b) Memory usage comparison



(c) Total running time comparison



(d) Algorithms calculated time comparison

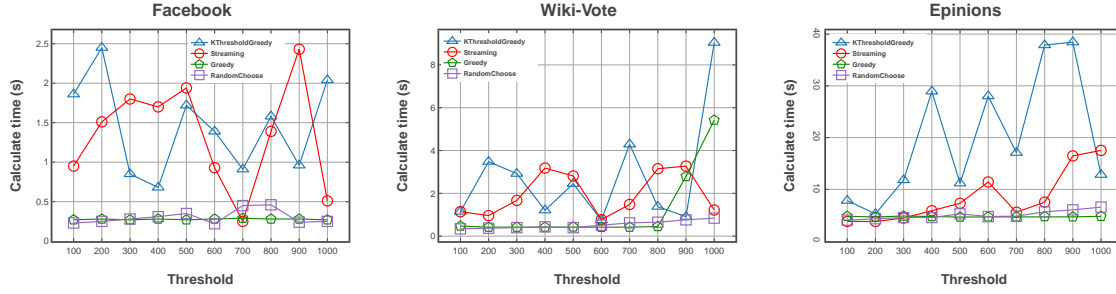


Figure 5.1: Performance of algorithms in experiment of small datasets with threshold T from 100 to 1000

two proposed algorithms' methods of multiple search solutions, with the number of solutions described in Table 5.3. Figure 5.1(d) illustrates the algorithm's search time for independent solutions, from which the time effect can be shown with the reduction of RIS sample generation

Table 5.3: Num of Solution for each algorithm based-on dataset num nodes

Dataset	KThresholdGreedy	Streaming	Greedy	RandomChoose
Facebook	37	37	1	1
Wiki-Vote	39	39	1	1
Epinions	49	49	1	1
Stanford	55	55	1	1
DBLP	58	58	1	1

time with the same number of four methods. Searching on several solutions, algorithms 9 and 10, would be many times more expensive than searching on a single solution of algorithms 11 and 12. Also, algorithm 9 employs a greedy strategy, which causes the algorithm to execute more slowly than other algorithms.

Figure 5.2 show the experimental result using the medium size node datasets, which are the Stanford 282K and DBLP 655K. While comparing our two algorithms with the others, we conclude there is not too big a difference between experiment results using small and medium datasets, where the KThresholdGreedy and the Streaming algorithms keep the smallest seed num while the RandomChoose is too high. On the other hand, their memory usage is more minor when compared with the Greedy Algorithm. Finally, the num of seed node that KThresholdGreedy return is usually lower than Streaming, while most of the running time and the method calculation time is higher than the Streaming.

5.4.5 Overview experimental results

Algorithms 9 and 10 outperform the other two analyzed algorithms regarding seed-cost search efficiency. With the approach combined with greedy, the KThresholdGreedy algorithm has a longer running time than Streaming algorithm, but the number of seed nodes found is often smaller than the number found by algorithm 10. The main algorithm 10 has been optimized with the goal of improving the search time that algorithm 9 has not, with the search time being reduced 2 to 40 times more effectively than algorithm 9 without taking the time cost of RIS prototyping and being 1.2 to 5 times more efficient when comparing total run times. Algorithms 9 and 10 propose a solution to help determine the distribution of seeds on K topics more effectively than Greedy and Random Choose algorithms based on experiments. Given the appropriate number of RIS samples and the threshold T within the searchable range, the running times of algorithms 9 and 10 are not significantly different from the other two compared algorithms.

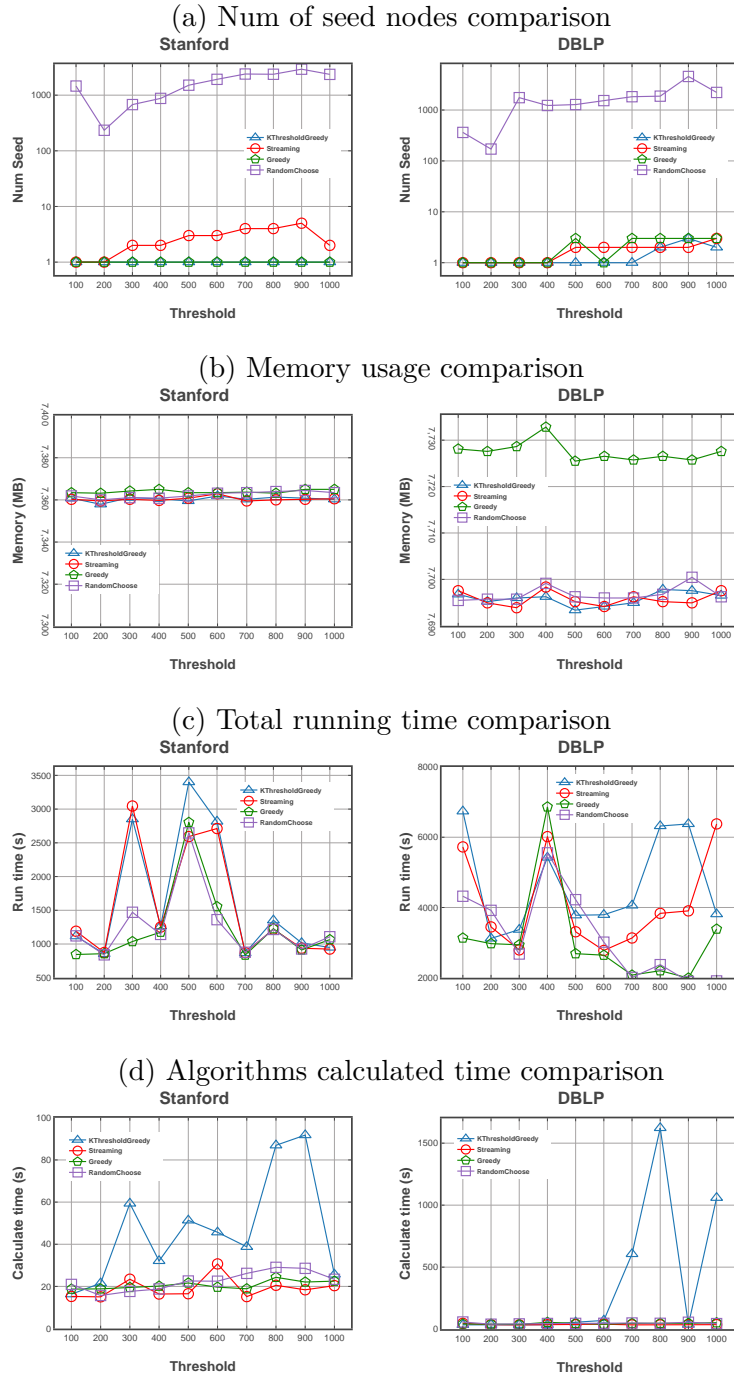


Figure 5.2: Performance of algorithms in experiment of medium datasets with threshold T from 100 to 1000

5.5 Discussion

In this chapter, we introduce an overview submodular and k -submodular function problem. Next, we investigate an efficient streaming k -submodular function maximization, named k SC, in which applied on Influence Maximization with k topics. Finally, we extensive experiments to conduct on some datasets in social network that show that the performance of our proposed algorithm with the other.

Chapter 6

Conclusion and Future work

This section summarizes the main goals and contributions of the dissertation in section 6.1. Moreover, the last section presents the IM problem in section 6.2, which focuses on future research with more challenges.

6.1 Summary of results

Information diffusion in social networks has been studied with much attention from many researchers in the last decade. In particular, Influence Maximization plays an essential role in the field of viral marketing such as e-Commerce, co-authors networks, social recommendation, rumor control, misinformation, and others.

The main goals of this dissertation are to study the Influence Maximization problems in social networks. In this Ph.D. thesis, we propose a variety of influence maximization in OSNs, which include three major categories as Influence Maximization for Multiple-threshold, Influence Maximization for Group influence, and Influence Maximization for k -topic in the complex networks, as follows:

- Firstly, the classical IM with constraint cost, time, and threshold is developed. In addition, we propose an effective method for multiple threshold in social network based on approximation algorithms. We then carry out extensive experiments using real-world network datasets to demonstrate the effectiveness of our algorithm with other in terms of running time, complexity and memory used.
- Secondly, we present the structure of characteristics that influence the spread to group nodes on the IC and LT models. Based on that, we will propose approximation algorithms to solve group influence with theoretical guarantee in both the general case and minimum cost.

- Finally, we show the Influence Maximization with the k -topic in social networks considering constraints setting. In addition, we will propose an efficient method for maximizing the k -submodular function with constraints in which running time is low. Complexity and variants of all topics are considered challenging tasks. Moreover, we propose a streaming algorithm combined with an efficient method to improve the solution quality, running time, and performance algorithm in social networks.

6.2 Future work

To summarize, the progress research results in this Ph.D. thesis are fulfilled and we continue to study these issues with better results. Motivated by practical applications, we will study and expand the problem with highly applicable variants of Influence Maximization to bring better results than the start-of-the-art methods. Specifically, we have been developing efficient IM algorithms that provide the provable approximate guarantee and apply a new novel metric to improve the quality of solutions more effectively. In addition, effective algorithms based on clique, a community for complex networks, will be considered.

Although the dissertation achieved the proposed results, there are still future research challenges in the literature. We are going to continue to focus on many issues as follows.

- Developing an efficient algorithm to evaluate the Influence Maximization problem in dynamic social networks with theoretical guarantees. [130] [131]
- Studying the characteristics and attributes of the Context-aware Influence Maximization problem, such as topic, time and location in various contexts. [132] [133] [134]
- Investigating the structures and features of Dynamic Influence Maximization subject to a diffusion model, which considers modeling the dynamic as a sequence of snapshot graphs in a social network. [135] [136]

Bibliography

1. KEMPE, David; KLEINBERG, Jon M.; TARDOS, Éva. Maximizing the spread of influence through a social network. In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*. 2003, pp. 137–146. Available from DOI: 10.1145/956750.956769.
2. BORGS, Christian; BRAUTBAR, Michael; CHAYES, Jennifer T.; LUCIER, Brendan. Maximizing Social Influence in Nearly Optimal Time. In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*. 2014, pp. 946–957. Available from DOI: 10.1137/1.9781611973402.70.
3. CHEN, Wei; COLLINS, Alex; CUMMINGS, Rachel; KE, Te; LIU, Zhenming; RINCÓN, David; SUN, Xiaorui; WANG, Yajun; WEI, Wei; YUAN, Yifei. Influence Maximization in Social Networks When Negative Opinions May Emerge and Propagate. In: *Proceedings of the Eleventh SIAM International Conference on Data Mining, April 28-30, Mesa, Arizona, USA*. 2011, pp. 379–390. Available from DOI: 10.1137/1.9781611972818.33.
4. TANG, Youze; XIAO, Xiaokui; SHI, Yanchen. Influence maximization: near-optimal time complexity meets practical efficiency. In: *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*. 2014, pp. 75–86. Available from DOI: 10.1145/2588555.2593670.
5. TANG, Youze; SHI, Yanchen; XIAO, Xiaokui. Influence Maximization in Near-Linear Time: A Martingale Approach. In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*. 2015, pp. 1539–1554. Available from DOI: 10.1145/2723372.2723734.
6. WANG, Xiaoyang; ZHANG, Ying; ZHANG, Wenjie; LIN, Xuemin. Efficient Distance-Aware Influence Maximization in Geo-Social Networks. *IEEE Trans. Knowl. Data Eng.* 2017, vol. 29, no. 3, pp. 599–612.

7. LI, Guoliang; CHEN, Shuo; FENG, Jianhua; TAN, Kian-lee; AND, Wen-Syan Li. Efficient Location-Aware Influence Maximization. In: *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*. 2018, pp. 1569–1572.
8. HE, Xinran; SONG, Guojie; CHEN, Wei; JIANG, Qingye. Influence Blocking Maximization in Social Networks under the Competitive Linear Threshold Model. In: *Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012*. 2012, pp. 463–474.
9. ASLAY, cCigdem; BARBIERI, Nicola; BONCHI, Francesco; BAEZA-YATES, Ricardo A. Online Topic-aware Influence Maximization Queries. In: *Proceedings of the 17th International Conference on Extending Database Technology, EDBT 2014, Athens, Greece, March 24-28, 2014*. 2014, pp. 295–306.
10. NGUYEN, Hung T.; THAI, My T.; DINH, Thang N. A Billion-Scale Approximation Algorithm for Maximizing Benefit in Viral Marketing. *IEEE/ACM Transactions on Networking*. 2017, vol. 25, no. 4, pp. 2419–2429. Available from DOI: 10.1109/TNET.2017.2691544.
11. PHAM, Canh V.; DUONG, Hieu V.; HOANG, Huan X.; THAI, My T. Competitive Influence Maximization within Time and Budget Constraints in Online Social Networks: An Algorithmic Approach. *Applied Sciences*. 2019, vol. 9, no. 11.
12. CHEN, Wei; LAKSHMANAN, Laks V. S.; CASTILLO, Carlos. *Information and Influence Propagation in Social Networks*. Morgan & Claypool Publishers, 2013. Synthesis Lectures on Data Management. ISBN 9781627051156. Available from DOI: 10.2200/S00527ED1V01Y201308DTM037.
13. LESKOVEC, Jure; ADAMIC, Lada A.; HUBERMAN, Bernardo A. The dynamics of viral marketing. *ACM Trans. Web*. 2007, vol. 1, no. 1, p. 5. Available from DOI: 10.1145/1232722.1232727.
14. HINZ, Oliver; SPANN, Martin. The Impact of Information Diffusion on Bidding Behavior in Secret Reserve Price Auctions. *Information Systems Research*. 2008-09, vol. 19, pp. 351–368. Available from DOI: 10.1287/isre.1080.0190.
15. SKIERA, Bernd; BARROT, Christian; BECKER, Jan; HINZ, Oliver. Seeding Strategies for Viral Marketing: An Empirical Comparison. *Journal of Marketing*. 2011-07, vol. 75, pp. 55–71. Available from DOI: 10.1509/jm.10.0088.
16. CHEN, Wei; LU, Wei; ZHANG, Ning. Time-Critical Influence Maximization in Social Networks with Time-Delayed Diffusion Process. *CoRR*. 2012, vol. abs/1204.3074. Available from arXiv: 1204.3074.

17. CHEN, Wei; WANG, Chi; WANG, Yajun. Scalable Influence Maximization for Prevalent Viral Marketing in Large-Scale Social Networks. In: *Proceedings of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'2010)*, Washington DC, U.S.A. 2010-07.
18. CHEN, Wei; YUAN, Yifei; ZHANG, Li. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In: *2010 IEEE International Conference on Data Mining*. 2010, pp. 88–97. Available from DOI: 10.1109/ICDM.2010.118.
19. KIM, Jinha; KIM, Seung-Keol; YU, Hwanjo. Scalable and parallelizable processing of influence maximization for large-scale social networks? In: 2013-04, pp. 266–277. ISBN 978-1-4673-4909-3. Available from DOI: 10.1109/ICDE.2013.6544831.
20. BANERJEE, Suman; JENAMANI, Mamata; PRATIHAR, Dilip. ComBIM: A Community-Based Solution Approach for the Budgeted Influence Maximization Problem. *Expert Systems with Applications*. 2019-07, vol. 125. Available from DOI: 10.1016/j.eswa.2019.01.070.
21. CHEN, Yi-Cheng; ZHU, Wen-Yuan; PENG, Wen-Chih; LEE, Wang-Chien; LEE, Suh-Yin. CIM: Community-Based Influence Maximization in Social Networks. *ACM Trans. Intell. Syst. Technol.* 2014, vol. 5, no. 2, 25:1–25:31. Available from DOI: 10.1145/2532549.
22. ZHANG, Xiaohang; ZHU, Ji; WANG, Qi; ZHAO, Han. Identifying influential nodes in complex networks with community structure. *Knowledge-Based Systems*. 2013-04, vol. 42, pp. 74–84. Available from DOI: 10.1016/j.knosys.2013.01.017.
23. GONG, Maoguo; YAN, Jianan; SHEN, Bo; LIJIA, Ma; CAI, Qing. Influence Maximization in Social Networks Based on Discrete Particle Swarm Optimization. *Information Sciences*. 2016-07, vol. 367. Available from DOI: 10.1016/j.ins.2016.07.012.
24. ZHANG, Kaiqi; DU, Haifeng; FELDMAN, Marcus. Maximizing influence in a social network: Improved results using a genetic algorithm. *Physica A: Statistical Mechanics and its Applications*. 2017-02, vol. 478. Available from DOI: 10.1016/j.physa.2017.02.067.
25. BUCUR, Doina; IACCA, Giovanni. Influence Maximization in Social Networks with Genetic Algorithms. In: 2016-03, pp. 379–392. ISBN 978-3-319-31203-3. Available from DOI: 10.1007/978-3-319-31204-0_25.
26. KRÖMER, Pavel; NOWAKOVA, Jana. Guided Genetic Algorithm for the Influence Maximization Problem. In: 2017-07, pp. 630–641. ISBN 978-3-319-62388-7. Available from DOI: 10.1007/978-3-319-62389-4_52.

27. CHEN, Wei; YUAN, Yifei; ZHANG, Li. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In: *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*. 2010, pp. 88–97. Available from DOI: 10.1109/ICDM.2010.118.
28. PHAM, Canh V.; DUONG, Hieu V.; THAI, My T. Importance Sample-Based Approximation Algorithm for Cost-Aware Targeted Viral Marketing. In: *Computational Data and Social Networks - 8th International Conference, Ho Chi Minh City, Vietnam, November 18-20, 2019*. 2019, pp. 120–132. Available from DOI: 10.1007/978-3-030-34980-6_14.
29. TANG, Jing; TANG, Xueyan; YUAN, Junsong. Profit Maximization for Viral Marketing in Online Social Networks: Algorithms and Analysis. *IEEE Trans. Knowl. Data Eng.* 2018, vol. 30, no. 6, pp. 1095–1108. Available from DOI: 10.1109/TKDE.2017.2787757.
30. GUO, Jianxiong; WU, Weili. Continuous Profit Maximization: A Study of Unconstrained Dr-Submodular Maximization. *IEEE Trans. Comput. Soc. Syst.* 2021, vol. 8, no. 3, pp. 768–779. Available from DOI: 10.1109/TCSS.2021.3061452.
31. NGUYEN, Hung T.; THAI, My T.; DINH, Thang N. Stop-and-Stare: Optimal Sampling Algorithms for Viral Marketing in Billion-scale Networks. In: *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*. 2016, pp. 695–710.
32. BOZORGI, Arastoo; SAMET, Saeed; KWISTHOUT, Johan; WAREHAM, Todd. Community-based influence maximization in social networks under a competitive linear threshold model. *Knowl.-Based Syst.* 2017, vol. 134, pp. 149–158.
33. GOYAL, Amit; BONCHI, Francesco; LAKSHMANAN, Laks V. S.; VENKATASUBRAMANIAN, Suresh. On minimizing budget and time in influence propagation over social networks. *Social Network Analysis and Mining*. 2013, vol. 3, pp. 179–192. Available from DOI: 10.1007/s13278-012-0062-z.
34. GIMENES, Gabriel P.; GUALDRON, Hugo; RADDI, Thiago R.; JR., José Fernando Rodrigues. Supervised-learning link recommendation in the DBLP co-authoring network. In: *2014 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom 2014 Workshops, Budapest, Hungary, March 24-28, 2014*. IEEE Computer Society, 2014, pp. 563–568. Available from DOI: 10.1109/PerComW.2014.6815268.
35. YANG, Jin-Xuan; ZHANG, Xiao-Dong. Revealing how network structure affects accuracy of link prediction. *European Physical Journal B*. 2017-08, vol. 90. Available from DOI: 10.1140/epjb/e2017-70599-4.

36. CHO, Eunjoon; MYERS, Seth A.; LESKOVEC, Jure. Friendship and mobility: user movement in location-based social networks. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011, pp. 1082–1090. Available from DOI: 10.1145/2020408.2020579.
37. LI, Cheng-Te; HUANG, Mei-Yuan; YAN, Rui. Team formation with influence maximization for influential event organization on social networks. *World Wide Web*. 2018, vol. 21, no. 4, pp. 939–959. Available from DOI: 10.1007/s11280-017-0492-7.
38. RAWASHDEH, Ahmad; RAWASHDEH, Mohammad; DIAZ, Irene; RALESCU, Anca. Measures of Semantic Similarity of Nodes in a Social Network. In: LAURENT, Anne; STRAUSS, Olivier; BOUCHON-MEUNIER, Bernadette; YAGER, Ronald R. (eds.). *Information Processing and Management of Uncertainty in Knowledge-Based Systems - 15th International Conference, IPMU 2014, Montpellier, France, July 15-19, 2014, Proceedings, Part II*. Springer, 2014, vol. 443, pp. 76–85. Communications in Computer and Information Science. Available from DOI: 10.1007/978-3-319-08855-6_9.
39. CHEN, Xuanhao; DENG, Liwei; ZHAO, Yan; ZHOU, Xiaofang; ZHENG, Kai. Community-based influence maximization in location-based social network. *World Wide Web*. 2021, vol. 24, no. 6, pp. 1903–1928. Available from DOI: 10.1007/s11280-021-00935-x.
40. PHAM, Canh V.; PHU, Quat V.; HOANG, Huan X.; PEI, Jun; THAI, My T. Minimum budget for misinformation blocking in onlinesocial networks. *Combinatorial Optimization*. 2019, vol. 38, no. 4, pp. 1101–1127. Available also from: <https://doi.org/10.1007/s10878-019-00439-5>.
41. SHEN, Yilin; DINH, Thang N.; ZHANG, Huiyuan; THAI, My T. Interest-matching information propagation in multiple online social networks. In: CHEN, Xue-wen; LEBANON, Guy; WANG, Haixun; ZAKI, Mohammed J. (eds.). *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*. ACM, 2012, pp. 1824–1828. Available from DOI: 10.1145/2396761.2398525.
42. NGUYEN, Hung T.; GHOSH, Preetam; MAYO, Michael L.; DINH, Thang N. Multiple Infection Sources Identification with Provable Guarantees. In: MUKHOPADHYAY, Snehasis; ZHAI, ChengXiang; BERTINO, Elisa; CRESTANI, Fabio; MOSTAFA, Javed; TANG, Jie; SI, Luo; ZHOU, Xiaofang; CHANG, Yi; LI, Yunyao; SONDHAI, Parikshit (eds.). *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*. ACM, 2016, pp. 1663–1672. Available from DOI: 10.1145/2983323.2983817.

43. PHAM, Canh V.; PHU, Quat V.; HOANG, Huan X.; PEI, Jun; THAI, My T. Minimum budget for misinformation blocking in online social networks. *J. Comb. Optim.* 2019, vol. 38, no. 4, pp. 1101–1127.
44. LESKOVEC, Jure; KRAUSE, Andreas; GUESTSTRIN, Carlos; FALOUTSOS, Christos; VANBRIESEN, Jeanne M.; GLANCE, Natalie S. Cost-effective outbreak detection in networks. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*. 2007, pp. 420–429. Available from DOI: 10.1145/1281192.1281239.
45. GOYAL, Amit; LU, Wei; LAKSHMANAN, Laks V.S. CELF++: Optimizing the Greedy Algorithm for Influence Maximization in Social Networks. In: *Proceedings of the 20th International Conference on World Wide Web*. 2011, pp. 47–48.
46. CHEN, Wei. An Issue in the Martingale Analysis of the Influence Maximization Algorithm IMM. In: CHEN, Xuemin; SEN, Arunabha; LI, Wei Wayne; THAI, My T. (eds.). *Computational Data and Social Networks - 7th International Conference, CSoNet 2018, Shanghai, China, December 18-20, 2018, Proceedings*. Springer, 2018, vol. 11280, pp. 286–297. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-04648-4_24.
47. BORODIN, Allan; FILMUS, Yuval; OREN, Joel. Threshold Models for Competitive Influence in Social Networks. In: SABERI, Amin (ed.). *Internet and Network Economics - 6th International Workshop, WINE 2010, Stanford, CA, USA, December 13-17, 2010. Proceedings*. Springer, 2010, vol. 6484, pp. 539–550. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-642-17572-5_48.
48. NGUYEN, Hung T.; NGUYEN, Tri P.; PHAN, NhatHai; DINH, Thang N. Importance Sketching of Influence Dynamics in Billion-Scale Networks. In: *2017 International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*. 2017, pp. 337–346. Available from DOI: 10.1109/ICDM.2017.43.
49. LI, Xiang; SMITH, J. David; DINH, Thang N.; THAI, My T. TipTop: (Almost) Exact Solutions for Influence Maximization in Billion-Scale Networks. *IEEE/ACM Trans. Netw.* 2019, vol. 27, no. 2, pp. 649–661.
50. DAI, Yajun; JIANG, Wenjun; LI, Kenli. Group-Based Competitive Influence Maximization. In: *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/ SCALCOM/ UIC/ ATC/ CBDCOM/ IOP/ SCI)*. 2009, pp. 199–208. Available from DOI: 10.1109/SmartWorld.2018.00176.

51. KIANIAN, Zahra Aghaee Sahar. Influence maximization algorithm based on reducing search space in the social networks. *SN Applied Sciences*. 2020, vol. 2, no. 2067.
52. ZHU, Jianming; GHOSH, Smita; WU, Weili. Group Influence Maximization Problem in Social Networks. *IEEE Trans. Comput. Social Systems*. 2019, vol. 6, no. 6, pp. 1156–1164.
53. BOZORGI, Arastoo; HAGHIGHI, Hassan; ZAHEDI, Mohammad Sadegh; REZVANI, Mojtaba. INCIM: A community-based algorithm for influence maximization problem under the linear threshold model. *Information Processing Management*. 2016, vol. 52, pp. 1188–1199. Available from DOI: 10.1016/j.ipm.2016.05.006.
54. BOUYER, Hamid Ahmadi Beni Asgarali. TI-SC: top-k influential nodes selection based on community detection and scoring criteria in social networks. *Journal of Ambient Intelligence and Humanized Computing*. 2020, vol. 11, pp. 4889–4908. Available from DOI: 10.1007/s12652-020-01760-2.
55. ZHU, Jianming; GHOSH, Smita; WU, Weili; GAO, Chuangen. Profit Maximization Under Group Influence Model in Social Networks. In: *Computational Data and Social Networks*. International Conference on Computational Data and Social Networks CSoNet 2019, 2019, pp. 108–119.
56. ZHONG, Yuting; GUO, Longkun; HUANG, Peihuang. Maximizing Group Coverage in Social Networks. In: *International Conference on Parallel and Distributed Computing: Applications and Technologies PDCAT 2020: Parallel and Distributed Computing, Applications and Technologies*. 2021, pp. 274–284.
57. LI, Yuchen; ZHANG, Dongxiang; TAN, Kian-Lee. Targeted Influence Maximization for Online Advertisements. *PVLDB*. 2015, vol. 8, no. 10, pp. 1070–1081.
58. CHEN, Shuo; FAN, Ju; LI, Guoliang; FENG, Jianhua; TAN, Kian-Lee; TANG, Jinhui. Online Topic-Aware Influence Maximization. *Proc. VLDB Endow*. 2015, vol. 8, no. 6, pp. 666–677. Available from DOI: 10.14778/2735703.2735706.
59. HUBER, Anna; KOLMOGOROV, Vladimir. Towards Minimizing k-Submodular Functions. In: MAHJOUB, Ali Ridha; MARKAKIS, Vangelis; MILIS, Ioannis; PASCHOS, Vangelis Th. (eds.). *Combinatorial Optimization - Second International Symposium, ISCO 2012, Athens, Greece, April 19-21, 2012, Revised Selected Papers*. Springer, 2012, vol. 7422, pp. 451–462. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-642-32147-4_40.
60. SINGH, Ajit; GUILLORY, Andrew; BILMES, Jeff. On Bisubmodular Maximization. In: LAWRENCE, Neil D.; GIROLAMI, Mark (eds.). *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*. La Palma, Canary Islands:

- PMLR, 2012-21-23 Apr, vol. 22, pp. 1055-1063. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v22/singh12.html>.
61. FUJISHIGE, Satoru; IWATA, Satoru. Bisubmodular Function Minimization. *SIAM Journal on Discrete Mathematics*. 2005, vol. 19, no. 4, pp. 1065-1073. Available from DOI: 10.1137/S0895480103426339.
 62. IWATA, Satoru. Submodular function minimization. *Math. Program.* 2008, vol. 112, no. 1, pp. 45-64. Available from DOI: 10.1007/s10107-006-0084-2.
 63. FUJISHIGE, Satoru; TANIGAWA, Shin-ichi. Polynomial combinatorial algorithms for skew-bisubmodular function minimization. *Math. Program.* 2018, vol. 171, no. 1-2, pp. 87-114. Available from DOI: 10.1007/s10107-017-1171-2.
 64. FUJISHIGE, Satoru; KIRÁLY, Tamás; MAKINO, Kazuhisa; TAKAZAWA, Kenjiro; TANIGAWA, Shin-ichi. Minimizing submodular functions on diamonds via generalized fractional matroid matchings. *Journal of Combinatorial Theory, Series B*. 2022, vol. 157, pp. 294-345. ISSN 0095-8956. Available from DOI: <https://doi.org/10.1016/j.jctb.2022.07.005>.
 65. SOMA, Tasuku. No-regret algorithms for online k -submodular maximization. In: CHAUDHURI, Kamalika; SUGIYAMA, Masashi (eds.). *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. PMLR, 2019-16-18 Apr, vol. 89, pp. 1205-1214. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v89/soma19a.html>.
 66. YU, Qimeng; KÜÇÜKYAVUZ, Simge. An exact cutting plane method for k -submodular function maximization. *Discrete Optimization*. 2021, vol. 42, p. 100670. ISSN 1572-5286. Available from DOI: <https://doi.org/10.1016/j.disopt.2021.100670>.
 67. MATSUOKA, Tatsuya; OHSAKA, Naoto. Maximization of Monotone k -Submodular Functions with Bounded Curvature and Non- k -Submodular Functions. In: BALASUBRAMANIAN, Vineeth N.; TSANG, Ivor (eds.). *Proceedings of The 13th Asian Conference on Machine Learning*. PMLR, 2021-17-19 Nov, vol. 157, pp. 1707-1722. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v157/matsuoka21b.html>.
 68. THAPPER, Johan; IVNÝ, Stanislav. The Power of Linear Programming for Valued CSPs. In: *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*. 2012, pp. 669-678. Available from DOI: 10.1109/FOCS.2012.25.
 69. OHSAKA, Naoto; YOSHIDA, Yuichi. Monotone k -Submodular Function Maximization with Size Constraints. In: CORTES, Corinna; LAWRENCE, Neil D.; LEE, Daniel D.; SUGIYAMA, Masashi; GARNETT, Roman (eds.). *Advances in Neural Information*

Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada. 2015, pp. 694–702.

70. SAKAUE, Shinsaku. On maximizing a monotone k -submodular function subject to a matroid constraint. *Discrete Optimization*. 2017, vol. 23, pp. 105–113. ISSN 1572-5286. Available from DOI: <https://doi.org/10.1016/j.disopt.2017.01.003>.
71. QIAN, Chao; SHI, Jing-Cheng; TANG, Ke; ZHOU, Zhi-Hua. Constrained Monotone k -Submodular Function Maximization Using Multiobjective Evolutionary Algorithms With Theoretical Guarantee. *IEEE Transactions on Evolutionary Computation*. 2018, vol. 22, no. 4, pp. 595–608. Available from DOI: 10.1109/TEVC.2017.2749263.
72. RAFIEY, Akbar; YOSHIDA, Yuichi. Fast and Private Submodular and k -Submodular Functions Maximization with Matroid Constraints. In: III, Hal Daumé; SINGH, Aarti (eds.). *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020-13–18 Jul, vol. 119, pp. 7887–7897. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v119/rafiey20a.html>.
73. NGUYEN, Lan; THAI, My T. Streaming k -Submodular Maximization under Noise subject to Size Constraint. In: III, Hal Daumé; SINGH, Aarti (eds.). *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020-13–18 Jul, vol. 119, pp. 7338–7347. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v119/nguyen20f.html>.
74. ZHENG, Leqian; CHAN, Hau; LOUKIDES, Grigorios; LI, Minming. Maximizing Approximately k -Submodular Functions. In: DEMENICONI, Carlotta; DAVIDSON, Ian (eds.). *Proceedings of the 2021 SIAM International Conference on Data Mining, SDM 2021, Virtual Event, April 29 - May 1, 2021*. SIAM, 2021, pp. 414–422. Available from DOI: 10.1137/1.9781611976700.47.
75. WARD, Justin; ŽIVNÝ, Stanislav. Maximizing Bisubmodular and k -Submodular Functions. In: *Proceedings of the 2014 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2014, pp. 1468–1481. Available from DOI: 10.1137/1.9781611973402.108.
76. IWATA, Satoru; TANIGAWA, Shin-ichi; YOSHIDA, Yuichi. Improved Approximation Algorithms for k -Submodular Function Maximization. In: KRAUTHGAMER, Robert (ed.). *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*. SIAM, 2016, pp. 404–413.
77. OSHIMA, Hiroki. Derandomization for k -Submodular Maximization. In: BRANKOVIC, Ljiljana; RYAN, Joe; SMYTH, William F. (eds.). *In Proc. of International Workshop Combinatorial Algorithms (IWOCA)*. 2017, pp. 88–99.

78. CHAKRABARTI, Amit; KALE, Sagar. Submodular maximization meets streaming: matchings, matroids, and more. *Math. Program.* 2015, vol. 154, no. 1-2, pp. 225–247. Available from DOI: 10.1007/s10107-015-0900-7.
79. HUANG, Chien-Chung; KAKIMURA, Naonori; YOSHIDA, Yuichi. Streaming Algorithms for Maximizing Monotone Submodular Functions Under a Knapsack Constraint. *Algorithmica.* 2020, vol. 82, no. 4, pp. 1006–1032. Available from DOI: 10.1007/s00453-019-00628-y.
80. WANG, Baoxiang; ZHOU, Huanjian. Multilinear extension of k-submodular functions. *CoRR.* 2021, vol. abs/2107.07103. Available from arXiv: 2107.07103.
81. TANG, Zhongzheng; WANG, Chenhao; CHAN, Hau. On maximizing a monotone k-submodular function under a knapsack constraint. *CoRR.* 2021, vol. abs/2105.15159. Available from arXiv: 2105.15159.
82. SVIRIDENKO, Maxim. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters.* 2004, vol. 32, no. 1, pp. 41–43. ISSN 0167-6377. Available from DOI: [https://doi.org/10.1016/S0167-6377\(03\)00062-2](https://doi.org/10.1016/S0167-6377(03)00062-2).
83. BADANIDIYURU, Ashwinkumar; MIRZASOLEIMAN, Baharan; KARBASI, Amin; KRAUSE, Andreas. Streaming submodular maximization: massive data summarization on the fly. In: MACSKASSY, Sofus A.; PERLICH, Claudia; LESKOVEC, Jure; WANG, Wei; GHANI, Rayid (eds.). *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. ACM, 2014, pp. 671–680. Available from DOI: 10.1145/2623330.2623637.
84. YANG, Ruiqi; XU, Dachuan; CHENG, Yukun; GAO, Chuangen; DU, Ding-Zhu. Streaming submodular maximization under noises. In: *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 348–357.
85. ENE, Alina; NGUYEN, Huy. Streaming Algorithm for Monotone k-Submodular Maximization with Cardinality Constraints. In: CHAUDHURI, Kamalika; JEGELKA, Stefanie; SONG, Le; SZEPESVARI, Csaba; NIU, Gang; SABATO, Sivan (eds.). *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 2022-17–23 Jul, vol. 162, pp. 5944–5967. Proceedings of Machine Learning Research. Available also from: <https://proceedings.mlr.press/v162/ene22a.html>.
86. ZHANG, Yuhui; LI, Ming; YANG, Dejun; XUE, Guoliang. A Budget Feasible Mechanism for k-Topic Influence Maximization in Social Networks. In: *2019 IEEE Global Communications Conference (GLOBECOM)*. 2019, pp. 1–6. Available from DOI: 10.1109/GLOBECOM38437.2019.9013859.

87. FELDMAN, Moran; NOROUZI-FARD, Ashkan; SVENSSON, Ola; ZENKLUSEN, Rico. Streaming Submodular Maximization with Matroid and Matching Constraints. *CoRR*. 2021, vol. abs/2107.07183. Available from arXiv: 2107.07183.
88. PHAM, Canh V.; VU, Quang C.; HA, Dung K.; NGUYEN, Tai T.; LE, Nguyen D. Maximizing k-submodular functions under budget constraint: applications and streaming algorithms. *J. Comb. Optim.* 2022, vol. 44, no. 1, pp. 723–751.
89. HA, Dung TK; PHAM, Canh V; HOANG, Huan X. Submodular Maximization Subject to a Knapsack Constraint Under Noise Models. *Asia-Pacific Journal of Operational Research*. 2022, p. 2250013.
90. KUHNLE, Alan; PAN, Tianyi; ALIM, Md Abdul; THAI, My T. Scalable Bicriteria Algorithms for the Threshold Activation Problem in Online Social Networks. In: *IEEE Conference on Computer Communications*. 2017. Available from DOI: 10.1109/INFOCOM.2017.8057068.
91. PHAM, Canh V.; DUONG, Hieu V.; BUI, Bao Q. Budgeted Competitive Influence Maximization on Online Social Networks. In: *Computational Data and Social Networks - 7th International Conference, CSoNet 2018, Shanghai, China, December 18-20, 2018, Proceedings*. Springer, 2018, vol. 11280, pp. 13–24. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-04648-4_2.
92. PHAM, Canh V.; THAI, My T.; HA, Dung K.; NGO, Dung Q.; HOANG, Huan X. Time-Critical Viral Marketing Strategy with the Competition on Online Social Networks. In: NGUYEN, Hien T.; SNÁSEL, Václav (eds.). *Computational Social Networks - 5th International Conference, CSoNet 2016, Ho Chi Minh City, Vietnam, August 2-4, 2016, Proceedings*. Springer, 2016, vol. 9795, pp. 111–122. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-319-42345-6_10.
93. PHAM, Canh V.; DINH, Hoang M.; NGUYEN, Hoa D.; XUAN, Huan Hoang; DANG, Huyen T. Limiting the Spread of Epidemics within Time Constraint on Online Social Networks. In: *Proceeding of the eight international symposium on information and communication technology, Nha Trang City, Viet Nam, December 7-8, 2017*. 2017, pp. 262–269. Available from DOI: 10.1145/3155133.3155157.
94. BUDAK, Ceren; AGRAWAL, Divyakant; EL ABBADI, Amr. Limiting the spread of misinformation in social networks. In: *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*. 2011, pp. 665–674. Available from DOI: 10.1145/1963405.1963499.
95. ZHANG, Huiling; ALIM, Md Abdul; LI, Xiang; THAI, My T.; NGUYEN, Hien T. Misinformation in Online Social Networks: Detect Them All with a Limited Budget.

- ACM Transactions on Information Systems*. 2016, vol. 34, no. 3, pp. 1–24. Available from DOI: 10.1145/2885494.
96. PHAM, Canh V.; PHAM, Dung V.; BUI, Bao Q.; NGUYEN, Anh V. Minimum budget for misinformation detection in online social networks with provable guarantees. *Optimization Letters*. 2021, pp. 1–20.
 97. GOYAL, Amit; LU, Wei; LAKSHMANAN, Laks V.S. Simpath: An Efficient Algorithm for InfluenceMaximization under the Linear Threshold Model. In: *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, pp. 211–220. Available from DOI: 10.1109/ICDM.2011.132.
 98. CRAWFORD, Victoria G.; KUHNLE, Alan; THAI, My T. Submodular Cost Submodular Cover with an Approximate Oracle. In: CHAUDHURI, Kamalika; SALAKHUTDINOV, Ruslan (eds.). *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. PMLR, 2019, vol. 97, pp. 1426–1435. Proceedings of Machine Learning Research. Available also from: <http://proceedings.mlr.press/v97/crawford19a.html>.
 99. PHAM, Phuong N. H.; NGUYEN, Bich-Ngan T.; PHAM, Canh V.; NGHIA, Nghia D.; SNÁSEL, Václav. Efficient Algorithm for Multiple Benefit Thresholds Problem in Online Social Networks. In: *RIVF International Conference on Computing and Communication Technologies, RIVF 2021, Hanoi, Vietnam, August 19-21, 2021*. IEEE, 2021, pp. 1–6. Available from DOI: 10.1109/RIVF51545.2021.9642099.
 100. CHUNG, Fan R. K.; LU, Lincoln. Survey: Concentration Inequalities and Martingale Inequalities: A Survey. *Internet Mathematics*. 2006, vol. 3, no. 1, pp. 79–127.
 101. SACHDEVA, Sushant; VISHNOI, Nisheeth K. Approximation Theory and the Design of Fast Algorithms. *CoRR*. 2013, vol. abs/1309.4882. Available from arXiv: 1309.4882.
 102. LESKOVEC, Jure; KLEINBERG, Jon M.; FALOUTSOS, Christos. Graph evolution: Densification and shrinking diameters. *TKDD*. 2007, vol. 1, no. 1, p. 2. Available from DOI: 10.1145/1217299.1217301.
 103. LESKOVEC, Jure; LANG, Kevin J.; DASGUPTA, Anirban; MAHONEY, Michael W. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Internet Mathematics*. 2009, vol. 6, no. 1, pp. 29–123. Available from DOI: 10.1080/15427951.2009.10129177.
 104. CHEN, Wei; WANG, Yajun; YANG, Siyu. Efficient influence maximization in social networks. In: *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2009, pp. 199–208. Available from DOI: 10.1145/1557019.1557047.


105. LESKOVEC, Jure; ADAMIC, Lada A.; HUBERMAN, Bernardo A. From Competition to Complementarity: Comparative Influence Diffusion and Maximization. *CoRR*. 2015, vol. abs/1507.00317. Available from arXiv: 1507.00317.
106. YANG, Jaewon; LESKOVEC, Jure. Defining and Evaluating Network Communities based on Ground-truth. *CoRR*. 2012, vol. abs/1205.6233. Available from arXiv: 1205.6233.
107. PHAM, Canh V.; THAI, My T.; DUONG, Hieu V.; BUI, Bao Q.; HOANG, Huan X. Maximizing misinformation restriction within time and budget constraints. *J. Comb. Optim.* 2018, vol. 35, no. 4, pp. 1202–1240.
108. NGUYEN, H. T.; CANO, A.; TAM, V.; DINH, T. N. Blocking Self-avoiding Walks Stops Cyber-epidemics: A Scalable GPU-based Approach. *IEEE Transactions on Knowledge and Data Engineering*. 2019, pp. 1–1.
109. ZHANG, Huiling; KUHNLE, Alan; ZHANG, Huiyuan; THAI, My T. Detecting misinformation in online social networks before it is too late. In: *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016, San Francisco, CA, USA, August 18-21, 2016*. 2016, pp. 541–548.
110. YE, Mao; LIU, Xingjie; LEE, Wang-Chien. Exploring social influence for recommendation: a generative model approach. In: *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*. 2012, pp. 671–680.
111. NGUYEN, Lan N.; ZHOU, Kunxiao; THAI, My T. Influence Maximization at Community Level: A New Challenge with Non-submodularity. In: *39th IEEE International Conference on Distributed Computing Systems, ICDCS 2019, Dallas, TX, USA, July 7-10, 2019*. 2019, pp. 327–337.
112. TSANG, Alan; WILDER, Bryan; RICE, Eric; TAMBE, Milind; ZICK, Yair. Group-Fairness in Influence Maximization. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. 2019, pp. 5997–6005.
113. FARNADI, Golnoosh; BABAKI, Behrouz; GENDREAU, Michel. A Unifying Framework for Fairness-Aware Influence Maximization. In: *Companion of The 2020 Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*. 2020, pp. 714–722.
114. TANG, Jing; TANG, Xueyan; XIAO, Xiaokui; YUAN, Junsong. Online Processing Algorithms for Influence Maximization. In: *Proceedings of the 2018 International Conference on Management of Data*. Houston, TX, USA: Association for Computing Machinery, 2018, pp. 991–1005. SIGMOD '18. ISBN 9781450347037.

115. YIN, Hao; BENSON, Austin R.; LESKOVEC, Jure; GLEICH, David F. Local Higher-Order Graph Clustering. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. 2017, pp. 555–564.
116. LESKOVEC, Jure; HUTTENLOCHER, Daniel P.; KLEINBERG, Jon M. Signed networks in social media. In: *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Atlanta, Georgia, USA, April 10-15, 2010*. 2010, pp. 1361–1370.
117. LESKOVEC, J.; KREVL, A. SNAP Datasets: Stanford large network dataset collection. 2014. Available also from: <http://snap.stanford.edu/data>.
118. RICHARDSON, Matthew; AGRAWAL, Rakesh; DOMINGOS, Pedro M. Trust Management for the Semantic Web. In: FENSEL, Dieter; SYCARA, Katia P.; MYLOPOULOS, John (eds.). *The Semantic Web - ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003, Proceedings*. Springer, 2003, vol. 2870, pp. 351–368. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-540-39718-2_23.
119. TAKAC, Lubos; ZABOVSKY, Michal. Data analysis in public social networks. In: *International scientific conference and international workshop present day trends of innovations*. Present Day Trends of Innovations Lamza Poland, 2012, vol. 1. No. 6.
120. NGUYEN, Huy; ZHENG, Rong. On Budgeted Influence Maximization in Social Networks. *IEEE J. Sel. Areas Commun.* 2013, vol. 31, no. 6, pp. 1084–1094.
121. HAN, K.; HE, Y.; HUANG, K.; XIAO, X.; TANG, S.; XU, J.; HUANG, L. Best Bang for the Buck: Cost-Effective Seed Selection for Online Social Networks. *IEEE Transactions on Knowledge and Data Engineering*. 2019, pp. 1–1.
122. DAGUM, Paul; KARP, Richard M.; LUBY, Michael; ROSS, Sheldon M. An Optimal Algorithm for Monte Carlo Estimation. *SIAM J. Comput.* 2000, vol. 29, no. 5, pp. 1484–1496.
123. KRAUSE, Andreas; GOLOVIN, Daniel. Submodular function maximization. *Tractability*. 2014, vol. 3, pp. 71–104.
124. RAFIEY, Akbar; YOSHIDA, Yuichi. Fast and Private Submodular and k-Submodular Functions Maximization with Matroid Constraints. In: *In Proc. of the International Conference on Machine Learning (ICML)*. 2020, pp. 7887–7897.
125. QIAN, Chao; SHI, Jing-Cheng; TANG, Ke; ZHOU, Zhi-Hua. Constrained Monotone k-Submodular Function Maximization Using Multiobjective Evolutionary Algorithms With Theoretical Guarantee. *IEEE Trans. Evol. Comput.* 2018, vol. 22, no. 4, pp. 595–608.

126. GOMES, Ryan; KRAUSE, Andreas. Budgeted nonparametric learning from data streams. In: *ICML*. 2010.
127. KUMAR, Ravi; MOSELEY, Benjamin; VASSILVITSKII, Sergei; VATTANI, Andrea. Fast greedy algorithms in mapreduce and streaming. *ACM Transactions on Parallel Computing (TOPC)*. 2015, vol. 2, no. 3, pp. 1–22.
128. NOROUZI-FARD, Ashkan; TARNAWSKI, Jakub; MITROVIC, Slobodan; ZANDIEH, Amir; MOUSAVIFAR, Aidasadat; SVENSSON, Ola. Beyond $1/2$ -approximation for submodular maximization on massive data streams. In: *International Conference on Machine Learning*. PMLR, 2018, pp. 3829–3838.
129. MCAULEY, Julian J.; LESKOVEC, Jure. Learning to Discover Social Circles in Ego Networks. In: BARTLETT, Peter L.; PEREIRA, Fernando C. N.; BURGESS, Christopher J. C.; BOTTOU, Léon; WEINBERGER, Kilian Q. (eds.). *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. 2012, pp. 548–556. Available also from: <https://proceedings.neurips.cc/paper/2012/hash/7a614fd06c325499f1680b9896beedeb-Abstract.html>.
130. BANERJEE, Suman; JENAMANI, Mamata; PRATIHAR, Dilip Kumar. A survey on influence maximization in a social network. *Knowledge and Information Systems*. 2020, vol. 62, no. 9, pp. 3417–3455.
131. LI, Yuchen; FAN, Ju; WANG, Yanhao; TAN, Kian-Lee. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*. 2018, vol. 30, no. 10, pp. 1852–1872.
132. CHEN, Wei; LIN, Tian; YANG, Cheng. Real-time topic-aware influence maximization using preprocessing. *Computational social networks*. 2016, vol. 3, no. 1, pp. 1–19.
133. WANG, Xiaoyang; ZHANG, Ying; ZHANG, Wenjie; LIN, Xuemin. Distance-aware influence maximization in geo-social network. In: *ICDE*. 2016, pp. 1–12.
134. GOMEZ-RODRIGUEZ, Manuel; SONG, Le; DU, Nan; ZHA, Hongyuan; SCHÖLKOPF, Bernhard. Influence estimation and maximization in continuous-time diffusion networks. *ACM Transactions on Information Systems (TOIS)*. 2016, vol. 34, no. 2, pp. 1–33.
135. CHEN, Xiaodong; SONG, Guojie; HE, Xinran; XIE, Kunqing. On influential nodes tracking in dynamic social networks. In: *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 2015, pp. 613–621.

136. TONG, Guangmo; WU, Weili; TANG, Shaojie; DU, Ding-Zhu. Adaptive influence maximization in dynamic social networks. *IEEE/ACM Transactions on Networking*. 2016, vol. 25, no. 1, pp. 112–125.

List of own publication activities

Phuong N.H. Pham ( <https://orcid.org/0000-0002-7041-9531>)

Publications and Outcomes Related to Thesis

1. **PHAM, Phuong N.H.**; NGUYEN, Bich-Ngan T.; CO, Quy T.N.; SNÁŠEL, Václav. Multiple Benefit Thresholds Problem in Online Social Networks: An Algorithmic Approach. *Mathematics, SCIE, Q1, IF: 2.592*. 2022-03, vol. 10, p. 876. Available from DOI: 10.3390/math10060876.
2. **PHAM, Phuong N.H.**; NGUYEN, Bich-Ngan T.; PHAM, Canh V.; NGHIA, Nghia D.; SNÁŠEL, Václav. Efficient Algorithm for Multiple Benefit Thresholds Problem in Online Social Networks. In: *RIVF International Conference on Computing and Communication Technologies, RIVF 2021, Hanoi, Vietnam, August 19-21, 2021*. IEEE, 2021, pp. 1–6. Available from DOI: 10.1109/RIVF51545.2021.9642099.
3. **PHAM, Phuong N.H.**; NGUYEN, Bich-Ngan T.; CO, Quy T. N.; PHAM, Canh V.; SNÁŠEL, Václav. Threshold Benefit for Groups Influence in Online Social Networks. In: *Future Data and Security Engineering - 8th International Conference, FDSE 2021, Virtual Event, November 24-26, 2021, Proceedings*. Springer, 2021, vol. 13076, pp. 53–67. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-91387-8_4.
4. **PHAM, Phuong N.H.**; NGUYEN, Bich-Ngan T.; CO, Quy T.N.; NGUYEN, Nguyen T.; TRAN, Phuoc; SNÁŠEL, Václav. An efficient hybrid algorithm for community structure detection in complex networks based on node influence. *ICIC Express Letters, SCOPUS*. 2021-10, vol. 12, pp. 899–908. Available from DOI: 10.24507/icicelb.12.10.899.
5. **PHAM, Phuong N.H.**; LE, Vang V.; SNÁŠEL, Václav. Community Detection in Complex Networks Using Algorithms Based on K-Means and Entropy. In: *Computational Collective Intelligence - 12th International Conference, ICCCI 2020, Da Nang, Vietnam, November 30 - December 3, 2020, Proceedings*. Springer, 2020, vol. 12496,

- pp. 241–251. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-63007-2_19.
6. **PHAM, Phuong N.H.**; PHAM, Canh V.; DUONG, Hieu V.; NGUYEN, Trung Thanh; THAI, My T. Groups Influence with Minimum Cost in Social Networks. In: *Computational Data and Social Networks - 10th International Conference, CSoNet 2021, Virtual Event, November 15-17, 2021, Proceedings*. Springer, 2021, vol. 13116, pp. 231–242. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-91434-9_21.
 7. NGUYEN, Bich-Ngan T.; **PHAM, Phuong N.H.**; PHAM, Canh V.; SU, Anh N.; SNÁŠEL, Václav. Streaming Algorithm for Submodular Cover Problem Under Noise. In: *RIVF International Conference on Computing and Communication Technologies, RIVF 2021, Hanoi, Vietnam, August 19-21, 2021. IEEE*, 2021, pp. 1–6. Available from DOI: 10.1109/RIVF51545.2021.9642118.
 8. NGUYEN, Bich-Ngan T.; **PHAM, Phuong N.H.**; TRAN, Loi H.; PHAM, Canh V.; SNÁŠEL, Václav. Fairness Budget Distribution for Influence Maximization in Online Social Networks. In: *Artificial Intelligence in Data and Big Data Processing*. Springer International Publishing, Springer, 2022, pp. 225–237. Available from DOI: 10.1007/978-3-030-97610-1_19.
 9. LE, Vang V.; **PHAM, Phuong N.H.**; TRAN, Toai K.; SNÁŠEL, Václav. An approach of anchor link prediction using graph attention mechanism. *Bulletin of Electrical Engineering and Informatics, SCOPUS*. 2022, vol. 5, pp. 2895–2902. Available from DOI: <https://doi.org/10.11591/eei.v11i5.4274>.
 10. LE, Vang V.; TRAN, Tin T.; **PHAM, Phuong N.H.**; SNÁŠEL, Václav. Anchor Link Prediction in Online Social Network Using Graph Embedding and Binary Classification. In: *Computational Collective Intelligence - 12th International Conference, ICCCI 2020, Da Nang, Vietnam, November 30 - December 3, 2020, Proceedings*. Springer, 2020, vol. 12496, pp. 229–240. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-030-63007-2_18.
 11. **PHAM, Phuong N.H.**; NGUYEN, Hien T.; SNÁŠEL, Václav. Improving Node Similarity for Discovering Community Structure in Complex Networks. In: *Computational Social Networks - 5th International Conference, CSoNet 2016, Ho Chi Minh City, Vietnam, August 2-4, 2016, Proceedings*. Springer, 2016, vol. 9795, pp. 74–85. Lecture Notes in Computer Science. Available from DOI: 10.1007/978-3-319-42345-6_7.
 12. **PHAM, Phuong N.H.**; PHAM, Canh V.; DUONG, Hieu V.; SNÁŠEL, Václav. Minimizing Cost for Influencing Target Groups in Social Network: A Model and Algorithmic

Approach. *Computer Communications Journal, Elsevier, SCIE, Q1, IF: 5.047 (Under review)*. [N.d.].

13. **PHAM, Phuong N.H.**; PHAM, Canh V.; CO, Quy T.N.; SNÁŠEL, Václav. Streaming algorithm for k-submodular cover problem. In: *CsoNet 2022, Springer (Under review)*. [N.d.].
14. NGUYEN, Bich-Ngan T.; **PHAM, Phuong N.H.**; LE, Vang V.; SNÁŠEL, Václav. Influence Maximization under Fairness Budget Distribution in Online Social Networks. *Mathematics, SCIE, Q1, IF: 2.592 (Under review)*. [N.d.].