Algorithms for influence maximization and seed minimization

Shi, Yanchen

2019

Shi, Y. (2019). Algorithms for influence maximization and seed minimization. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/104632>

<https://doi.org/10.32657/10220/49508>

NANYANG TECHNOLOGICAL UNIVERSITY

Algorithms for Influence Maximization and Seed Minimization

A thesis

Submitted to the School of Computer Science and Engineering
of the Nanyang Technological University

by

Shi Yanchen

to the Degree of Master of Engineering

March 25, 2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

31 Mar 2019

.....
Date

.....
Shi Yanchen

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

31 Mar 2019

.....
Date

.....
Dr. Sun Aixin

Authorship Attribution Statement

This thesis contains material from 2 paper(s) published in the following peer-reviewed journal(s) / from papers accepted at conferences in which I am listed as an author.

Title : Influence Maximization: Near-Optimal Time Complexity Meets Practical Efficiency.

Authors : Youze Tang, Xiaokui Xiao, Yanchen Shi

Venue : Proceedings of the ACM International Conference on Management of Data (SIGMOD)

Pages : 75-86, 2014

Title : Influence Maximization in Near-Linear Time: A Martingale Approach

Authors : Youze Tang, Yanchen Shi, Xiaokui Xiao

Venue : Proceedings of the ACM International Conference on Management of Data (SIGMOD)

Pages : 1539-1554, 2015

Some contents of Chapter 3 are published as Influence Maximization: Near-Optimal Time Complexity Meets Practical Efficiency. Youze Tang, Xiaokui Xiao, Yanchen Shi, Proceedings of the ACM International Conference on Management of Data (SIGMOD), pages 75-86, 2014.

The contributions of Shi Yanchen are as follows:

1. Shi Yanchen helped recheck the proofs in the paper.
2. Shi Yanchen collaborated with me test some benchmarks in experiments
3. Figures and some miscellaneous images were drawn by Shi Yanchen
4. Some coding works were implemented by Shi Yanchen
5. The poster was made by Shi Yanchen

Some contents in Chapter 3, 5 are published as Influence Maximization in Near-Linear Time: A Martingale Approach. Youze Tang, Yanchen Shi, Xiaokui Xiao, Proceedings of the ACM International Conference on Management of Data (SIGMOD), pages 1539-1554, 2015.

The contributions of Shi Yanchen are as follows:

1. Dr Xiao Xiaokui, Shi Yanchen, and Tang Youze together raised the main idea of this paper based on the state-of-the-art benchmark TIM.
2. Shi Yanchen made clear on the estimation of three complicatedly related parameters ε_1 , δ_1 and δ_2 , which is shown in Equation 5 and Lemma 5. This improvement largely reduces the size of sample set we require without affecting the approximations.
3. The most proofs, including the majority in the main part and all in appendix, were written by Shi Yanchen.
4. Shi Yanchen helped me in some coding works as well as experimental testing
6. Figures and some miscellaneous images are drawn by Shi Yanchen
7. The poster was made by Shi Yanchen

31 Mar 2019

.....
Date

.....
Shi Yanchen

Acknowledgments

I would like to express my sincere appreciation to my former supervisor Dr. Xiaokui Xiao, current supervisor Dr. Aixin Sun and the Assistant Chair Dr. Kwoh Chee Keong. Under Dr. Xiao's supervision, I have learned not only cutting-edge knowledge on database, but also an inspiring and down-to-earth working attitude. Dr. Sun and Dr. Kwoh Chee Keong have been providing me with great support in my study.

I would also like to acknowledge my colleagues at Nanyang Technological University and Fudan University for their friendship and insightful discussions. They include Dr. Sibow Wang, Keke Huang, Renchi Yang, Dr. Jun Zhang, Dr. Youze Tang, Dr. Zongyang Ma and Sheng Zeng.

Contents

State of Originality	i
Supervisor Declaration Statement	ii
Authorship Attribution Statement	iii
Acknowledgments	v
Abstract	ix
List of Figures	xi
List of Tables	xiii
1 Introduction	1
1.1 Background	1
1.2 Our Contribution	4
1.3 Report Organization	5
2 Literature Survey	6
2.1 Influence Spread on Social Networks	6
2.1.1 Diffusion Models	7
2.2 Researches on Influence Maximization	11
2.3 Researches on Seed Minimization	16
3 Influence Maximization via Martingale	18
3.1 Introduction	18
3.2 Preliminaries	19
3.2.1 Problem Definition	20
3.2.2 Kempe et al.'s Greedy Approach	21
3.2.3 Revisiting <i>TIM</i> and <i>TIM</i> ⁺	22

3.3	Proposed Solution	25
3.3.1	A Martingale View of RR Sets	26
3.3.2	Node Selection Phase	28
3.3.3	Sampling Phase	36
3.3.4	Putting It Together	43
3.4	Extensions	48
3.4.1	Generalization of <i>IMM</i>	48
3.4.2	Application to Continuous-Time Models	50
4	Greedy Seed Minimization	57
4.1	Introduction	57
4.2	Preliminaries	58
4.2.1	Goyal et al.’s Approach	58
4.2.2	Zhang et al.’s Approach	58
4.2.3	Problem Definition	59
4.3	Proposed Solution	60
4.3.1	Sampling Phase	60
4.3.2	Node Selection Phase	62
4.4	Comparison	67
5	Experiments	71
5.1	Influence Maximization via Martingale	71
5.1.1	Experimental Settings	71
5.1.2	Results under the IC Model	73
5.1.3	Results under the LT model	75
5.1.4	Results under the CTIC Model	76
5.2	Greedy Seed Minimization	78
5.2.1	Experimental Settings	78
5.2.2	Results under the IC Model	79
5.2.3	Results under the LT Model	80

6 Conclusion and Future Work	82
6.1 Conclusions	82
6.2 Future Works	83
References	84
Publication List	90

Abstract

Graph is a basic mathematical tool that models information about identities as well as their complex relationships from various real world problems. It has been found important applications in analysis on social networks, route planning, telecommunication, etc. In recent years, the complexity and scale of real world graphs have increased dramatically. In particular, international social networks can comprise of hundreds of millions of users and up to billions of relationships. Thus, even algorithms with decent time or space complexities meet challenges dealing with large-scale networks for the queries like influence maximization and seed minimization on social networks. In this thesis, we investigate two problems on large scale networks for aforementioned applications, i.e., influence maximization and seed minimization.

Given G a social network, M a probabilistic propagation model, and a small number $k > 0$, the *influence maximization* problem aims to find the largest expectation of the number of influenced nodes that k nodes can trigger under this pre-defined model M . This problem is derived from *viral marketing*, where a company gives away free samples to a small number of *influential* individuals in order to create a cascade of adoption via *word-of-mouth* effect. This study proposes a two-phase approach *Influence Maximization via Martingales (IMM)* that meets both practical efficiency and theoretical guarantees. In particular, *IMM* returns an $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - n^{-\ell}$ probability in an $O((k + \ell)(n + m) \log n / \varepsilon^2)$ running time. *IMM* is further extended to fit in triggering model and time-continuous model. We experimentally evaluate *IMM* with the state-of-the-art benchmarks under several diffusion models and parameter settings, using large networks with up to 1.4 billion edges. The experimental results show that our approach consistently outperforms the states of the art in terms of efficiency.

The *seed minimization* problem is a variant problem of the influence maximization with the same origin from advertising. Given a social network G and a covering threshold t , the seed minimization problem is aimed to find a seed set S that has an expected influence nodes not less than $t \cdot n$ and minimizes the size of S . Compared to the influence maximization that maximizes the influence given a certain budget, the seed minimization problem hopes to retrench the expense to the minimum number while keeping the influence above a pre-defined threshold. To solve the problem, we propose *GSM*, a greedy algorithm with tight approximations, high generalization and easy implementations. In particular, it yields a $\lceil (1 + \phi) \log(tn) \rceil$ -approximate solution with at least $1 - n^{-\ell}$ probability, where ℓ and ϕ are both tunable. We experimentally evaluate *GSM* in several settings of both t and β , and it is often orders of magnitude faster compared to traditional greedy benchmark *MINTSS*. *GSM* also gives an impressive performance on a large graph *Twitter* with more than a billion edges.

List of Figures

2.1	Social network G of IC model.	9
2.2	Activated nodes for seed node set $\{v_4\}$	9
2.3	Social network G of LT model.	10
2.4	Activated nodes for seed node set $\{v_4\}$	10
3.1	Social network G	24
3.2	Random graph g_1	24
3.3	λ^*/λ vs. ε	35
3.4	λ^*/λ vs. k	35
5.1	Running time and expected influence on <i>Twitter</i> under the IC model ($\ell = 1$ for <i>IMM</i> and <i>TIM</i> ⁺).	74
5.2	Running time vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for <i>IMM</i> , <i>TIM</i> , and <i>TIM</i> ⁺).	74
5.3	Expected influence vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for <i>IMM</i> , <i>TIM</i> , and <i>TIM</i> ⁺).	74
5.4	Running time and expected influence on <i>Twitter</i> under the LT model ($\ell = 1$ for <i>IMM</i> and <i>TIM</i> ⁺).	76
5.5	Running time vs. k under the LT model ($\varepsilon = 0.5$ and $\ell = 1$ for <i>IMM</i> , <i>TIM</i> , and <i>TIM</i> ⁺).	76
5.6	Expected influence vs. k under the LT model ($\varepsilon = 0.5$ and $\ell = 1$ for <i>IMM</i> , <i>TIM</i> , and <i>TIM</i> ⁺).	77
5.7	Running time of <i>IMM</i> and <i>ConTinEst</i> under the CTIC model.	77
5.8	Running time vs. t under the IC model for <i>GSM</i> and <i>MINTSS</i>	79
5.9	Running time vs. t and vs. <i>beta</i> under the IC model for <i>GSM</i>	79

5.10	Running time vs. <i>t</i> under the LT model for <i>GSM</i> and <i>MINTSS</i>	80
5.11	Running time vs. <i>t</i> and vs. <i>beta</i> under the LT model for <i>GSM</i>	80

List of Tables

2.1	Generalize IC model in triggering model.	11
3.1	Frequently used notations for <i>IMM</i>	26
4.1	Frequently used notations for <i>GSM</i>	61
4.2	Comparison with Goyal et al's method	70
5.1	Datasets.	72

Chapter 1

Introduction

Graphs provide a natural way to demonstrate real world entities and their complex relationships. Hence they are extensively utilized in various domains such as social network, road networks, telecommunications etc. Since the rapid increasing scale of real world networks, algorithms with decent time and space complexity may incur unsatisfactory computation overheads, and then motivate a plethora of studies addressing these obstacles. In particular, a social network like *Twitter* consists of billions of relationships is a great challenge for designing an algorithm extracting any information. This study focuses on handling two topics on large social networks, i.e., influence maximization and seed minimization. In what follows, we briefly introduce the background and motivation of two problems, as well as the main contributions in the thesis.

1.1 Background

Influence Maximization. The first problem we study is the influence maximization on social networks. Let $G = (V, E)$ be a social network, and M be a pre-defined probabilistic diffusion model. Given G , M and a small positive number k , *influence maximization* problem asks for the largest expectation of number of influence that k nodes can trigger under this model M . The influence maximization problem is derived from the study on how information is propagated in a social network, especially via the word-of-mouth

effect [22, 53]. It is an important application in *viral marketing*, where a company provides a few *influential* individuals with free samples, hoping that this product will be recursively adopted by recommendation from his/her friends and followees, and further create a cascade of product adoption. In other word, the objective of the influence maximization problem is to find the a small set of users to maximize the effect of advertising given a limited budget. A good example is *celebrity branding* where a company hires celebrities as influential nodes to enhance advertising effectiveness. At first glance, the influential nodes are easily detected since more influential nodes intuitively have more followers (i.e., higher degrees in the graph). However, the recursive spreading process and overlapping of influenced nodes result in complicated propagation process. The influence maximization turns out to be a complicated problem which can be reduced to a set cover problem proving its NP-harness [39].

Kempe et al. are the first to formulate influence maximization problem [39], and to propose several diffusion models. *Linear threshold* and *independent cascade* are the most prominent ones, and widely adopted by other groups. Kempe et al. show that it is an NP-hard problem, and present a general greedy approach that yields a $(1 - 1/e - \epsilon)$ -approximate solution. Since then, a large bunch of approaches have been proposed [15, 32, 14, 58, 10] to increase the practical efficiency.

Nevertheless, most techniques either fail retaining the superb $(1 - 1/e - \epsilon)$ -approximations when shortening the running time, or incur prohibitive time complexity although they provide guaranteed solutions. In particular, methods [39, 38, 13, 14, 8] rely on heuristic techniques to offer empirical efficiency, at the cost of offering no worst case guarantees. A few exceptions [11, 32] retain the $(1 - 1/e - \epsilon)$ -approximations, but cannot process large graphs due to their high time complexity.

This dilemma motives us to propose *IMM*, an algorithm meets both efficiency and accuracy. It leverages *Martingale*, a statistical tool, to return a $(1 - 1/e - \epsilon)$ -approximate

solution with at least $1 - n^{-\ell}$ probability within an $O((k + \ell)(n + m) \log n / \varepsilon^2)$ running time.

Seed Minimization. The second part is on seed minimization problem. Given a social network G and a covering threshold t , seed minimization problem is to find a seed set S that has an expected influence not less than $t \cdot n$ and the minimize the size of S . Apparently, seed minimization problem is derived from the original influence maximization problem. In contrast to the original influence maximization problem where the budge is predefined, seed minimization is to minimize the expenditure as long as the advertising effect is above a threshold. Both problems share several aspects in common. They are both NP-hard problems and rely on greedy algorithms to find approximate solutions.

It is firstly described in Goyal et al.'s work as Minimum Target Set Selection (*MINTSS*) [28, 29]. They present a $(1 + \ln(t/\varepsilon))$ -approximate solution by employing greedy algorithm, where ε is a predefined parameter. But their method does not meet any theoretical guarantees such that the yielded solution fails to have an expected influence spread larger than $t \cdot n$. Zhang et al. present a guaranteed solution at the sacrifice of larger time complexity [64].

Despite the high accuracy declared in Goyal et al.'s approach [28, 29], it requires an estimator of influence involving an unknown parameter. In other words, their experimental results do not reflect the alleged theoretical guarantees. While Zhang et al.'s solution suffers a high time complexity [64].

The deficiencies of the existing algorithms drive us to develop *GSM*, a more sophisticated algorithm on the seed minimization problem. *GSM* returns a $\lceil (1 + \phi) \log(tn) \rceil$ -approximate solution, where ϕ is a parameter derived from two predefined parameters.

1.2 Our Contribution

In summary, our contributions to the influence maximization problem are as follows:

- (i) We propose *IMM*, an influence maximization algorithm that reaches a balance between theoretical accuracy and practical efficiency under a general diffusion model —*triggering model* [39]. It returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - 1/n^\ell$ in an $O((k + \ell)(n + m) \log n / \varepsilon^2)$ expected running time.
- (ii) We further extend the usage of *IMM* to temporal diffusion model. In particular, we present a *continuous-time independent cascade (CTIC)* model, which retains the $(1 - 1/e - \varepsilon)$ -approximation ratio.
- (iii) We experimentally evaluate *IMM* under both the triggering and the CTIC model with very large social networks in which the largest one holds up to 1.4 billion edges. The results show that *IMM* is orders of magnitude faster than the existing works.

And our contributions to the seed minimum problem are curated below:

- (i) We propose *GSM*, a seed minimum algorithm that returns an $O(\log(\frac{t \cdot n}{\beta}))$ -approximate solution given a predefined β . And *GSM* overcomes a deficiency of existing seed minimum problems that the theoretical guarantees depend on an unknown value, the size of ideal seed set S_t^o .
- (ii) We modify Goyal et al.’s *MINTSS* method [28, 29], such that it is no longer relying on $|S_t^o|$. And our theoretical analysis suggests that *GSM* outperforms this modified *MINTSS* in terms of theoretical accuracy.
- (iii) We experimentally compare *GSM* with other benchmarks under various datasets. *GSM* shows better running time in all tested datasets under both the IC and the

LT models. We further evaluate the running time of *GSM* in a billion-edge dataset. It can terminates within 30 minutes while other benchmarks cannot finish in 12 hours.

1.3 Report Organization

The remainder of the thesis is organized as follows. Chapter 2 describes the literature survey of current research work on influence maximization and seed minimization. Chapter 3 demonstrates *IMM* approach on influence maximization problem. Chapter 4 shows *GSM* method handling seed minimization problem. Chapter 5 presents our experimental results of two queries. Chapter 6 concludes the work we finished and presents the future work.

Chapter 2

Literature Survey

This chapter first describes the definition of influence maximization and seed minimization, and then presents existing approaches addressing the problems.

2.1 Influence Spread on Social Networks

The size of social networks is booming to billions of users. In 2018, *Twitter* has reached 335 million users in total [1], and *Facebook* has even outnumbered 2.2 billion users [2]. This large scale of users, as well as the interactions among them draw much attention on how information spread on social networks. Especially, *word-of-mouth* effect [22, 53], which describes how a piece of information propagates from users to their friends via social network, demonstrates the huge power of social networks. It has found important applications in many areas [35, 6], like political campaign and social network challenge. A good example is *ice bucket challenge*, where people dump a bucket of ice and water over their heads, and nominate their friends to follow this challenge. This activity is aimed to promote public awareness of *amyotrophic lateral sclerosis (ALS)*, a neurone disease that causes patients to gradually lose their control of voluntary muscles. It went viral in August 2014, as over 6 million videos were shared on *YouTube* during that summer [62].

Both influence maximization and seed minimization problems are originated from *viral marketing* [22, 53], another important application of influence spread, where a com-

pany provides some free products to a set of selected *influential* users, and wishes other individuals can adopt this product via the recommendation from his/her friends. This adoption by recommendation can be triggered recursively in the network, to fulfill the company's advertising purpose. And due to the limited budget, the company can only provide these free samples to a small set of *influential* users.

Kempe et al. [39] are the first group that formally define the problem of influence maximization. Let G be a social network, and M be a diffusion model that captures the characteristics of how nodes in G influence their neighbors' behavior. Given G , M and a positive integer k , the influence maximization problem asks for a size- k node set S_k with the maximum expected influence. Seed minimization is a variant problem derived from influence maximization. Given G , a diffusion model M , a tunable parameter $t \in (0, 1)$, the seed minimization problem is aimed to find the minimum seed set S_t with expected influence satisfying $\mathbb{E}[I(S_t)] \geq t \cdot n$.

2.1.1 Diffusion Models

Both influence maximization and seed minimization problems rely on a certain diffusion model. Here we first present a generic time-stamped diffusion model:

- (i) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
- (ii) When a node u is first activated at timestamp i , each inactive outgoing neighbor v of u is to be judged whether become activated according to a certain diffusion model. The node v is set to be activated at timestamp $i + 1$ if this judgement is true. After that, u cannot activate any other nodes.
- (iii) Once a node becomes activated, it remains activated in all subsequent timestamps.
- (iv) This process terminates when there are no more nodes that can be influenced.

Note that an activated node cannot be de-activated in the whole process. This property is named as *progressive*. And it ensures the convergence of the diffusion model, such that the process cannot run indefinitely. Three prominent models are widely used among literature — the *Linear Threshold (LT)* model, the *Independent Cascade (IC)* model and the *Triggering* model, which are discussed in this section.

2.1.1.1 Independence Cascade Model

The independence cascade (IC) model is first described in Goldenberg et al's work [24], and then adopted by Kempe et al. [39] as one of their diffusion models. The basic idea of the IC model is that a user can be activated with a certain probability, whenever one of his/her followees (i.e., incoming neighbors in a graph) adopts a new product. For a given social network $G = (V, E)$, each edge $e = (u, v)$ is associated with a propagation probability $p(e)$. Now we can rewrite the diffusion model in IC framework:

- (i) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
- (ii) When a node u is first activated at timestamp i , for each inactive outgoing neighbor v of u , we flop a coin according to the propagation probability $p(e)$, where $e = (u, v)$. The node v is activated in the next timestamp if the result is true. After that, u cannot activate any other nodes.
- (iii) Once a node becomes activated, it remains activated in all subsequent timestamps.
- (iv) This process terminates when there are no more nodes that can be influenced.

To put it more specifically, Figure 2.1 illustrates a social network under IC model, in which each edge (u, v) is associated with a value indicating the probability that u could influence v . If we set $k = 1$, and select $\{v_4\}$ to be the seed set S , we can see that v_1 can be

 Figure 2.1: Social network G of IC model.

 Figure 2.2: Activated nodes for seed node set $\{v_4\}$.

definitely influenced and no more nodes can be activated. Thus, we can obtain the result of influenced nodes, as dyed black ones in Figure 2.2. Let $I(S)$ is the influence of a node set. The expected influence of $\{v_4\}$ can be written as $\mathbb{E}[I(\{v_4\})] = 1 + 1 \times 1 = 2$, where the first term 1 represents v_4 at the first timestamp and 1×1 indicates that v_1 is activated with 1 probability at the second timestamp. By similar means, we can calculate the influence of other seed sets, as $\mathbb{E}[I(\{v_1\})] = 1$, $\mathbb{E}[I(\{v_2\})] = 1 + 0.01 + (0.01 + 0.01 \times 1 \times (1 - 0.01))$, in which the second term shows the instance that v_1 is activated at the second timestamp, and the third term refers to v_1 being activated at (i) the second timestamp via v_2 , or (ii) at the third timestamp via v_4 given the condition that (i) not happens. $\mathbb{E}(I(\{v_3\}))$ has the same value as $= \mathbb{E}[I(\{v_2\})]$.

2.1.1.2 Linear Threshold (LT) Model

The linear threshold (LT) model is also one of the models adopted by Kempe et al. [39]. Granovetter et al. [34] and Schelling et al. [54] are the first introducing LT model. The LT model is based on the idea that a user also adopts the product when the number of his/her followees that recommend the product has outnumbered a certain threshold. Thus, instead of a probability as in the IC model, each edge e in the LT model is associated with a weight $w(e)$. And each node v is associated with a threshold $th(v)$ selected from $(0, 1]$. Let $IN(v)$ be the set of all incoming edges of v . The equation $\sum_{e \in IN(v)} w(e) \leq 1$ holds. The LT model works in the following steps:

- (i) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.

 Figure 2.3: Social network G of LT model.

 Figure 2.4: Activated nodes for seed node set $\{v_4\}$.

- (ii) For an inactivated node v , once the equation $\sum_{e=(u,v) \in \{IN(v) | u \text{ is activated}\}} w(e) \geq th(v)$ is satisfied, it is set to be activated in the next timestamp.
- (iii) Once a node becomes activated, it remains activated in all subsequent timestamps.
- (iv) This process terminates when there are no more nodes that can be influenced.

Figure 2.3 demonstrates an example of the LT model. Note that each edge (u, v) is associated with a weight and each node is associated with its threshold. By setting $k = 1$ and seed set as $S = \{v_4\}$, the activated nodes as shown in Figure 2.4. Similar to the case of the IC model, both v_4 and v_1 can be activated, and the rest remains inactivated.

2.1.1.3 Triggering Model

The triggering model is proposed by Kempe et al. [39] as the generalization of the IC and the LT model. It assumes that each node v is associated with a triggering distribution $\mathcal{T}(v)$, that maps a subset of incoming neighbors to a probability. In other words, at each timestamp, whether an inactivated node v can be activated or not is depending on a subset of incoming neighbors. A triggering model proceeds in following steps:

- (i) For each node v , we take a sample from $\mathcal{T}(v)$, denoted as the *triggering set* of v .
- (ii) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
- (iii) At a timestamp i , if there exists an activated node u in $\mathcal{T}(v)$ of an inactive node v , then v is activated at timestamp $i + 1$.

Triggering set	Probability
$\{v_2\}$	0.01×0.99
$\{v_3\}$	0.01×0.99
$\{v_2, v_3\}$	0.01×0.01
$\emptyset$	$1 - 0.01 \times 0.01 - 0.01 \times 0.99 - 0.01 \times 0.99$

Table 2.1: Generalize IC model in triggering model.

(iv) Once a node becomes activated, it remains activated in all subsequent timestamps.

(v) This process terminates when there are no more nodes that can be activated.

Both the IC and the LT models are proven to be special cases of the triggering model [39]. To reduce the IC model to the triggering model, we can assign a triggering distribution of each node v , such that each incoming node of edge $e = (u, v)$ independently appears in the triggering set of v , with the probability of $p(e)$. If we take v_4 in Figure 2.1 as the example, all triggering sets of the power set of v_4 's incoming neighbors are $\{v_2\}$, $\{v_3\}$, $\{v_2, v_3\}$ and $\emptyset$. The corresponding triggering probability can be computed as in Table 2.1.

Expressing the LT model in triggering model is also straightforward. For each node v , its triggering distribution $\mathcal{T}(v)$ only maps $\emptyset$ and singleton $\{u\}$ for each incoming neighbor u of v to a probability.

2.2 Researches on Influence Maximization

Kempe et al. [39] are the first that formally define the problem of influence maximization. Let G be a social network, and M be a probabilistic diffusion model that captures the characteristics of how nodes in G influence their neighbors' behavior. Given G , M and a positive integer k , the influence maximization problem asks for a size- k node set S_k with the maximum expected influence.

As mentioned above, Kempe et al. [39] are the first showing the formal definition of influence maximization as a combinatorial optimization problem, and proving its NP-hardness. They propose a general greedy approach that yields a $(1 - 1/e - \varepsilon)$ -approximate solution under several diffusion models. Its generality and effectiveness ensure that it can be easily implemented under any diffusion model, and provide a good theoretical guaranteed solution. However, the greedy algorithm incurs prohibitive computation overheads because of its $k m n r$ time complexity, where n and m refer to the numbers of nodes and edges, respectively. Motivated to design more practically efficient approaches, a plethora of techniques [38, 32, 9, 13, 8, 26, 41, 16, 30, 33, 31, 15, 14, 61, 43, 8, 55, 39, 40, 9, 58, 25, 27, 26, 23] have been proposed to handle influence maximization on large social networks.

However, those algorithms either achieve practical efficiency by sacrificing theoretical guarantees, or vice versa. In particular, heuristic techniques are adopted by most approaches to efficiently estimate a large influence of a certain node set. The idea is that intuitively the evaluation of the influence spread has a high relation with some well-studied problems, like PageRank [52] or shortest path [21]. Heuristic methods, rank all users in graph G according to some evaluation functions.

Group-PageRank [47] proposed by Liu et al can estimate the influence spread in a linear time complexity. As indicated in the its name, they sum up PageRank scores of all nodes in a node set S , and remove the mutual influence between the nodes. Also based on PageRank, Jung et al. present *Influence Ranking Influence Estimation (IRIE)* algorithm [38]. *IRIE* derives a system of n linear equations for $\{I(u|S)\}_{u \in V}$, where $I(\cdot)$ is the influence spread and $I(u|S)$ is the margin gain of adding u into the node set, i.e., $I(u|S) = I(u \cup S) - I(S)$. This equation set can be solved in an iterative manner.

Maximum influence arborescence (IMA) as well as its extension *prefix excluded IMA (PMIA)* proposed by Chen [14] are well-known heuristic benchmarks for IC model. For a node v , instead of considering all complicated influence from u to v , *IMA* only traverses

the most *influential* path from u to v . This path is named as a *maximum influence path (MIP)* whose probability is the product of $p(e)$ for all edges e in the path. By this means, the influence to v is structured as a tree rooted at u . After setting a threshold θ , *maximum influence in-arborescence (MIIA)* (v, θ) is generated by containing all MIPs ending at v with propagation probability larger or equal to θ . A related term *maximum influence out-arborescence (MIOA)* (v, θ) is defined as all MIPs starting from v . The margin influence $I(u|S)$ can be efficiently calculated from MIIAs and MIOAs.

Heuristic approaches mentioned above are all based on the IC model, but there are some conducted on LT models, like *SIMPAT*H algorithm proposed by Goyal et al. [33]. They convert the influence maximization problem on the LT model to enumerating all simple paths starting from each node v of the node set. This conversion does not reduce the computation complexity, since enumerating all simple paths is also NP-hard. Therefore, when exploring the simple paths, *SIMPAT*H prunes out-going edges whose weights are smaller than a certain threshold. Goyal et al. introduce two optimization techniques to further reduce the computing overheads. However, the running time of *SIMPAT*H is highly sensitive to the size of seed node k . Much more simple paths have to be enumerated if k is set to a large number.

Despite these approaches' high efficiency and decent experimental results, the superb $(1-1/e-\epsilon)$ -approximations of Kempe et al.'s greedy method are no longer retained. A few exceptions design approaches that improve experimental efficiency without sabotaging these approximations of the greedy solution [43, 32, 11]. These approaches keep the basic greedy framework and aim to reduce the number of Monte Carlo simulations. Leksovec et al. present *CELF* [43] based on the *power law* property of social networks which indicates that the majority of nodes in G have little influence and can be pruned without impacting the approximations. To put it more specifically, it estimates an upper bound of margin influence $\hat{I}(u|S)$ for each visited node u and gets updated at each iteration.

By visiting nodes in a descending order of upper bounds, *CELF* can terminate earlier once $\max\{\hat{I}(u|S)\} < I'(v|S)$ holds, where $\max\{\hat{I}(u|S)\}$ is the maximum upper bound of all unvisited nodes, and $I'(v|S)$ is the estimated margin influence of a visited node v by Monte Carlo simulations. This early termination does not affect the approximations of *CELF* due to the submodularity of influence function. A improved version *CELF++* [32] is later presented by more sophisticated pruning. But they both have the same worst case time complexity with Kempe et al.'s method and still incur prohibitive overheads on large graphs.

Borgs et al. [9] make a theoretical breakthrough bridging practical efficiency and theoretical guarantees. They leverage probabilistic tools like Chernoff bounds to design an algorithm without compromising either approximations or efficiency. In particular, their proposed method returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - n^{-\ell}$ probability, in an $O(k\ell^2(m + n) \log^2 n / \varepsilon^3)$ running time. They further prove that this method is near optimal since all algorithms that provide same approximations with a constant probability, must have a time complexity of $\Omega(m + n)$ [9]. Its experimental results, however, are not satisfactory, due to its large hidden coefficient in the time complexity.

Inspired by Borgs et al., Tang et al. propose *TIM* and *TIM*⁺ [58]. Both methods run in $O((k + \ell)(n + m) \log n / \varepsilon^2)$ expected time, and return $(1 - 1/e - \varepsilon)$ -approximate solutions with at least $1 - 1/n^{-\ell}$ probability. Their experimental results outperform Borg et al.'s counterpart by a wide margin in terms of handling large networks. Specifically, *TIM* (as well as its variant *TIM*⁺) is the first approach that can process large real networks with up to 1.4 billion edges. Nevertheless, *TIM* and *TIM*⁺ have two deficiencies. The first is that its estimated lower bound of OPT is rather conservative such that a large number of unnecessary samples are incurred, where OPT is the ideal maximum influence of k nodes; The second is that the computation of the lower bound is time-consuming.

This novel frameworks proposed by Borgs et al. [9] and Tang et al. [58] encourage a bunch of researchers follow their work to address influence maximization problem from different perspectives. Li et al. [44] name this kind of methods as *Sketch-based Approaches*. They first construct theoretically grounded sketches under some diffusion models. The core of Cohen et al.’s *SKIM* approach [19] is *combined reachability sketches* that succinctly represents influences spread. By applying *BFS* walks on each reversed sample graph g , *SKIM* obtains a sketch $Y_v^{(g)}$ for each node v on the graph g . But its worst case performance is $O(k \cdot n^2 m \cdot \log \binom{n}{k} / \varepsilon^2)$, which is still prohibitive on large graphs.

Motivated by the large space consumption of *TIM* and *TIM*⁺, Wang et al. [60] propose a *bottom- k sketch based reverse influence sampling (BKRIS)*. It employs a heuristic method to estimate a lower bound of *OPT*. Given θ samples derived from *OPT*, *BKRIS* uses the bottom- k sketch to find the k seeds covers the most samples. It can achieve an $O(1 - 1/e - \varepsilon - \varepsilon')$ -approximate solution, where ε and ε' is the error brought by estimation of sampling and bottom- k sketches, respectively. However, it’s possible that *BKRIS* could violate the theoretical guarantees [60].

Nguyen et al. improve *TIM* and *TIM*⁺ and propose a *stop-and-stare* optimization [51]. It presents an “early termination” of estimation — that the algorithm returns the seed set if the incremental influence of iteration i is close to 0. However, this method has a glitch in its theoretical proofs, as pointed in Huang et al.’s work [37]. The claimed $(1 - 1/e - \varepsilon)$ approximations of *stop-and-stare* can hardly be reached.

Except for seed minimization problem discussed in 2.3, influence maximization has been extended to fit in other situations. Topic-aware influence maximization takes *topics* into considerations. Tang et al. are the first introducing the concept of “topic” into the influence maximization problem [56]. To put it more specifically, each topic represents users’ interest on a certain area. The topic-aware traits can be associated with either the nodes [36, 45, 50], or the edges [7, 12]. If the nodes are topic-aware, only a subset of users

are relevant to a certain topic. And the algorithms focus on maximizing the influence on these relevant users. If the edges are topic-aware, the strength of influence over a user u to its follower v is affected by the corresponding topic. Thus, the propagation probability of edge e should be written as $p(e, z)$ for a certain topic z .

There are extensions considering even more factors. Lu et al. [49] formulate a problem aimed to maximize the “revenue” over a certain budget. This revenue maximization problem is not only topic-aware, but also sensitive to time. The returned result is a strategy over $V \times I \times [T]$, where V is the node set, I is the item sets, and $[T]$ is the time horizon.

2.3 Researches on Seed Minimization

The seed minimization is a variant problem derived from the influence maximization. Given G , a diffusion model M and a tunable parameter $t \in (0, 1)$, the seed minimization problem is aimed to find the minimum seed set S_t with expected influence satisfying $\mathbb{E}[I(S_t)] \geq t \cdot n$. It originates from the scenario in viral marketing to retrench the expense as much as possible to reach a preconcerted advertising effect. Chen et al. [10] study the seed minimization problem under the linear threshold (LT) model. And they prove that this problem is also NP-hard.

Goyal et al. [29, 28] present a $(1 + \ln(t/\varepsilon))$ -approximate solution by adopting a greedy scheme, where ε is a predefined parameter. But their method fails to guarantee that the yielded solution has an expected influence spread larger than $t \cdot n$, and it requires an unpractical oracle to estimate influence. Thus, their experimental results cannot reflect their proposed theoretical analysis.

Cordasco et al. [20] also study this problem on the LT model, where a node v is activated when the number of activated neighbors is greater or equal to a predefined threshold $th(v)$. Compared to other seed minimization algorithms, the target set of

Cordasco et al.'s algorithm is the whole graph G . In other words, this approach will not terminate until all nodes are activated. Since it adopts the LT model, it is feasible to activate all nodes if we set the threshold $th(v)$ properly.

A variant of the seed minimization problem studied by Long et al. [48] is named as *J-Min-Seed* in their paper. Instead of covering a predefined percentage of n nodes, *J-Min-Seed*, as indicated in its name, is aimed to minimize the number of seed set with at least J nodes activated. They also extend their algorithm to the case of full coverage, similar to Cordasco et al.'s work. Nevertheless, *J-Min-Seed* as well as its extension have a vital drawback that their time complexity is at least $\frac{(n-1)^2 \ln(2/(1-c))}{2\varepsilon^2 |S|^2}$ with the confidence at least c , where $|S|$ is the seed set size. In full coverage extension, the number of sampling process is further increased to no less than $\frac{(n-1)^2 \ln(2/(1-c))}{2\varepsilon^2}$ with at least c confidence.

Zhang et al. [64] propose a greedy algorithm that provides guaranteed solutions. In spite of its superb approximations, it has a higher time complexity than other methods. It is because the margin gain of adding a new node into the seed set S is too small when $I'(S)$, the estimated influence of S , is approaching $t \cdot n$.

Chapter 3

Influence Maximization via Martingale

3.1 Introduction

Kempe et al. [39] prove that influence maximization problem is NP-hard by reducing to a set cover problem. They present two classic models, the linear threshold (LT) and the independent cascade (IC) model, to mathematically describe the influence maximization problem on social network. And based on the greedy algorithm for the set cover problem, they propose a $(1 - 1/e - \varepsilon)$ -approximate greedy approach for either model, where ε is a constant number.

Despite the theoretical guarantees and simplicity of Kempe et al.’s greedy approach, it is known to be practically very time-consuming. Its $\Omega(kmn \cdot \text{poly}(\varepsilon^{-1}))$ [9] time complexity hinders this greedy algorithm from being applied on large networks, where n and m are the numbers of nodes and edges in G , respectively.

Our Contribution. We present *Influence Maximization via Martingales (IMM)*, an algorithm that overcomes deficiencies of *TIM* and *TIM*⁺. Both *TIM* and *TIM*⁺ utilize Chenoff bounds [17], a probabilistic tool on Bernoulli random variables, to obtain an estimation of expected influence with both bounded approximation and high probability in a near-linear time. *IMM* retains the superior approximation guarantee as $(1 - 1/e - \varepsilon)$ at

least $1 - n^{-\ell}$ probability, and its $O((k + \ell)(n + m) \log n / \varepsilon^2)$ expected time. *IMM* adopts the two-phase paradigm of *TIM*, but incorporates a drastically different parameter estimation phase. The core of *IMM* is a set of estimation techniques based on *martingales* [63]. With this classic statistical tool, *IMM* can obtain a lower bound of *OPT* that is not less than $OPT \cdot (1 - 1/e)/(1 + \varepsilon')^2$ with a high probability, where ε' is a tunable small parameter. This tighter estimation of *OPT* results in a smaller number of sample sets required in *IMM* sampling phase, which incurs a significantly smaller overheads than *TIM* and *TIM*⁺. In addition, the node selection phase is considerably improved with a martingale approach. The trait of martingale ensures that we can repeatedly use the sample sets generated in previous iterations without degrading its theoretical guarantees. In contrast, *TIM* is established on the assumption that all samples are independent and identically distributed (i.i.d.) random variables, which incurs a large amount of unnecessary computation that is avoidable in *IMM*. With the above optimizations, *IMM* significantly outperforms *TIM* and *TIM*⁺ in terms of practical performance. In particular, we experimentally evaluate *IMM* against *TIM* and *TIM*⁺ on a variety of social networks with up to 1.4 billion edges, and our results show that *IMM*'s running time is up to two orders of magnitude smaller than that of *TIM* and *TIM*⁺. Then we further extend *IMM* to a larger class of diffusion models. Particularly, we present an extended version of *IMM* under a temporal diffusion model.

3.2 Preliminaries

In this section, we present an overview of Kempe et al. and Tang et al.'s solutions [39, 58]. For ease of exposition, we focus on the *independent cascade (IC)* model [39] considered by Borgs et al. [9].

3.2.1 Problem Definition

Let G be a social network with a set V of n nodes and a set E of m directed edges. For any two nodes $u, v \in V$, if $\langle u, v \rangle$ is an edge in E , we say that v is an outgoing neighbor of u , and u is an incoming neighbor of v . We consider a time-stamped diffusion process on G as follows:

- (i) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
- (ii) When a node u is first activated at timestamp i , it samples a set of its outgoing inactive neighbors according to a certain probability distribution, and activates them at timestamp $i + 1$. After that, u cannot activate any other nodes.
- (iii) Once a node becomes activated, it remains activated in all subsequent timestamps.

Let $I(S)$ be the number of activated nodes in G when the above process converges, i.e., when no more nodes can be activated. We refer to S as the *seed set*, and $I(S)$ as the *influence* of S . Note that $I(S)$ is a random variable that depends on the probability distributions from which each node samples out-neighbors to activate. Intuitively, the influence propagation process under the IC model mimics the spread of an infectious disease: the seed set S is conceptually similar to an initial set of infected individuals, while the activation of a node by its neighbors is analogous to the transmission of the disease from one individual to another. We refer to the collection of those probability distributions as the *diffusion model* that underlies the diffusion process.

Problem Statement. Given G , a diffusion model M , and a positive integer k , the influence maximization problem asks for a size- k node set S_k with the maximum expected influence $\mathbb{E}[I(S_k)]$.

3.2.2 Kempe et al.'s Greedy Approach

In a nutshell, Kempe et al.'s approach [39] (referred to as *Greedy* in the following) starts from an empty seed set $S = \emptyset$, and then iteratively adds into S the node u that leads to the largest increase in $\mathbb{E}[I(S)]$, until $|S| = k$. That is,

$$u = \arg \max_{v \in V} \left(\mathbb{E}[I(S \cup \{v\})] - \mathbb{E}[I(S)] \right).$$

In concept, the main idea of *Greedy* is rather simple. The algorithm recursively selects the best node yielding the best margin gain until it collects enough set set. However, its implementation meets non-trivial challenges, since the computation of $\mathbb{E}[I(S)]$ is #P-hard [14]. To circumvent this prohibitive computing overheads, Kempe et al. propose an estimation method in a Monte Carlo manner, which can approximate the value of $\mathbb{E}[I(S)]$ to a reasonable accuracy in an acceptable time [39]. Specifically, for each edge e in G we flip a coin, and e is removed with $1 - p(e)$ probability. After flipping coins for all edge, we obtain a resulting graph g . We define a node v in g is *reachable* from the set S , if there exists a directed path in g from a node in S and ends at v . And let $R(S)$ be the set of nodes in g that are reachable from S . In Kempe et al.'s paper, a detailed proof is given showing that the expected value of $[I(S)]$ equals the expected size of $R(S)$. In other words, the equation $\mathbb{E}[I(S)] = \mathbb{E}[R(S)]$ holds. Hence, instead of directly working on $\mathbb{E}[I(S)]$, we can choose to estimate $\mathbb{E}[R(S)]$ which is much simpler. We first generate an enough large set samples of g , then calculate $R(S)$ on each sample. The arithmetic mean among all measurements of samples can be regarded an estimation of $\mathbb{E}[I(S)]$.

Let r be the number of samples in the estimation of each $\mathbb{E}[I(S)]$. Then, with a high probability, *Greedy* yields a $(1 - 1/e - \varepsilon)$ -approximate solution under the IC model [39], where ε is a constant that depends on both G and r [39, 9]. In general, *Greedy* achieves the same approximation ratio under any cascade model as well as LT model, given $\mathbb{E}[I(S)]$ meeting a *submodular* property over S [39]. To our knowledge, however,

there is no formal analysis in the literature on how r should be set to achieve a given ε on G . Instead, Kempe et al. suggest setting $r = 10000$, and majority of follow-up methods adopt similar choices of r .

Despite of generality and effectiveness of *Greedy*, it incurs significant computation overheads due to its $O(kmnr)$ time complexity. To be specific, it runs in k iterations, each of which involves expected influence estimation of $O(n)$ node sets. In addition, it requires measurements on r graphs for each estimation of expected spread, and each measurement further consumes $O(m)$ time exploring a graph. All factors above result in multiple nested loops yielding an $O(kmnr)$ total running time.

3.2.3 Revisiting *TIM* and *TIM*⁺

Tang et al.'s both *TIM* and *TIM*⁺ [58] have a parameter estimation phase and a node selection phase, the latter of which samples numerous node sets from the social network G to derive a solution for influence maximization. The type of node sets sampled by *TIM* and *TIM*⁺ is referred to as *random RR sets*, defined as follows:

Definition 3.1 (Reverse Reachable Set) *For any node v from a graph G , a reverse reachable (RR) set is generated by following two steps:*

- (i) *Flip coins to obtain a sampling graph g from $\mathcal{G}$ by removing edges;*
- (ii) *Construct a set of nodes in g that can reach v as the RR set.*

Since v can be uniformly selected at random from V , we define an RR set derived from a random v as a *random RR Set* for v . Here we give a formal definition:

Definition 3.2 (Random RR Set) *Given the randomness in removals of edges in creating g from G , we introduce a distribution of g as $\mathcal{G}$. For an instance of g randomly selected from $\mathcal{G}$, a random RR set is generated for a uniform-randomly selected node from g .*

Given two nodes u and v , if u belongs to an RR set of v , then intuitively any seed set including u should have a non-zero probability to activate v under a certain diffusion model. This connection between RR sets and node activations leads us to the following lemma:

Lemma 3.1 ([9]) *For any node v and any seed set S , the probability that the intersection of S and an RR set for v is not empty equals the probability that a pre-defined diffusion model from S can activate v .*

Based on Lemma 3.1, the node selection phase of TIM (resp. TIM⁺) adopts the following approach for influence maximization:

- (i) Generate a sizable set $\mathcal{R}$ of *independent* random RR sets.
- (ii) Consider the *maximum coverage* problem [59] of selecting k nodes to *cover*¹ the maximum number of RR sets in $\mathcal{R}$. Apply the standard greedy algorithm to obtain a $(1 - 1/e)$ -approximate solution S_k^* to the problem.
- (iii) Return S_k^* as the answer for influence maximization.

To understand why the above approach works, observe that for any seed set S , the fraction of RR sets in $\mathcal{R}$ covered by S is an unbiased estimator of $\mathbb{E}[I(S)]$ (this is due to Lemma 3.1). Therefore, a seed set S_k^* that covers a large number of RR sets in $\mathcal{R}$ is likely to have a large expected influence, which makes S_k^* a good solution to the influence maximization problem.

To ensure that S_k^* provides a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability, it is shown [58] that the number of RR sets in $\mathcal{R}$ should be at least λ/OPT , where OPT is the maximum expected influence of any size- k node set in G , and λ is a function of k , ℓ , n , and ε . Since OPT is unknown in advance, TIM sets $|\mathcal{R}| = \lambda/KPT$,

¹We say that a node set S covers an RR set R if $S \cap R \neq \emptyset$.

 Figure 3.1: Social network G .

 Figure 3.2: Random graph g_1 .

where KPT is a lower bound of OPT that is derived by the parameter estimation phase of TIM . However, KPT can be considerably smaller than OPT (which leads to an excessive number of RR sets in $\mathcal{R}$), and its derivation often incurs significant computational costs. To alleviate this issue, TIM^+ incorporates a heuristic approach to refine KPT . In particular, it selects a certain size- k node set S_k , and then estimates S_k 's expected influence $\mathbb{E}[I(S_k)]$. After that, TIM^+ uses the larger one between KPT and $\mathbb{E}[I(S_k)]$ as an improved lower bound of OPT . Although this heuristic approach tends to yield a tighter lower bound of OPT , it does not reduce the significant overhead in deriving KPT . Furthermore, in the worst case when $KPT \geq \mathbb{E}[I(S_k)]$, the heuristic approach would fail to offer any improvement over TIM .

Example 1 Consider that we invoke TIM on the social network G in Figure 3.1, setting $k = 1$. (The number on each edge indicates the propagation probability of the edge.) RIS first generates a number of random RR sets, each of which is pertinent to (i) a node sampled uniformly at random from G and (ii) a random graph obtained by removing each edge e in G with $1 - p(e)$ probability (see Definition 3.2). Assume that the first RR set R_1 is pertinent to v_1 and the random graph g_1 in Figure 3.2. Then, we have $R_1 = \{v_1, v_4\}$, since v_1 and v_4 are the only two nodes in g_1 that can reach v_1 .

Suppose that, besides R_1 , TIM only constructs three other random RR sets R_2 , R_3 , and R_4 , which are pertinent to three random graphs g_2 , g_3 , and g_4 , respectively. For simplicity, assume that (i) g_2 , g_3 , and g_4 are identical to g_1 and (ii) the node that TIM samples from g_i ($i \in [2, 4]$) is v_i . Then, we have $R_2 = \{v_2\}$, $R_3 = \{v_3\}$, and $R_4 = \{v_4\}$. In that case, v_4 is the node that covers the most number of RR sets, since it appears in two

RR sets (i.e., R_1 and R_4), whereas any other node only covers one RR set. Consequently, TIM returns $S_k^* = \{v_4\}$ as the result.

3.3 Proposed Solution

This section presents IMM , an influence maximization technique that borrows ideas from TIM and TIM^+ [58] but adopts a novel algorithm design to reduce computational costs. At a high level, IMM consists of two phases as follows:

- (i) **Sampling.** This phase iteratively generates random RR sets which together make up a set $\mathcal{R}$. This phase terminates when it meets a certain stopping condition.
- (ii) **Node Selection.** This phase iteratively selects k nodes, each of which gains the maximum margin gain of covered RR sets in $\mathcal{R}$ as in greedy manner. The returned a size- k node set S_k^* which can cover a large number of RR sets in $\mathcal{R}$ is the final result.

The node selection phase of IMM is similar to that of TIM and TIM^+ ; however, TIM and TIM^+ requires all random RR sets in $\mathcal{R}$ to be independent, which is not the case in IMM . In particular, the random RR sets generated by IMM are dependent because, in the sampling phase of IMM , whether the $(i + 1)$ -th RR set is generated relies on whether the first i RR sets satisfy the stopping condition. Allowing such dependencies among RR sets makes IMM 's theoretical guarantees more difficult to analyze, but it is crucial for IMM 's efficiency as it enables IMM to eliminate a considerable amount of redundant computation that is unavoidable in TIM and TIM^+ (see Section 3.3.4). Meanwhile, the sampling phase of IMM considerably differs from the parameter estimation phase of TIM and TIM^+ , in that it employs a more advanced statistical method to decide the number θ of RR sets required. This method not only provides more accurate choice of θ , but also

Notation	Description
$G = (V, E)$	a social network G with a node set V and an edge set E
n, m	the numbers of nodes and edges in G , respectively
k	the size of the seed set for INFMAX
$\mathcal{T}(v)$	the triggering distribution of a node $v \in V$
$\mathcal{G}$	the distribution of social networks induced by the triggering distribution of each node (see Section 3.2.1)
$I(S)$	the influence of a node set S in a diffusion process on G
$\mathbb{E}(\cdot)$	the expectation of a random variable
$\mathcal{R}$	the set of RR sets generated by <i>IMM</i> 's sampling phase
θ	the number of RR sets in $\mathcal{R}$
$F_{\mathcal{R}}(S)$	the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S
OPT	the maximum expected influence of any size- k node set S

Table 3.1: Frequently used notations for *IMM*.

leads to a significantly smaller cost of parameter estimation than in the case of *TIM* and *TIM*⁺.

In the following, we first introduce the concept of *martingales* [17], which is crucial in our analysis of the dependent RR sets in $\mathcal{R}$. After that, we discuss the condition under which the node selection phase of *IMM* returns a good solution, and then we clarify the details of the sampling phase. For ease of reference, Table 3.1 lists the notations that we frequently use. Unless specified otherwise, all logarithms in this thesis are to the base e .

3.3.1 A Martingale View of RR Sets

Let $R_1, R_2, \dots, R_\theta$ be the sequence of random RR sets generated in the sampling phase of *IMM*. Let S be any set of nodes in G , and x_i ($i \in [1, \theta]$) be a random variable that equals 0 if $S \cap R_i = \emptyset$, and 1 otherwise. By Lemma 3.1 and the linearity of expectation,

$$\mathbb{E}[I(S)] = \frac{n}{\theta} \cdot \mathbb{E} \left[\sum_{i=1}^{\theta} x_i \right]. \quad (\text{Eq. 3.1})$$

Accordingly, the node selection phase of *IMM* uses $\frac{n}{\theta} \cdot \sum_{i=1}^{\theta} x_i$ as an estimator of $\mathbb{E}[I(S)]$. To ensure that this estimation is accurate, it is essential that $\sum_{i=1}^{\theta} x_i$ should not deviate significantly from its expectation. A classic tool to prove such concentration results is the Chernoff bounds [17], but it requires all x_i to be independent, which contradicts the dependencies among x_i caused by the stopping condition in *IMM*'s sampling phase. We circumvent this issue with an analysis based on *martingales* [17], defined as follows:

Definition 3.3 (Martingale) *A sequence of random variables $Y_1, Y_2, Y_3, \dots$ is a martingale, if and only if $\mathbb{E}[|Y_i|] < +\infty$ and $\mathbb{E}[Y_i \mid Y_1, Y_2, \dots, Y_{i-1}] = Y_{i-1}$ for any i .*

To establish the connections between x_i ($i \in [1, \theta]$) and martingales, we first introduce two properties of *IMM*'s sampling phase. First, each R_i ($i \in [1, \theta]$) is generated for a node v selected uniformly at random, and it contains nodes that can reach v on a graph g sampled from $\mathcal{G}$. Second, the choices of v and g are independent of $R_1, R_2, \dots, R_{i-1}$ (although the decision to generate R_i depends on $R_1, R_2, \dots, R_{i-1}$). Due to these properties, for any $i \in [1, \theta]$,

$$\mathbb{E}[x_i \mid x_1, x_2, \dots, x_{i-1}] = \mathbb{E}[x_i] = \mathbb{E}[I(S)]/n.$$

Let $p = \mathbb{E}[I(S)]/n$ and $M_i = \sum_{j=1}^i (x_j - p)$. Then, we have $\mathbb{E}[M_i] = 0$, and

$$\mathbb{E}[M_i \mid M_1, M_2, \dots, M_{i-1}] = M_{i-1}.$$

Therefore, by Definition 3.3, $M_1, M_2, \dots, M_{\theta}$ is a martingale.

The following lemma from [17] shows two concentration results for martingales that have similar flavor to the Chernoff bounds.

Lemma 3.2 ([17]) *Let $Y_1, Y_2, Y_3, \dots$ be a martingale, such that $|Y_1| \leq a$, $|Y_j - Y_{j-1}| \leq a$ for any $j \in [1, i]$, and*

$$\text{Var}[Y_1] + \sum_{j=2}^i \text{Var}[Y_j \mid Y_1, Y_2, \dots, Y_{j-1}] \leq b,$$

where $\text{Var}[\cdot]$ denotes the variance of a random variable. Then, for any $\lambda > 0$,

$$\Pr \left[Y_i - \mathbb{E}[Y_i] \geq \gamma \right] \leq \exp \left(-\frac{\gamma^2}{\frac{2}{3}a\gamma + 2b} \right). \quad \square$$

Consider the martingale $M_1, M_2, \dots, M_\theta$. We have $|M_1| \leq 1$ and $|M_j - M_{j-1}| \leq 1$ for any $j \in [2, \theta]$. By the properties of R_i ($i \in [1, \theta]$), we also have

$$\begin{aligned} \text{Var}[M_1] + \sum_{j=2}^{\theta} \text{Var}[M_j \mid M_1, M_2, \dots, M_{j-1}] \\ = \sum_{j=1}^{\theta} \text{Var}[M_j] = \theta p \cdot (1 - p). \end{aligned}$$

Then, by Lemma 3.2 and $M_\theta = \sum_{i=1}^{\theta} (x_i - p)$, we have the following corollary.

Corollary 3.1 *For any $\varepsilon > 0$,*

$$\Pr \left[\sum_{i=1}^{\theta} x_i - \theta p \geq \varepsilon \cdot \theta p \right] \leq \exp \left(-\frac{\varepsilon^2}{2 + \frac{2}{3}\varepsilon} \cdot \theta p \right). \quad \square$$

In addition, by applying Lemma 3.2 on the martingale $-M_1, -M_2, \dots, -M_\theta$, we have the following result.

Corollary 3.2 *For any $\varepsilon > 0$,*

$$\Pr \left[\sum_{i=1}^{\theta} x_i - \theta p \leq -\varepsilon \cdot \theta p \right] \leq \exp \left(-\frac{\varepsilon^2}{2} \cdot \theta p \right). \quad \square$$

In Section 3.3.2, we will utilize the above corollaries to establish the approximation guarantees of *IMM*.

3.3.2 Node Selection Phase

Let $\mathcal{R} = \{R_1, R_2, \dots, R_\theta\}$ be the set of random RR sets generated by the sampling phase of *IMM*, and $F_{\mathcal{R}}(S)$ be the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S . Algorithm 1 shows the pseudo-code of the node selection phase of *IMM*, which takes as input $\mathcal{R}$ and a positive integer k , and returns a size- k node set S_k^* . The algorithm

Algorithm 1 NodeSelection ($\mathcal{R}, k$)

- 1: Initialize a node set $S_k^* = \emptyset$.
 - 2: **for** $i = 1$ to k **do**
 - 3: Identify the vertex v that maximizes $F_{\mathcal{R}}(S_k^* \cup v) - F_{\mathcal{R}}(S_k^*)$
 - 4: Insert v into S_k^*
 - 5: **end for**
 - 6: **return** S_k^*
-

corresponds to the standard greedy approach for the maximum coverage problem [59], which guarantees that $F_{\mathcal{R}}(S_k^*)$ is at least $(1 - 1/e)$ times the fraction of RR sets covered by any size- k node set. In what follows, we show that when the number θ of RR sets in $\mathcal{R}$ is sufficiently large, S_k^* ensures a $(1 - 1/e - \varepsilon)$ -approximation to the influence maximization problem with a high probability.

Consider the size- k node set S_k° with the maximum expected influence. Let $OPT = \mathbb{E}[I(S_k^\circ)]$. By Equation Eq. 3.1, $n \cdot F_{\mathcal{R}}(S_k^\circ)$ is an unbiased estimator of OPT . Then, by Corollary 3.2, $F_{\mathcal{R}}(S_k^\circ)$ should be close to OPT when θ is sizable, as shown in the following lemma.

Lemma 3.3 *Let $\delta_1 \in (0, 1)$, $\varepsilon_1 > 0$, and*

$$\theta_1 = \frac{2n \cdot \log(1/\delta_1)}{OPT \cdot \varepsilon_1^2}. \quad (\text{Eq. 3.2})$$

If $\theta \geq \theta_1$, then $n \cdot F_{\mathcal{R}}(S_k^\circ) \geq (1 - \varepsilon_1) \cdot OPT$ holds with at least $1 - \delta_1$ probability.

Proof: Define a random variable x_i for each $R_i \in \mathcal{R}$, such that $x_i = 1$ if $S_k^\circ \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. Then, $F_{\mathcal{R}}(S_k^\circ) = \sum_{i=1}^{\theta} x_i / \theta$. Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k^\circ)]$. By Lemma 3.1,

$$p = \mathbb{E}[F_{\mathcal{R}}(S_k^\circ)] = \mathbb{E}[I(S_k^\circ)] / n = OPT / n.$$

By Corollary 3.2,

$$\begin{aligned} & \Pr [n \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1 - \varepsilon_1) \cdot OPT] \\ &= \Pr [n \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1 - \varepsilon_1) \cdot np] \end{aligned}$$

$$\begin{aligned}
 &= \Pr [\theta \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1 - \varepsilon_1) \cdot \theta p] \\
 &= \Pr [\sum_{i=1}^{\theta} x_i - \theta p \leq -\varepsilon_1 \cdot \theta p] \\
 &\leq \exp \left(-\varepsilon_1^2 \cdot p\theta/2 \right) \leq \exp \left(-\varepsilon_1^2 \cdot p\theta_1/2 \right) \\
 &= \delta_1.
 \end{aligned}$$

Thus, the lemma is proved. ■

Suppose that $n \cdot F_{\mathcal{R}}(S_k^\circ) \geq (1 - \varepsilon_1) \cdot OPT$ holds. By the properties of the greedy approach,

$$\begin{aligned}
 n \cdot F_{\mathcal{R}}(S_k^*) &\geq (1 - 1/e) \cdot n \cdot F_{\mathcal{R}}(S_k^\circ) \\
 &\geq (1 - 1/e) \cdot (1 - \varepsilon_1) \cdot OPT.
 \end{aligned} \tag{Eq. 3.3}$$

Intuitively, Equation Eq. 3.3 indicates that the expected influence of S_k^* is likely to be large, since $n \cdot F_{\mathcal{R}}(S_k^*)$ is an indicator of $\mathbb{E}[I(S_k^*)]$. We formalize this intuition in the following lemma:

Lemma 3.4 *Let $\delta_2 \in (0, 1)$, $\varepsilon_1 < \varepsilon$, and*

$$\theta_2 = \frac{(2 - 2/e) \cdot n \cdot \log \left(\binom{n}{k} / \delta_2 \right)}{OPT \cdot (\varepsilon - (1 - 1/e) \cdot \varepsilon_1)^2}. \tag{Eq. 3.4}$$

If Equation Eq. 3.3 holds and $\theta \geq \theta_2$, then with at least $1 - \delta_2$ probability, $\mathbb{E}[I(S_k^)] \geq (1 - 1/e - \varepsilon) \cdot OPT$.*

Proof: Let S_k be an arbitrary size- k node set. We say that S_k is *bad*, if $\mathbb{E}[I(S_k)] < (1 - 1/e - \varepsilon) \cdot OPT$. To prove the lemma, we will show that each bad size- k node set has at most $\delta_2 / \binom{n}{k}$ probability to be returned by the node selection phase of *IMM*. This suffices to establish the lemma because (i) there exist only $\binom{n}{k}$ bad size- k node sets, and (ii) if each of them has at most $\delta_2 / \binom{n}{k}$ probability to be returned, then by the union bound, there is at least $1 - \delta_2$ probability that none of them is output by the node selection phase.

Consider any bad size- k node set S_k . Define a random variable x_i for each $R_i \in \mathcal{R}$, such that

$$x_i = \begin{cases} 1, & \text{if } S_k \cap R_i \neq \emptyset \\ 0, & \text{otherwise} \end{cases}.$$

We have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i / \theta$. Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n$ and $\varepsilon_2 = \varepsilon - (1 - 1/e) \cdot \varepsilon_1$.

We have

$$\begin{aligned} & \Pr [n \cdot F_{\mathcal{R}}(S_k) - \mathbb{E}[I(S_k)] \geq \varepsilon_2 \cdot OPT] \\ &= \Pr \left[\theta \cdot F_{\mathcal{R}}(S_k) - \theta p \geq \frac{\varepsilon_2 \cdot OPT}{np} \cdot \theta p \right] \end{aligned} \quad (\text{Eq. 3.5})$$

Let $\zeta = \frac{\varepsilon_2 \cdot OPT}{np}$. By Corollary 3.1 and $p < ((1 - 1/e - \varepsilon) \cdot OPT)/n$,

$$\begin{aligned} & \text{r.h.s of Eqn. Eq. 3.5} \\ & \leq \exp \left(-\frac{\zeta^2}{2 + \frac{2}{3}\zeta} \cdot \theta p \right) \\ &= \exp \left(-\frac{\varepsilon_2^2 \cdot OPT^2}{2n^2p + \frac{2}{3}\varepsilon_2 n \cdot OPT} \cdot \theta \right) \\ &< \exp \left(-\frac{\varepsilon_2^2 \cdot OPT^2}{2n(1 - 1/e - \varepsilon) \cdot OPT + \frac{2}{3}\varepsilon_2 n \cdot OPT} \cdot \theta \right) \\ &< \exp \left(-\frac{(\varepsilon - (1 - 1/e)\varepsilon_1)^2 \cdot OPT}{(2 - 2/e) \cdot n} \cdot \theta \right) \\ &\leq \exp \left(-\frac{(\varepsilon - (1 - 1/e)\varepsilon_1)^2 \cdot OPT}{(2 - 2/e) \cdot n} \cdot \theta_2 \right) \\ &\leq \delta_2 / \binom{n}{k}. \end{aligned}$$

Therefore, the lemma holds. ■

Lemmas 3.3 and 3.4 lead to the following theorem.

Theorem 3.1 *Given any $\varepsilon_1 \leq \varepsilon$ and any $\delta_1, \delta_2 \in (0, 1)$ with $\delta_1 + \delta_2 \leq 1/n^\ell$, setting $\theta \geq \max\{\theta_1, \theta_2\}$ ensures that the node selection phase of IMM returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - 1/n^\ell$ probability.*

Proof: By Lemma 3.4, $\mathbb{E}[I(S_k^*)] \geq (1 - 1/e - \varepsilon) \cdot OPT$ holds with at least $1 - \delta_2$ probability under the condition that Equation Eq. 3.3 holds. And by Lemma 3.3, Equation Eq. 3.3 holds with at least $1 - \delta_1$ probability. By the union bound, $\mathbb{E}[I(S_k^*)] \geq (1 - 1/e - \varepsilon) \cdot OPT$ holds with at least $1 - \delta_1 - \delta_2 \geq 1/n^\ell$ probability. Thus, the theorem is proved. $\blacksquare$

Parametrization. Assume that OPT is known. Then, by Theorem 3.1, we can obtain a $(1 - 1/e - \varepsilon)$ -approximation from IMM with at least $1 - 1/n^\ell$ probability as follows: We first choose ε_1 , δ_1 , and δ_2 according to the conditions stated in Theorem 3.1; after that, we calculate θ_1 and θ_2 based on Equations Eq. 3.2 and Eq. 3.4, respectively, and compute $\theta = \max\{\theta_1, \theta_2\}$; finally, we ensure that $\mathcal{R}$ contains at least θ RR sets. Here we meet a challenge of selecting ε_1 , δ_1 , and δ_2 to minimize θ . Since the relationships among ε_1 , δ_1 , δ_2 , and θ are rather complicated, it is difficult to obtain the optimal values of these parameters. Nevertheless, there exists a near-optimal θ that can be derived in a relatively simple means as shown below.

First, we set $\delta_1 = \delta_2 = 1/(2n^\ell)$. Under this setting, θ is minimized when $\theta_1 = \theta_2$. In turn, $\theta_1 = \theta_2$ holds if and only if $\varepsilon_1 = \varepsilon \cdot \frac{\alpha}{(1-1/e) \cdot \alpha + \beta}$, where

$$\begin{aligned} \alpha &= \sqrt{\ell \log n + \log 2}, \text{ and} \\ \beta &= \sqrt{(1 - 1/e) \cdot (\log \binom{n}{k} + \ell \log n + \log 2)}. \end{aligned} \tag{Eq. 3.6}$$

In that case,

$$\theta = \frac{2n \cdot ((1 - 1/e) \cdot \alpha + \beta)^2}{OPT \cdot \varepsilon^2}.$$

Denote the above θ as θ^* . The following lemma shows that θ^* is near-optimal with respect to Theorem 3.1:

Lemma 3.5 *Given ε, ℓ , let θ° be the smallest possible θ under the conditions stated in Theorem 3.1. We have*

$$\theta^\circ \leq \theta^* \leq \theta^\circ \cdot \frac{\ell \log n + \log 2}{\ell \log n}. \quad \square$$

Proof:

We rewrite θ_1 (resp. θ_2) as a function f_1 (resp. f_2) as follows:

$$\begin{aligned} f_1(\varepsilon_1, \delta_1) &= \frac{2n \cdot \log(1/\delta_1)}{OPT \cdot \varepsilon_1^2}, \text{ and} \\ f_2(\varepsilon_1, \delta_2) &= \frac{(2 - 2/e) \cdot n \cdot \log\left(\binom{n}{k}/\delta_2\right)}{OPT \cdot (\varepsilon - (1 - 1/e) \cdot \varepsilon_1)^2}. \end{aligned}$$

Observe that f_1 monotonically decreases with the increase of ε_1 or δ_1 , and f_2 monotonically increases with ε_1 but monotonically decreases with the increase of δ_2 . Assume that θ^o is obtained when $\varepsilon_1 = \varepsilon_1^o$, $\delta_1 = \delta_1^o$, and $\delta_2 = \delta_2^o$. Since θ^o is the smallest possible θ , the following equations hold:

$$\begin{aligned} f_1(\varepsilon_1^o, \delta_1^o) &= f_2(\varepsilon_1^o, \delta_2^o) = \theta^o, \text{ and} \\ \delta_1^o + \delta_2^o &= 1/n^\ell. \end{aligned}$$

Meanwhile, we have $\theta^* = f_1(\varepsilon_1^*, 1/(2n^\ell)) = f_2(\varepsilon_1^*, 1/(2n^\ell))$, where $\varepsilon_1^* = \varepsilon \cdot \frac{\alpha}{(1-1/e) \cdot \alpha + \beta}$.

Since $\theta^o \leq \theta$, we have

$$f_1(\varepsilon_1^o, \delta_1^o) \leq f_1(\varepsilon_1^*, 1/(2n^\ell)) \quad (\text{Eq. 3.7})$$

The remaining part of our proof consists of three steps. First, we prove that $f_1(\varepsilon'_1, 1/n^\ell) < f_1(\varepsilon_1^o, \delta_1^o)$, where ε'_1 is the positive real root of the equation $f_1(\varepsilon'_1, 1/n^\ell) = f_2(\varepsilon'_1, 1/n^\ell)$. For this purpose, we consider two cases based on whether $\varepsilon_1^o \leq \varepsilon'_1$. When $\varepsilon_1^o \leq \varepsilon'_1$, it can be verified that $f_1(\varepsilon'_1, 1/n^\ell) < f_1(\varepsilon_1^o, \delta_1^o)$ given $\delta_1^o < 1/n^\ell$. Meanwhile, if $\varepsilon_1^o > \varepsilon'_1$, we have

$$f_1(\varepsilon'_1, 1/n^\ell) = f_2(\varepsilon'_1, 1/n^\ell) < f_2(\varepsilon_1^o, \delta_2^o) = f_1(\varepsilon_1^o, \delta_1^o). \quad (\text{Eq. 3.8})$$

Second, we show that $M \leq \frac{\ell \log n + \log 2}{\ell \log n} \cdot f_1(\varepsilon'_1, 1/n^\ell)$, where $M = \max\{f_1(\varepsilon'_1, 1/(2n^\ell)), f_2(\varepsilon'_1, 1/(2n^\ell))\}$.

We have

$$\begin{aligned} \frac{f_1(\varepsilon'_1, 1/(2n^\ell))}{f_1(\varepsilon'_1, 1/n^\ell)} &= \frac{\ell \log n + \log 2}{\ell \log n}, \text{ and} \\ \frac{f_2(\varepsilon'_1, 1/(2n^\ell))}{f_1(\varepsilon'_1, 1/n^\ell)} &= \frac{f_2(\varepsilon'_1, 1/(2n^\ell))}{f_2(\varepsilon'_1, 1/n^\ell)} \end{aligned}$$

$$\begin{aligned}
 &= \frac{\ell \log n + \log \binom{n}{k} + \log 2}{\ell \log n + \log \binom{n}{k}} \\
 &< \frac{\ell \log n + \log 2}{\ell \log n}.
 \end{aligned}$$

It then follows that

$$M \leq \frac{\ell \log n + \log 2}{\ell \log n} \cdot f_1(\varepsilon'_1, 1/n^\ell). \quad (\text{Eq. 3.9})$$

Finally, we prove that $f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq M$. We consider two cases based on whether $\varepsilon_1^* \geq \varepsilon'_1$. When $\varepsilon_1^* \geq \varepsilon'_1$, we have

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq f_1(\varepsilon'_1, 1/(2n^\ell)) \leq M.$$

Meanwhile, if $\varepsilon_1^* < \varepsilon'_1$, we have

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) = f_2(\varepsilon_1^*, 1/(2n^\ell)) \leq f_2(\varepsilon'_1, 1/(2n^\ell)) \leq M.$$

Therefore, the following equation holds

$$f_1(\varepsilon_1^*, 1/(2n^\ell)) \leq M. \quad (\text{Eq. 3.10})$$

Combining Equations Eq. 3.7, Eq. 3.8, Eq. 3.9 and Eq. 3.10, we have

$$f_1(\varepsilon_1^o, \delta_1^o) \leq f_1(\varepsilon_1^*, 1/(2n^\ell)) < \frac{\ell \log n + \log 2}{\ell \log n} \cdot f_1(\varepsilon_1^o, \delta_1^o).$$

Thus, the lemma is proved. ■

In summary, the node selection phase of *IMM* returns a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability, if we ensure that the number θ of RR sets in $\mathcal{R}$ is at least λ^*/OPT , where

$$\lambda^* = 2n \cdot ((1 - 1/e) \cdot \alpha + \beta)^2 \cdot \varepsilon^{-2}. \quad (\text{Eq. 3.11})$$

Nevertheless, ensuring $\theta \geq \lambda^*/OPT$ is non-trivial, since OPT is unknown in advance. In Section 3.3.3, we address this issue by setting $\theta = \lambda^*/LB$, where LB is a lower bound of OPT that is almost tight.

Finally, we have the following result regarding the time complexity of Algorithm 1.

 Figure 3.3: λ^*/λ vs. ε .

 Figure 3.4: λ^*/λ vs. k .

Corollary 3.3 *Algorithm 1 runs in $O(\sum_{R \in \mathcal{R}} |R|)$ time.*

The above corollary follows from the fact that Algorithm 1 corresponds to the standard greedy approach for maximum coverage, which runs in time linear to the total size of its input [59].

Comparison with the Node Selection Phase of TIM and TIM⁺. Previous work [58] utilizes Algorithm 1 as part of the node selection phase of TIM and TIM⁺, and establishes two conditions for Algorithm 1 to return a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability. First, all random RR sets in $\mathcal{R}$ should be independent. Second, $\theta \geq \lambda/OPT$, where

$$\lambda = (8 + 2\varepsilon)n \cdot (\ell \log n + \log \binom{n}{k} + \log 2) \cdot \varepsilon^{-2}. \quad (\text{Eq. 3.12})$$

In comparison, our results are more general in that we allow dependencies among the RR sets in $\mathcal{R}$. Furthermore, our analysis leads to a θ that is near-optimal with respect to Theorem 3.1, whereas the θ given in [58] roughly corresponds to a suboptimal case in Theorem 3.1 with $\delta_1 = 1/(n^\ell \cdot \binom{n}{k})$, $\delta_2 = 1 - \delta_1$, and $\varepsilon_1 = \varepsilon/2$.

To illustrate the differences between our results and those in [58], Figure 3.3 compares λ^* (in Equation Eq. 3.11) and λ (in Equation Eq. 3.12) under a realistic setting where $n = 10^6$, $\ell = 1$, $k = 50$, and ε varies from 0.1 to 0.5. Observe that $\lambda^* \leq 0.2\lambda$ in all cases.

Figure 3.4 shows λ^*/λ as a function of k , with $\varepsilon = 0.1$, $n = 10^6$, and $\ell = 1$. Again, λ^* is considerably smaller than λ , regardless of the value of k .

3.3.3 Sampling Phase

Recall that node selection phase of *IMM* requires an input set $\mathcal{R}$ of random RR sets whose cardinality $|\mathcal{R}|$ is no less than λ^*/OPT , where λ^* is as defined in Equation Eq. 3.11. Since OPT is an unknown value, we aim to calculate a lower bound LB of OPT and then set $|\mathcal{R}| = \lambda^*/LB \geq \lambda^*/OPT$. Intuitively, the gap between LB and OPT should be as small as possible, in order to minimize the computing overheads of generating $\mathcal{R}$. Nevertheless, it is a tough challenge to design such an algorithm yielding a satisfactory LB , since the estimation of OPT is influence maximization problem itself involving NP-hard computation.

Our basic idea to address the above challenge is to construct a statistical test $B(x)$. The equation $B(x) = \text{false}$ holds with a high probability if $OPT < x$. Therefore, we can easily identify a lower bound of OPT by running $B(x)$ on $O(\log n)$ values of x , e.g., $x = n/2, n/4, n/8, \dots, 2$. (Note that $OPT \in [1, n]$.) Interestingly, an algorithm in the node selection phase of *IMM* has provided an approach to implement $B(x)$, as shown in the following lemma:

Lemma 3.6 *Let $x \in [1, n]$ and $\varepsilon', \delta_3 \in (0, 1)$. Suppose that we call Algorithm 1 on a set $\mathcal{R}$ containing θ random RR sets, with*

$$\theta \geq \frac{(2 + \frac{2}{3}\varepsilon') \cdot (\log \binom{n}{k} + \log(1/\delta_3))}{\varepsilon'^2} \cdot \frac{n}{x}. \quad (\text{Eq. 3.13})$$

Let S_k^ be the returned result of Algorithm 1. Then, if $OPT < x$, the equation $n \cdot F_{\mathcal{R}}(S_k^*) < (1 + \varepsilon') \cdot x$ holds with at least $1 - \delta_3$ probability.*

Proof: Let S_k be an arbitrary size- k node set and $p = \mathbb{E}[F_{\mathcal{R}}(S_k)]$. Define a random variable x_i for each $R_i \in \mathcal{R}$ such that $x_i = 1$ if $S_k \cap R_i \neq \emptyset$, and $x_i = 0$ otherwise. Then,

we have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i / \theta$ and

$$p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n \leq OPT/n < x/n. \quad (\text{Eq. 3.14})$$

let $\zeta = \frac{(1+\varepsilon') \cdot x}{np} - 1$. By Equation Eq. 3.14, we have $\zeta > \varepsilon' \cdot x/(np) > \varepsilon'$. Then, by Corollary 3.1,

$$\begin{aligned} & \Pr[n \cdot F_{\mathcal{R}}(S_k) \geq (1 + \varepsilon') \cdot x] \\ &= \Pr \left[\theta \cdot F_{\mathcal{R}}(S_k) - \theta p \geq \left(\frac{(1 + \varepsilon') \cdot x}{np} - 1 \right) \cdot \theta p \right] \\ &\leq \exp \left(-\frac{\zeta^2}{2 + 2\zeta/3} \cdot \theta p \right) \\ &< \exp \left(-\frac{\varepsilon' \cdot x/(np)}{2/\zeta + 2/3} \cdot \frac{(2 + \frac{2}{3}\varepsilon') \cdot \log(\binom{n}{k}/\delta_3)}{\varepsilon'^2} \cdot \frac{np}{x} \right) \\ &< \exp \left(-\frac{1}{2/\varepsilon' + 2/3} \cdot \frac{(2 + \frac{2}{3}\varepsilon') \cdot \log(\binom{n}{k}/\delta_3)}{\varepsilon'} \right) \\ &= \delta_3 / \binom{n}{k}. \end{aligned}$$

By the union bound, we can have $n \cdot F_{\mathcal{R}}(S_k^*) < (1 + \varepsilon') \cdot x$ with at least $1 - \delta_3$ probability. Thus, the lemma is proved. ■

The rationale of Lemma 3.6 is as follows. First, if OPT is large, then Algorithm 1 is likely to return a node set S_k^* that covers a large portion of RR sets in $\mathcal{R}$, i.e., $F_{\mathcal{R}}(S_k^*)$ should be large. Conversely, if we observe that $F_{\mathcal{R}}(S_k^*)$ is small, then OPT is also likely to be small (modulo the sampling error in $\mathcal{R}$). Therefore, we can use $F_{\mathcal{R}}(S_k^*)$ as an indicator of the magnitude of OPT .

Based on Lemma 3.6, the pseudo-code of a sampling phase of *IMM* is demonstrated in Algorithm 2. The algorithm takes four input arguments G , k , ε , and ℓ , then initializes a set $\mathcal{R} = \emptyset$ and an integer $LB = 1$ (Line 1). After that, it sets a parameter $\varepsilon' = \sqrt{2} \cdot \varepsilon$, and enters a for loop with at most $\log_2 n - 1$ iterations (Lines 3 - 12). (Our choice of ε' is to be explained shortly.)

Algorithm 2 Sampling $(G, k, \varepsilon, \ell)$

```

1: Initialize a set  $\mathcal{R} = \emptyset$  and an integer  $LB = 1$ 
2: Let  $\varepsilon' = \sqrt{2} \cdot \varepsilon$ 
3: for  $i = 1$  to  $\log_2 n - 1$  do
4:   Let  $x = n/2^i$ 
5:   Let  $\theta_i = \lambda'/x$ , where  $\lambda'$  is as defined in Equation Eq. 3.15
6:   while  $|\mathcal{R}| \leq \theta_i$  do
7:     Select a node  $v$  from  $G$  uniformly at random
8:     Generate an RR set for  $v$ , and insert it into  $\mathcal{R}$ 
9:   end while
10:  Let  $S_i = \text{NodeSelection}(\mathcal{R})$ 
11:  if  $n \cdot F_{\mathcal{R}}(S_i) \geq (1 + \varepsilon') \cdot x$  then
12:     $LB = n \cdot F_{\mathcal{R}}(S_i) / (1 + \varepsilon')$ 
13:    break
14:  end if
15: end for
16: Let  $\theta = \lambda^*/LB$ , where  $\lambda^*$  is as defined in Equation Eq. 3.11
17: while  $|\mathcal{R}| \leq \theta$  do
18:   Select a node  $v$  from  $G$  uniformly at random
19:   Generate an RR set for  $v$ , and insert it into  $\mathcal{R}$ 
20: end while
21: return  $\mathcal{R}$ 

```

In the i -th iteration, the algorithm computes $x = n/2^i$, and then derives $\theta_i = \lambda'/x$, where

$$\lambda' = \frac{(2 + \frac{2}{3}\varepsilon') \cdot (\log \binom{n}{k} + \ell \cdot \log n + \log \log_2 n) \cdot n}{\varepsilon'^2}. \quad (\text{Eq. 3.15})$$

It can be proved that θ_i equals the smallest θ that satisfies Equation Eq. 3.13 given $\delta_3 = n^\ell \cdot \log_2 n$. In the next step, the algorithm inserts random RR sets into $\mathcal{R}$ until its cardinality meets the condition $|\mathcal{R}| = \theta_i$ (Lines 6-8), after which it invokes Algorithm 1 on $\mathcal{R}$ to obtain a size- k node set S_i . By Lemma 3.6, if

$$n \cdot F_{\mathcal{R}}(S_i) \geq (1 + \varepsilon') \cdot x, \quad (\text{Eq. 3.16})$$

then $OPT \geq x$ should hold with at least $1 - 1/(n^\ell \cdot \log_2 n)$ probability. In other words, once Equation Eq. 3.16 holds, we can terminate the for loop and identify x as a lower bound of OPT . This explains Line 10-12 in Algorithm 2, except that the algorithm uses

$LB = n \cdot F_{\mathcal{R}}(S_i)/(1 + \varepsilon')$ as a lower bound of OPT , instead of setting $LB = x$. (Notice that the former is a tighter lower bound, due to Equation Eq. 3.16.) Our choice of LB is established on the following lemma:

Lemma 3.7 *Define $x, \varepsilon', \delta_3, \mathcal{R}$, and S_k^* same as ones in Lemma 3.6. If $OPT \geq x$, then we have $OPT \geq n \cdot F_{\mathcal{R}}(S_k^*)/(1 + \varepsilon')$ with at least $1 - \delta_3$ probability.*

Proof: Let S_k be an arbitrary size- k node set. We say that S_k is *bad*, if $OPT < n \cdot \mathbb{E}[F_{\mathcal{R}}(S_k)]/(1 + \varepsilon')$. To prove the lemma, we will demonstrate that each bad size- k node set can be returned by Algorithm 1 with at most $\delta_3/\binom{n}{k}$ probability. This suffices to establish the lemma because (i) there exist only $\binom{n}{k}$ bad size- k node sets, and (ii) if each of them has at most $\delta_3/\binom{n}{k}$ probability to be returned, then by the union bound, there is at least $1 - \delta_3$ probability that none of them is output by the node selection phase.

Consider any bad size- k node set S_k . Let x_i be a random variable for each $R_i \in \mathcal{R}$, satisfying

$$x_i = \begin{cases} 1, & \text{if } S_k \cap R_i \neq \emptyset \\ 0, & \text{otherwise} \end{cases}.$$

We have $F_{\mathcal{R}}(S_k) = \sum_{i=1}^{\theta} x_i/\theta$. Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k)] = \mathbb{E}[I(S_k)]/n$. We have

$$\begin{aligned} & \Pr [OPT < n \cdot F_{\mathcal{R}}(S_k)/(1 + \varepsilon')] \\ & \leq \Pr \left[\theta \cdot F_{\mathcal{R}}(S_k) - \theta p \geq \frac{\varepsilon' \cdot OPT}{np} \cdot \theta p \right] \end{aligned} \quad (\text{Eq. 3.17})$$

Let $\zeta = \frac{\varepsilon' \cdot OPT}{np}$. By Corollary 3.1 and $OPT \geq x$,

$$\begin{aligned} & \text{r.h.s of Eqn. Eq. 3.17} \\ & \leq \exp \left(-\frac{\zeta^2}{2 + \frac{2}{3}\zeta} \cdot \theta p \right) \\ & = \exp \left(-\frac{\varepsilon'^2 \cdot OPT^2}{2n^2p + \frac{2}{3}\varepsilon'n \cdot OPT} \cdot \theta \right) \\ & < \exp \left(-\frac{\varepsilon'^2 \cdot OPT}{2n + \frac{2}{3}\varepsilon'n} \cdot \frac{2 + \frac{2}{3}\varepsilon'}{\varepsilon'^2} \cdot \frac{n}{x} \cdot \log(\binom{n}{k}/\delta_3) \right) \end{aligned}$$

$$\begin{aligned}
 &< \exp\left(-(\text{OPT}/x) \cdot \log\left(\binom{n}{k}/\delta_3\right)\right) \\
 &\leq \delta_3/\binom{n}{k}.
 \end{aligned}$$

Therefore, the lemma holds. ■

The above discussions assume that Equation Eq. 3.16 holds in a certain iteration of the for loop in Algorithm 2. On the other hand, if Equation Eq. 3.16 does not hold in any of the $\log_2 n - 1$ iterations, then the algorithm sets $LB = 1$. In either case, once LB is determined, the algorithm computes $\theta = \lambda^*/LB$, where λ^* is as defined in Equation Eq. 3.11, and then it adds more random RR sets into $\mathcal{R}$ until $|\mathcal{R}| = \theta$ (Lines 13-16). Finally, the algorithm terminates by returning $\mathcal{R}$.

Theoretical Analysis. Let $j = \lceil \log_2 \frac{n}{\text{OPT}} \rceil$. By Lemma 3.6 and considering the union bound, Algorithm 2 can terminate its for loop before the j -th iteration with at most $(j - 1)/(n^\ell \cdot \log_2 n)$ probability. Then, according to Lemma 3.7 and taking the consideration of the union bound, Algorithm 2 has at least $1 - n^\ell$ probability to set $LB \leq \text{OPT}$. Therefore, we have the following result.

Theorem 3.2 *With at least $1 - n^\ell$ probability, Algorithm 2 returns a set $\mathcal{R}$ of RR sets with $|\mathcal{R}| \geq \lambda^*/\text{OPT}$, where λ^* is as defined in Equation Eq. 3.11.*

Proof: Let $j = \lceil \log_2 \frac{n}{\text{OPT}} \rceil$. By Lemma 3.6 and the union bound, Algorithm 2 has at most $(j - 1)/(n^\ell \cdot \log_2 n)$ probability to terminate its for loop before the j -th iteration. Then, by Lemma 3.7 and the union bound, Algorithm 2 sets $LB \leq \text{OPT}$ with at least $1 - n^\ell$ probability. Therefore, the theorem holds. ■

In addition, we can show that LB is close to OPT with a high probability. Towards this end, we provide a proof showing Algorithm 2 inclines to stop its for loop soon after the j -th iteration, as shown in the following lemma.

Lemma 3.8 *Let x , ε' , δ_3 , $\mathcal{R}$, and S_k^* be as defined in Lemma 3.6. Let $c \geq 1$. If $\text{OPT} \geq c \cdot \frac{(1+\varepsilon')^2}{1-1/e} \cdot x$, then $n \cdot F_{\mathcal{R}}(S_k^*) < (1 + \varepsilon') \cdot x$ holds with at most $(\delta_3)^c$ probability.*

Proof: Let S_k^+ be the set of k nodes that maximizes $F_{\mathcal{R}}(S_k^+)$ among all size- k node set. According to the properties of the greedy algorithm for maximum coverage, we trivially obtain

$$F_{\mathcal{R}}(S_k^*) \geq (1 - 1/e) \cdot F_{\mathcal{R}}(S_k^+) \geq (1 - 1/e) \cdot F_{\mathcal{R}}(S_k^\circ).$$

Let $p = \mathbb{E}[F_{\mathcal{R}}(S_k^\circ)]$. We have $p = OPT/n$. Since $OPT \geq c \cdot \frac{(1+\varepsilon')^2}{1-1/e} \cdot x$,

$$\begin{aligned} \Pr[n \cdot F_{\mathcal{R}}(S_k^*) \leq (1 + \varepsilon') \cdot x] & \leq \Pr[n(1 - 1/e) \cdot F_{\mathcal{R}}(S_k^\circ) \leq (1 + \varepsilon') \cdot x] \\ & \leq \Pr[n \cdot F_{\mathcal{R}}(S_k^\circ) \leq OPT/(1 + \varepsilon')] \\ & \leq \Pr[\theta_i \cdot F_{\mathcal{R}}(S_k^\circ) - \theta_i p \leq -\varepsilon' \cdot \theta_i p / (1 + \varepsilon')] \end{aligned} \quad (\text{Eq. 3.18})$$

Let $\zeta = \varepsilon'/(1 + \varepsilon')$. By Corollary 3.2,

$$\begin{aligned} \text{r.h.s of Eqn. Eq. 3.18} & \leq \exp\left(-\frac{\zeta^2}{2} \cdot \theta_i p\right) \\ & = \exp\left(-\frac{\varepsilon'^2}{2 \cdot (1 + \varepsilon')^2} \cdot \frac{2 + 2\varepsilon'/3}{\varepsilon'^2} \cdot \frac{np}{x} \cdot \log((\binom{n}{k})/\delta_3)\right) \\ & = \exp\left(-\frac{1 + \varepsilon'/3}{(1 + \varepsilon')^2} \cdot \frac{OPT}{x} \cdot \log((\binom{n}{k})/\delta_3)\right) \\ & \leq \exp\left(-\frac{1 + \varepsilon'/3}{(1 + \varepsilon')^2} \cdot \frac{(1 + \varepsilon')^2}{1 - 1/e} \cdot \log((\binom{n}{k})/\delta_3) \cdot c\right) \\ & < (\delta_3/\binom{n}{k})^c \end{aligned}$$

By the union bound, for all possible size- k node set S_k^* we have $n \cdot F_{\mathcal{R}}(S_k^*) \leq (1 + \varepsilon') \cdot x$ holds with at most $(\delta_3)^c$ probability. Thus, the lemma is proved. $\blacksquare$

Based on Lemma 3.8, we have the following result regarding LB .

Lemma 3.9 *Let $\mathcal{R}$ be the output of Algorithm 2, and LB be as in Line 13 of Algorithm 2.*

We have $LB \geq \frac{1-1/e}{(1+\varepsilon')^2} \cdot OPT$ with at least $1 - 1/n^\ell$ probability. In addition, when $1/n^\ell \leq 1/2$,

$$\mathbb{E}[|\mathcal{R}|] \leq \frac{3 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT},$$

where λ' is as defined in Equation Eq. 3.15.

Proof: Assume that $OPT \geq \frac{(1+\varepsilon')^2}{1-1/e} \cdot x$ first holds in the j -th iteration. Let $z \geq j$, and $c = 2^{z-j}$. Then, in the z -th iteration, we have $OPT \geq c \cdot \frac{(1+\varepsilon')^2}{1-1/e} \cdot \frac{n}{2^z}$. Let I_z be the event that IMM executes the z -th iteration of its for loop. By Lemma 3.8,

$$P[I_z] \leq 1/(n^\ell \cdot \log_2 n)^c \leq 1/(n^\ell \cdot \log_2 n)$$

Taking into account all iterations from the j -th one to the $(\log_2 n - 1)$ -th one, we have

$$\sum_{z \geq j} P[I_z] \leq 1/n^\ell$$

Thus, Algorithm 2 terminates before the j -th iteration with at least $1 - 1/n^\ell$ probability.

In that case, we have $OPT \leq \frac{(1+\varepsilon')^2}{1-1/e} \cdot x$, which leads to

$$LB = n \cdot F_{\mathcal{R}}(S_i)/(1 + \varepsilon') \geq x \geq \frac{1 - 1/e}{(1 + \varepsilon')^2} \cdot OPT.$$

Meanwhile, the expected total number of RR sets generated in the first $j - 1$ iterations is at most

$$\begin{aligned} & \max \left\{ \frac{\lambda^*}{LB}, \frac{\lambda'}{n/2^{j-1}} \right\} \cdot (1 - \sum_{z \geq j} P[I_z]) \\ & \leq \frac{2 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT} \cdot (1 - 1/n^\ell) \\ & < \frac{2 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT}. \end{aligned} \tag{Eq. 3.19}$$

In addition, when $1/n^\ell \leq 1/2$, the expected total number of RR sets from the j -th to the $(\log_2 n - 1)$ -th iteration is at most

$$\begin{aligned} & \sum_{z \geq j} \max \left\{ \frac{\lambda^*}{LB}, \frac{\lambda'}{n/2^z} \right\} \cdot P[I_z] \\ & = \sum_{z \geq j} \frac{\max\{\lambda^*, \lambda'\}}{n/2^z} \cdot P[I_z] \\ & \leq \frac{1}{\log_2 n} \cdot \sum_{z \geq j} \frac{\max\{\lambda^*, \lambda'\}}{(n^\ell)^c \cdot n/2^z} \\ & \leq \frac{\max\{\lambda^*, \lambda'\}}{n \cdot \log_2 n} \cdot \sum_{z \geq j} 2^{-2^{z-j}+z} \end{aligned}$$

$$\begin{aligned}
 &< \frac{\max\{\lambda^*, \lambda'\}}{n \cdot \log_2 n} \cdot \log_2 n \\
 &\leq \frac{\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT}.
 \end{aligned} \tag{Eq. 3.20}$$

Combining Equations Eq. 3.19 and Eq. 3.20, we have

$$\mathbb{E}[|\mathcal{R}|] \leq \frac{3 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT},$$

which proves the lemma. ■

In Section 3.3.4, we will utilize Lemma 3.9 to analyze the total running time of *IMM* and compare it with that of TIM and TIM⁺ [58].

Parametrization. It remains to explain why we set $\varepsilon' = \sqrt{2} \cdot \varepsilon$ in Algorithm 2. As shown in Lemma 3.9, the expected number of RR sets generated by Algorithm 2 is bounded by a quantity proportional to $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$. Ideally, we should choose an ε' that minimizes $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$, but it is difficult given the complex expressions of λ^* and λ' . As an alternative, we roughly approximate $\max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2$ with a simple function of ε' and ε , and then derive $\varepsilon' = \sqrt{2} \cdot \varepsilon$ as a minimizer of the function. Our experiments show that this choice of ε' leads to reasonably good performance of *IMM*.

3.3.4 Putting It Together

In summary, *IMM* first invokes Algorithm 2 to generate a set $\mathcal{R}$ of RR sets, and then feeds $\mathcal{R}$ to Algorithm 1 to obtain a size- k node set S_k^* as the final output. By Theorems 3.1 and 3.2 as well as the union bound, *IMM* returns a $(1 - 1/e - \varepsilon)$ -approximate solution to influence maximization with at least $1 - 2/n^\ell$ probability. The success probability of *IMM* can be easily improved to $1 - 1/n^\ell$, by increasing ℓ by a factor of $1 + \log 2 / \log n$. Algorithm 3 shows the pseudo-code of a version of *IMM* with the improved success probability.

In what follows, we discuss the time complexity of *IMM*. We focus on the sampling phase of *IMM* (i.e., Algorithm 2), since the time complexity of the node estimation

Algorithm 3 IMM (G, k, ε, ℓ)

```

1:  $\ell = \ell \cdot (1 + \log 2 / \log n)$ 
2:  $\mathcal{R} = \text{Sampling} ( G, k, \varepsilon, \ell )$ 
3:  $S_k^* = \text{NodeSelection} ( \mathcal{R}, k )$ 
4: return  $S_k^*$ 
    
```

phase (i.e., Algorithm 1) has been established in Corollary 3.3. Observe that the major computational cost of Algorithm 2 is incurred by the construction of RR sets. On the other hand, the time required in generating a random RR set R is $O(w(R))$, where $w(R)$ denotes the number of edges in G that point to the nodes in R . Specifically, the generation of R requires (i) sampling an edge set from the triggering distribution of each node in R and (ii) traversing all edges in the edge sets sampled. The former requires $O(w(R))$ time, since a sample from a node v 's triggering distribution incurs a linear cost to the number of incoming edges of v . Meanwhile, the latter takes $O(w(R))$ time, since the sampled edge sets contain $O(w(R))$ time in total.

Let EPT be the expected value of $w(R)$. We have the following result from [58].

Lemma 3.10 ([58]) $n \cdot EPT \leq m \cdot OPT$.

Now consider the set $\mathcal{R}$ of RR sets generated by Algorithm 2. By Lemma 3.9,

$$\begin{aligned}
 \mathbb{E}[|\mathcal{R}|] &\leq \frac{3 \max\{\lambda^*, \lambda'\} \cdot (1 + \varepsilon')^2}{(1 - 1/e) \cdot OPT} \\
 &= O((k + \ell)n \log n \cdot \varepsilon^{-2} / OPT). \tag{Eq. 3.21}
 \end{aligned}$$

If $|\mathcal{R}|$ is independent of $w(R)$ for each $R \in \mathcal{R}$, then by Lemma 3.10 and Equation Eq. 3.21, the expected total time to generate $\mathcal{R}$ is

$$\begin{aligned}
 \mathbb{E}\left[\sum_{R \in \mathcal{R}} w(R)\right] &= \mathbb{E}[|\mathcal{R}|] \cdot EPT \\
 &= O((k + \ell)(n + m) \log n / \varepsilon^2). \tag{Eq. 3.22}
 \end{aligned}$$

However, $|\mathcal{R}|$ is correlated with the RR sets in $\mathcal{R}$, because $|\mathcal{R}|$ depends on when Algorithm 2 terminates its for loop, which in turn is decided by the RR sets in $\mathcal{R}$. But

interestingly, we can prove that Equation Eq. 3.22 still holds, by utilizing the *optional stopping theorem* [63] for martingales. For our discussion, we introduce a simplified version of the theorem² as follows.

Theorem 3.3 ([63]) *Let $Y_1, Y_2, Y_3, \dots$ be a martingale, and τ be a random variable such that the event $\tau = i$ is independent of any Y_j with $j > i$. If $\Pr[\tau < +\infty] = 1$, then*

$$\mathbb{E}[Y_\tau] = \mathbb{E}[Y_1]. \quad (\text{Eq. 3.23})$$

An intuitive way to understand Theorem 3.3 is as follows. Suppose that we retrieve $Y_1, Y_2, \dots$ one by one, and we have a certain algorithm that examines the Y_i that we obtain and decides when we should stop the retrieval process. (That is, the algorithm determines the value of τ .) Then, as long as the algorithm always terminates in finite time, the expectation of the last retrieved Y_i is equal to $\mathbb{E}[Y_1]$.

Based on Theorem 3.3, we prove Equation Eq. 3.22 as follows. First, let R_i be the i -th RR set generated by Algorithm 2, and $Z_i = \sum_{j=1}^i (w(R_j) - EPT)$. It can be verified that $Z_1, Z_2, \dots$ is a martingale. Let τ be a random variable that denotes the total number of RR sets generated by Algorithm 3. Observe that $\tau < +\infty$ always holds. Thus, by Theorem 3.3, $\mathbb{E}[Z_\tau] = \mathbb{E}[Z_1]$. Notice that

$$\begin{aligned} \mathbb{E}[Z_1] &= \mathbb{E}[w(R_1) - EPT] = 0, \text{ and} \\ \mathbb{E}[Z_\tau] &= \mathbb{E}\left[\sum_{j=1}^{\tau} (w(R_j) - EPT)\right] \\ &= \mathbb{E}\left[\sum_{j=1}^{\tau} w(R_j)\right] - \mathbb{E}[\tau] \cdot EPT \\ &= \mathbb{E}\left[\sum_{R \in \mathcal{R}} w(R)\right] - \mathbb{E}[|\mathcal{R}|] \cdot EPT. \end{aligned}$$

Therefore, Equation Eq. 3.22 holds.

With the above discussions, we can prove that the expected running time of *IMM* equals the right hand side of Equation Eq. 3.22. This leads to the following theorem.

²In its original form, the optional stopping theorem states three sufficient conditions for Equation Eq. 3.23. For simplicity, we include only one condition in Theorem 3.3, but it suffices for our purpose.

Theorem 3.4 *Algorithm 3 returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - 1/n^\ell$ probability, and runs in $O((k + \ell)(n + m) \log n / \varepsilon^2)$ expected time.*

Proof: By combining Theorems 3.1 and 3.2, we obtain that Algorithm 3 returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - 1/n^\ell$. In the following, we focus on *IMM*'s time complexity. Observe that the computation cost of *IMM* mainly consists of two parts, namely, the cost of generating the set $\mathcal{R}$ of random RR sets, and the cost of invoking Algorithm 1. As discussed in Section 3.3.4, the expected total cost of constructing $\mathcal{R}$ is $O((k + \ell)(n + m) \log n / \varepsilon^2)$. Therefore, it suffices to prove that the total cost incurred by Algorithm 1 is no more than $O((k + \ell)(n + m) \log n / \varepsilon^2)$.

Observe that *IMM* invokes Algorithm 1 at most $\log_2 n$ times, i.e., once in each iteration of the for loop in Algorithm 2, and once for the node selection phase. Let R_i be the version of $\mathcal{R}$ in the i -th iteration of Algorithm 2. By Corollary 3.3, when given $\mathcal{R}_i$ as input, Algorithm 1 runs in $O(\sum_{R \in \mathcal{R}_i} |R|)$ time. Notice that $|\mathcal{R}_i|$ doubles after each iteration, i.e., $|\mathcal{R}_1|, |\mathcal{R}_2|, \dots$ is a geometric sequence with a common ratio 2. Combining this with Theorem 3.3, it can be verified that the expected total cost incurred by invoking Algorithm 1 in Algorithm 2 is $O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|])$. On the other hand, the node selection phase executes Algorithm 1 once on the $\mathcal{R}$ generated by Algorithm 2, which takes $O(\sum_{R \in \mathcal{R}} |R|)$ time. In summary, the expected total cost incurred by Algorithm 1 is $O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|])$.

Note that $|R| \leq w(R)$ for any $R \in \mathcal{R}$. Then, by Equation Eq. 3.22,

$$\begin{aligned} O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|]) &\leq O(\mathbb{E}[\sum_{R \in \mathcal{R}} w(R)]) \\ &= O((k + \ell)(n + m) \log n / \varepsilon^2), \end{aligned}$$

which proves the theorem. ■

Comparison with TIM and TIM⁺. *IMM*, TIM, and TIM⁺ have the same approximation guarantee and expected time complexity, but *IMM* provides much higher efficiency

in practice since it requires generating a much smaller number of RR sets. Specifically, the number of RR sets constructed in *IMM* is $O((k + \ell)n \log n \cdot \varepsilon^{-2}/OPT)$, whereas TIM requires $O((k + \ell)n \log n \cdot \varepsilon^{-2}/KPT)$ RR sets, where *KPT* is the expected influence of a node set obtained by sampling k nodes with replacement from G , with the sampling probability of each node being proportional to its in-degree [58]. It can be verified that $KPT < OPT$ always holds, and $KPT/OPT = O(k/n)$ in the worst case. Therefore, TIM incurs a much larger overhead than *IMM*. Meanwhile, although TIM⁺ improves upon TIM in terms of efficiency, its parameter estimation phase is identical to that of TIM, due to which it also requires producing a large number of RR sets to estimate *KPT*. As a consequence, TIM⁺'s practical performance is much inferior to that of *IMM*, as we show in Section 5.1.

In addition, there exists another crucial difference between *IMM* and TIM (resp. TIM⁺). In *IMM*, all RR sets generated in the sampling phase are re-used in the node selection phase. In contrast, TIM's node selection phase is unable to recycle any RR sets produced in the parameter estimation phase, due to which it has to re-generate a large number of RR sets from scratch for influence maximization. The reason is that the node selection phase of TIM demands that all RR sets used should be independent, whereas the RR sets produced in TIM's parameter estimation phase are correlated. This results in a considerable amount of redundant computation, which degrades TIM's efficiency. TIM⁺ suffers from the same issue since its node selection and parameter estimation phases are the same as those of TIM. Meanwhile, *IMM* circumvents this issue because its node selection phase accommodates dependencies among RR sets, which in turn is due to the martingale approach adopted in *IMM*.

3.4 Extensions

This section extends *IMM* beyond the triggering model, and demonstrates an application of *IMM* to the continuous-time model [25], a diffusion model not supported by TIM and TIM⁺ [57]

3.4.1 Generalization of *IMM*

Our discussions in the previous sections have assumed the triggering model. However, observe that the algorithms used by *IMM* (i.e., Algorithms 1, 2, and 3) do not rely on anything specific to the triggering model, except that they require a method to generate random RR sets. To extend *IMM* to other diffusion models, we first generalize the definition of RR sets in a model-independent manner as follows.

Definition 3.4 (Reverse Influence Set) *Let v be a node in V . A reverse influence (RI) set for v is a node set $R \subseteq V$, such that for any node set $S \subseteq V$, the probability that $R \cap S \neq \emptyset$ equals the probability that S as a seed set can activate v in a diffusion process. A random RI set is an RI set for a node selected uniformly at random from V .*

■

Notice that RR sets are a special case of RI sets under the triggering model, due to Lemma 3.1.

Let $\mathcal{R}$ be a set of RI sets under a certain diffusion model $\mathcal{M}$. Then, by Definition 3.4 and the linearity of expectation, we have

$$\mathbb{E}[I(S)] = n \cdot \mathbb{E}[F_{\mathcal{R}}(S)], \quad (\text{Eq. 3.24})$$

where $I(S)$ denotes the influence of S under $\mathcal{M}$, and $F_{\mathcal{R}}(S)$ denotes the fraction of RI sets in $\mathcal{R}$ that overlaps S . Given Equation Eq. 3.24 and the concentration results in Corollaries 3.1 and 3.2, it can be verified that all theoretical analysis in Section 3.3.2

holds under $\mathcal{M}$, if the input to Algorithm 1 is a set $\mathcal{R}$ of random RI sets generated under $\mathcal{M}$. It then follows that all results in Section 3.3.3 are still valid, even if we modify Algorithm 2 so that it generates random RI sets under $\mathcal{M}$ instead of random RR sets. As a consequence, Algorithm 3 is guaranteed to return a $(1 - 1/e - \varepsilon)$ -approximation under $\mathcal{M}$, with at least $1 - 1/n^\ell$ probability.

On the other hand, the time complexity of *IMM* under $\mathcal{M}$ may differ from that stated in Theorem 3.4, since Theorem 3.4 assumes $n \cdot EPT \leq m \cdot OPT$ (see Lemma 3.10), which does not necessarily hold under $\mathcal{M}$. Nonetheless, if we re-define *EPT* as the expected cost to generate a random RI set under $\mathcal{M}$, then it follows from Theorem 3.3 that *IMM* has an expected time complexity of $O\left(\frac{EPT}{OPT} \cdot (k + \ell)(n + m) \log n / \varepsilon^2\right)$.

In summary, *IMM* can be extended to any diffusion model $\mathcal{M}$ for which random RI sets can be generated efficiently. In addition, it provides the following theoretical guarantees under $\mathcal{M}$.

Theorem 3.5 *Under a diffusion model where a random RI set takes $O(EPT)$ expected time to generate, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability, and runs in $O\left(\frac{EPT}{OPT} \cdot (k + \ell)(n + m) \log n / \varepsilon^2\right)$ expected time. ■*

Proof: The approximation guarantees of *IMM* under $\mathcal{M}$ follow from the discussions in Section 3.4.1. In the following, we focus on *IMM*'s time complexity. Similar to the case of the triggering models, the computation cost of *IMM* under $\mathcal{M}$ also mainly consists of two parts, namely, the cost of generating the set $\mathcal{R}$ of random RI sets, and the cost of invoking Algorithm 1.

Let $w(R)$ be a random variable that denote the cost of generating a specific RI set R under $\mathcal{M}$. Using a derivation similar to the proof of Equation Eq. 3.22 in Section 3.3.4, we can prove that

$$\mathbb{E}\left[\sum_{R \in \mathcal{R}} w(R)\right] = \mathbb{E}[|\mathcal{R}|] \cdot EPT$$

$$= O\left(\frac{EPT}{OPT}(k + \ell)(n + m) \log n / \varepsilon^2\right).$$

To put it differently, the expected total cost of generating $\mathcal{R}$ is $O\left(\frac{EPT}{OPT}(k + \ell)(n + m) \log n / \varepsilon^2\right)$.

It suffices a proof of the theorem if we can show that the expected total cost incurred by Algorithm 1 is also no more than $O\left(\frac{EPT}{OPT}(k + \ell)(n + m) \log n / \varepsilon^2\right)$.

Following from the analysis in the proof of Theorem 3.4, the expected total cost incurred by Algorithm 1 is $O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|])$. Note that $|R| \leq w(R)$ for any $R \in \mathcal{R}$. Therefore, we have

$$\begin{aligned} O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|]) &\leq O(\mathbb{E}[\sum_{R \in \mathcal{R}} w(R)]) \\ &= O\left(\frac{EPT}{OPT}(k + \ell)(n + m) \log n / \varepsilon^2\right), \end{aligned}$$

which completes the proof. ■

The above extension of *IMM* requires that random RI sets exist under the diffusion model $\mathcal{M}$, which is not always the case. For example, consider a diffusion model $\mathcal{M}$ under which the expected influence of a node set S is not a submodular function of S . Then, it can be verified that it is impossible to generate RI sets under $\mathcal{M}$ that satisfy the condition stated in Definition 3.4. The following lemma presents a sufficient condition for the existence of random RI sets under $\mathcal{M}$.

Lemma 3.11 *Let $\Pr(S, v)$ be the probability that a node set S can influence a node v in a diffusion process under $\mathcal{M}$. If, for every v , $\Pr(S, v)$ is a submodular and monotone function of S , then random RI sets exist under $\mathcal{M}$.* ■

3.4.2 Application to Continuous-Time Models

The *continuous-time independent cascade (CTIC) model* [25, 27, 13, 46] assumes that each directed edge e in the social network G is associated with a *length distribution* $\mathcal{L}(e)$. Given a seed set S and a threshold t , a diffusion process under the CTIC model first

samples a length $l(e)$ for each edge e , and then activates any node that can be reached from a node in S via a path with a length no more than t . The rationale behind this model is that $w(e)$ represents the time required to transmit information from the starting point of e to its ending point, and accordingly, the diffusion process from S with a threshold t captures the effect of information cascade within a certain time frame. Such a model is useful in practice since, in viral marketing, a marketer is often concerned about the effectiveness of a marketing campaign within a given time (e.g., one month from the start of the campaign) [23].

In what follows, we demonstrate that our generalization of *IMM* in Section 3.4.1 can be applied to handle the CTIC model. Towards that end, we first present an efficient algorithm to construct an RI set $R(v)$ for any given node v . Conceptually, the algorithm works in three steps. First, it decides the length of each edge e in G by taking a sample from $\mathcal{L}(e)$. Let g be the resulting weighted graph. Then, the algorithm invokes the Dijkstra's algorithm [21] on g to identify the shortest distance from each node to v . Finally, it returns the set $R(v)$ of all nodes whose distance to v is no more than t (i.e., the threshold parameter in the CTIC model).

For practical efficiency, we implement the above algorithm as follows. We invoke the Dijkstra's algorithm to traverse G with v as the starting point, following only the incoming edge of each node. Each time we encounter an edge e whose length has not been decided, we sample its length from $\mathcal{L}(e)$. By the properties of the Dijkstra's algorithm [21], it would maintain a priority queue Q during the traverse of G , and it removes a node u from Q only if the shortest path from u to v is determined. Accordingly, we monitor each node u' removed from Q by the algorithm, and we terminate the algorithm whenever the shortest path from u' to v is longer than t . After that, we return the set $R(v)$ of all nodes that have been removed from Q , except u' . We refer to this algorithm as *RI-Gen*.

It can be verified that the running time of *RI-Gen* is $O(n^* \log n^* + m^*)$, where n^* and m^* are the numbers of nodes and edges inspected by *RI-Gen* during the traversal of G . Although $n^* = n$ and $m^* = m$ in the worst case, we will show that when v is selected from G uniformly at random, the expected time complexity of *RI-Gen* can be much smaller than $O(n \log n + m)$. Before that, we first establish the correctness of *RI-Gen*.

Lemma 3.12 *Suppose that we feed a node v as input to *RI-Gen*, and obtain a node set $R(v)$ as the output. Then, for any node set S , the probability that S overlaps $R(v)$ equals the probability that S as a seed set can activate v in a diffusion process under the CTIC model.* ■

Proof: First, we show that if $R(v)$ is constructed by the conceptual algorithm in Section 3.4.2, then the lemma holds. Let p_1 be the probability that S overlaps $R(v)$, and p_2 be the probability that S as a seed set can activate v in the diffusion process under the CTIC model. Observe that the conceptual algorithm first constructs g and then identifies nodes within t distance to v . Hence, p_1 equals the probability that g contains a directed path that starts from a node in S , ends at v , and has a length no more than t . In addition, p_2 equals the probability that v is reachable from S within t distance. Thus, we have $p_1 = p_2$.

Second, we prove that our implementation of the conceptual algorithm does not alter its output. In our implementation, we only sample the length of an edge at the first time that we encounter it. For any edge inspected in the algorithm, the effect of sampling its length on the fly is the same as sampling its length at the very beginning of the algorithm. Therefore, the node set $R(v)$ returned by our implementation is identical to the output of the conceptual algorithm. Thus, the lemma is proved. ■

Based on Lemma 3.12, we have the following result related to *RI-Gen*'s running time.

Lemma 3.13 *Let OPT^* be the largest expected influence of any size-1 node set in G under the CTIC model. Suppose that we choose a node v from G uniformly at random, and feed v as input to $RI\text{-}Gen$. Let m^* be the number of edges inspected by $RI\text{-}Gen$. Then,*

$$n \cdot \mathbb{E}[m^*] \leq m \cdot OPT^*. \quad \square$$

Proof: Let R be the RI set generated from v by $RI\text{-}Gen$, and $w(R)$ be the number of edges in G that point to the nodes in R . $\mathbb{E}[m^*] \leq \mathbb{E}[w(R)]$ holds because every edge inspected by $RI\text{-}Gen$ must point to a node in R . Define a probability distribution $\mathcal{V}'$ over all nodes, such that each node has the probability mass that proportional to its in-degree in G . Let v' be a random sample from $\mathcal{V}'$, and p_R be the probability that an edge selected uniformly at random from G would point to a node in R . By the definition of $w(R)$ and p_R , we have

$$m \cdot \mathbb{E}[p_R] = \mathbb{E}[p_R \cdot m] = \mathbb{E}[w(R)].$$

Define a function $f(v', R)$ for each v' sampled from $\mathcal{V}'$, such that $f(v', R) = 1$ if $v' \in R$, and $f(v', R) = 0$ otherwise. Regarding v' as a random variable and R as a fixed set, we have

$$p_R = \sum_{v'} \Pr[v'] \cdot f(v', R).$$

Next, we fix v' and consider R as a random variable. Define $p_{v'}$ to be the probability that R overlaps $\{v'\}$. We have

$$p_{v'} = \sum_R \Pr[R] \cdot f(v', R).$$

By Lemma 3.12, we have $p_{v'}$ equals the probability that $\{v'\}$ as a seed set can activate v . Thus, we have

$$n \cdot \mathbb{E}[p_{v'}] = \mathbb{E}[I(\{v'\})].$$

Since OPT^* is the largest expected influence of any size-1 node set, $\mathbb{E}[I(\{v'\})] \leq OPT^*$ holds. Combining all equations above, we have

$$\begin{aligned}
 n \cdot \mathbb{E}[m^*] &\leq n \cdot \mathbb{E}[w(R)] \\
 &= n \cdot m \cdot \mathbb{E}[p_R] \\
 &= n \cdot m \cdot \sum_R \Pr[R] \cdot p_R \\
 &= n \cdot m \cdot \sum_R (\Pr[R] \cdot \sum_{v'} \Pr[v'] \cdot f(v', R)) \\
 &= n \cdot m \cdot \sum_{v'} (\Pr[v'] \cdot \sum_R \Pr[R] \cdot f(v', R)) \\
 &= n \cdot m \cdot \sum_{v'} \Pr[v'] \cdot p_{v'} \\
 &= n \cdot m \cdot \mathbb{E}[p_{v'}] \\
 &= m \cdot \mathbb{E}[I(\{v'\})] \\
 &\leq m \cdot OPT^*.
 \end{aligned}$$

Thus, the lemma is proved. ■

Given Lemma 3.13, it can be derived that the expected time complexity of *RI-Gen* is $O\left(\frac{n \log n}{m} OPT\right)$. Meanwhile, Lemma 3.12 shows that we can extend *IMM* to tackle the CTIC model, by adopting *RI-Gen* to generate random RI sets. Then, by Theorem 3.5, we have the following result.

Theorem 3.6 *Under the CTIC model, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability, and runs in $O\left((k + \ell)(n + m) \log^2 n / \varepsilon^2\right)$ expected time.* ■

Proof: Let n^* and m^* be the number of nodes and edges inspected by *RI-Gen* respectively. Recall that *RI-Gen* runs in $O(n^* \log n^* + m^*)$ time. We have $n^* \log n^* + m^* = O(m^* \log n)$. By Lemma 3.13, we have

$$\mathbb{E}[m^* \log n] \leq \frac{m}{n} \cdot OPT^* \log n.$$

Observe that OPT^* is no more than the largest expected influence of a size- k node set, i.e., $OPT^* \leq OPT$ for any $k \geq 1$. Then, by Theorem 3.5, Algorithm 3 returns a $(1 - 1/e - \varepsilon)$ -approximation with at least $1 - 1/n^\ell$ probability under the CTIC model, and runs in $O((k + \ell)(n + m) \log^2 n / \varepsilon^2)$ expected time. Thus, the theorem is proved. ■

Comparison with *ConTinEst* [23]. *ConTinEst* is the state-of-the-art solution for influence maximization under the CTIC model. It first generates c weighted versions of G , denoted as $g_1, g_2, \dots, g_c$, such that in each g_i ($i \in [1, c]$), the length of each edge e is sampled from $\mathcal{L}(e)$ independently. After that, it constructs a *sketch* [18] on each g_i , and utilizes the sketches for influence maximization. Du et al. [23] provide an analysis of *ConTinEst*'s accuracy guarantee, but do not formally discuss its time complexity. In the following, we analyze the time complexity of *ConTinEst* based on the theoretical results from [23], and compare it with *IMM*.

First, the sketch on each g_i ($i \in [1, c]$) is parameterized with a positive integer d , and it takes $O(dn \log^2 n + dm \log n)$ time to construct [18]. Second, by Theorems 1 and 2 in [23], *ConTinEst* returns a $(1 - 1/e - \varepsilon)$ -approximate solution with at least $1 - 1/n^\ell$ probability, if

$$c \geq \frac{\Lambda \cdot 4k^3(\ell + 1)(\log n + \log 2)}{\varepsilon^2 \cdot OPT^2}, \text{ where} \quad (\text{Eq. 3.25})$$

$$\Lambda = \max_{S \subseteq V, |S| \leq k} \left(\frac{2}{d-2} \mathbb{E}[I(S)]^2 + \frac{d-1}{d-2} \text{Var}[I(S)] + \frac{4\varepsilon \cdot r(S)}{3k} OPT \right),$$

and $r(S)$ denotes the maximum number of nodes that S can activate under the CTIC model. We observe that Λ is unknown in advance, and hence, it is unclear how a user can select an appropriate c to ensure a $(1 - 1/e - \varepsilon)$ -approximation. However, this issue is not addressed in [23]. For our discussion, we use a loose lower bound of Λ to derive an optimistic estimation of *ConTinEst*'s time complexity, and then compare it with that of *IMM*. (Note that this is to the advantage of *ConTinEst*.)

Let S_k° be the size- k node set with the largest expected influence under the CTIC model. Then, $E[I(S_k^\circ)] = OPT$ and $r(S_k^\circ) \geq OPT$. Therefore,

$$\begin{aligned} \Lambda &\geq \left(\frac{2}{d-2} \mathbb{E}[I(S_k^\circ)]^2 + \frac{4\varepsilon \cdot r(S_k^\circ)}{3k} OPT \right) \\ &\geq \left(\frac{2}{d-2} + \frac{4\varepsilon}{3k} \right) OPT^2 \end{aligned}$$

By setting $\Lambda = \left(\frac{2}{d-2} + \frac{4\varepsilon}{3k} \right) OPT^2$ in Equation Eq. 3.25, we have

$$c \geq \left(\frac{8}{d-2} + \frac{16\varepsilon}{3k} \right) \cdot k^3(\ell + 1)(\log n + \log 2)/\varepsilon^2. \quad (\text{Eq. 3.26})$$

Given that *ConTinEst* needs to construct a sketch on each g_i ($i \in [1, c]$), and that each sketch takes $O(dn \log^2 n + dm \log n)$ time to generate, the time complexity of *ConTinEst* is at least

$$O \left(k^3(\ell + 1)(n \log n + m) \log^2 n / \varepsilon^2 \right).$$

Observe that this time complexity is larger than that of *IMM* (see Theorem 3.6) by a factor of more than k^2 . In Section 5.1, we will demonstrate that the empirical efficiency of *ConTinEst* is also inferior to that of *IMM*.

Chapter 4

Greedy Seed Minimization

4.1 Introduction

The seed minimization is an alternate influence maximization problem. Given a social network G and a covering threshold t , seed minimization problem are aiming to find a seed set S that has an expected influence not less than $t \cdot n$ and the minimize the size of S . This problem is firstly described in Goyal et al.'s work as *Minimum Target Set Selection (MINTSS)* [28, 29]. They present a $(1 + \ln(t/\varepsilon))$ -approximate solution by employing a greedy algorithm, where ε is a predefined parameter. But their method does not guarantee that the yielded solution has an expected influence spread larger than $t \cdot n$. Zhang et al. present a guaranteed solution at the sacrifice of a larger time complexity [64].

Our Contribution. This study presents *GSM*, a greedy algorithm solving seed minimization problem on social networks. It retains the succinctness of implementation in Goyal et al.s work, and achieves an $O(\log(\frac{t \cdot n}{\beta}))$ -approximate solution, where β is a predefined parameter. Meanwhile, *GSM* overcomes the drawback in Goyal et al.'s algorithm that its theoretical accuracy relies on the size of the unknown ideal seed set S_t^o . By applying an absolute oracle from *IMM*, *GSM* bridges the theory and practice in seed minimization problem.

4.2 Preliminaries

In this section, we present an overview of Goyal et al.’s solution [28, 29] and Zhang et al.’s work [64]. For ease of exposition, we also focus on the *independent cascade (IC)* model [39] like the influence maximization problem.

4.2.1 Goyal et al.’s Approach

Goyal et al. use simple greedy algorithm [28, 29] to find the minimum seed set that has at least $(t - \beta) \cdot n$ influence spread, where β is a tunable parameter. Since the seed minimization problem can be reduced to real-valued submodular set cover problem, theoretically the greedy approach can return an $O(1 + \log(t/\beta))$ -approximate solution. However, the influence spread cannot be exactly measured, another error control parameter ε_0 is introduced. Goyal et al. prove that for any $\phi > 0$, there exists a $\varepsilon_0 \in (0, 1)$ such that by using a $(1 - \varepsilon_0)$ -approximate estimation of influence $I(S)$, the greedy algorithm can return a $(1 + \phi)(1 + \log(t/\beta))$ -approximate solution.

Although Goyal et al.’s greedy algorithm is effective and succinct, it requires an unknown value in its estimation of influence. A compromise is to make 10000 Monte Carlo samples to evaluate the influence, similar to Kempe et al.’s experiments [39]. We modify this drawback and compare it with our *GSM* in section 4.4.

4.2.2 Zhang et al.’s Approach

Note that in Goyal et al.’s solution [28, 29], the greedy algorithm terminates when the condition $I'(S) > (t - \beta) \cdot n$ satisfies, where $I'(S)$ is the estimated influence. In other word, the solution is not guaranteed to have a influence larger than $t \cdot n$. The reason behind it is that the margin gain of adding a new node into S is so small it can hardly cover the error of influence estimation. Zhang et al. utilize the fact that adding one new node can increase at least one expected influence to assure its proposed solution has $t \cdot n$ influence spread. In contrast, their algorithm has a higher time complexity

4.2.3 Problem Definition

Let $G(V, E)$ be a social network with a node set V of n nodes and an edge set E of m directed edges. For any two nodes $u, v \in V$, if $\langle u, v \rangle$ is an edge in E , we refer to v as an outgoing neighbor of u , and u an incoming neighbor of v . We consider a time-stamped diffusion process on G as follows:

- (i) At the first timestamp, we *activate* a set S of nodes, and set the remaining nodes *inactive*.
- (ii) When a node u is first activated at timestamp i , it samples a set of its outgoing inactive neighbors according to a certain probability distribution, and activates them at timestamp $i + 1$. The node u can no longer activate any other nodes after that.
- (iii) All activated nodes cannot be deactivated.

Let $I(S)$ be the number of activated nodes in G when the above process converges, i.e., when no more nodes can be activated. We refer to S as the *seed set*, and $I(S)$ as the *influence* of S . Note that $I(S)$ is a random variable that depends on the probability distributions from which each node samples out-neighbors to activate. Intuitively, the influence propagation process under the IC model mimics the spread of an infectious disease: the seed set S is conceptually similar to an initial set of infected individuals, while the activation of a node by its neighbors is analogous to the transmission of the disease from one individual to another. We refer to the collection of those probability distributions as the *diffusion model* that underlies the diffusion process.

Problem Statement. Given G , a diffusion model M , a tunable parameter $t \in (0, 1)$, the seed minimization problem is aimed to find the minimum seed set S_t with expected influence satisfying $\mathbb{E}[I(S_t)] \geq t \cdot n$.

4.3 Proposed Solution

This section presents *GSM*, a greedy seed minimization approach based two-phase manner similar to *IMM*. And we compare it with Goyal et al.'s solutions [28, 29] in next section.

- (i) **Sampling.** This phase iteratively generates in total θ random RR sets which together constitute a set $\mathcal{R}$.
- (ii) **Node Selection.** This phase estimates maximum coverage in a greedy manner until meeting a certain stopping condition. The returned value S_t^* of this phase is regarded as the final result.

The sampling phase of *GSM* is similar to that of *IMM*. However, *GSM* does not need to estimate OPT as in *IMM* since t is a known parameter. Meanwhile, the node selection phase of *GSM* is close to the manner of Goyal et al.'s solutions [28] [29], we employ the greedy algorithm until the selected seeds have a larger estimated influence than $(t - \beta) \cdot n$, where β is a tunable parameter.

We use the list of notations similar to *IMM* as shown in Table 4.1 as that we frequently use. All logarithms are to the base e , unless specified otherwise.

4.3.1 Sampling Phase

Let $\mathcal{R} = \{R_1, R_2, \dots, R_\theta\}$ be the set of random RR sets generated by the sampling phase of *GSM*, $F_{\mathcal{R}}(S)$ be the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S and the seed set S_t^o as the minimum seed set that the equation $I(S_t^o) \geq t \cdot n$ satisfies. In the sampling phase, *GSM* generates a set $\mathcal{R}$ of random RR sets that is large enough for greedy algorithm in the node selection phase. In the sampling phase in *IMM*, the number of RR sets (i.e., θ) involves sophisticated designs and calculations, since OPT is an unknown parameter. However *GSM* can directly use t as the lower bound of $I(S_t^o)$. It leads to an easy calculation of θ as shown in Theorem 4.7.

Notation	Description
$G = (V, E)$	a social network G with a node set V and an edge set E
n, m	the numbers of nodes and edges in G , respectively
$\mathcal{G}$	the distribution of social networks induced by the triggering distribution of each node (see Section 3.2.1)
$I(S)$	the influence of a node set S in a diffusion process on G
$\mathbb{E}(\cdot)$	the expectation of a random variable
$\mathcal{R}$	the set of RR sets generated by <i>IMM</i> 's sampling phase
θ	the number of RR sets in $\mathcal{R}$
$F_{\mathcal{R}}(S)$	the fraction of RR sets in $\mathcal{R}$ that are covered by a node set S
t	the predefined influence
θ	the number of RR sets

 Table 4.1: Frequently used notations for *GSM*

Theorem 4.7 *Let $\varepsilon_0 > 0$, and*

$$\theta \geq (2 + \varepsilon_0) \frac{\log 2 + \log S(t) + \log(1/\delta)}{\varepsilon_0^2 t^2}. \quad (\text{Eq. 4.1})$$

If Equation Eq. 4.1 holds, $|n \cdot F_{\mathcal{R}}(S) - \mathbb{E}[I(S)]| \leq \varepsilon_0 \cdot tn$ with at least $\delta/S(t)$ probability, where $S(t) = \sum_{i=1}^{t \cdot n} \binom{n}{i}$.

Proof: Let $\rho = \mathbb{E}[F_{\mathcal{R}}(S)] = \mathbb{E}[I(S)]/n, \zeta = \frac{\varepsilon_0 t}{\rho}$.

$$\begin{aligned}
 & \Pr[|n \cdot F_{\mathcal{R}}(S) - \mathbb{E}[I(S)]| \geq \varepsilon_0 tn] \\
 &= \Pr[|\theta \cdot F_{\mathcal{R}}(S) - \theta \rho| \geq \frac{\varepsilon_0 t}{\rho} \theta \rho] \\
 &\leq 2 \exp\left(-\frac{\zeta^2}{2 + \zeta} \rho \theta\right) \\
 &\leq 2 \exp\left(-\frac{\varepsilon_0^2 t^2}{2\rho + \varepsilon_0 t} \theta\right) \\
 &\leq 2 \exp\left(-\frac{\varepsilon_0^2 t^2}{2 + \varepsilon_0 t} \theta\right) \\
 &= \frac{\delta}{S(t)}.
 \end{aligned}$$

Thus, the theorem is proved. ■

Algorithm 4 NodeSelection ($\mathcal{R}, t, \varepsilon_0, \ell, \beta$)

```

1: Initialize a set  $S_t^* = \emptyset$ 
2: while  $n \cdot F_{\mathcal{R}}(S_t^*) < (t - \beta) \cdot n$  do
3:   Identify the vertex  $v_i$  that minimize  $\frac{w_i}{F_{\mathcal{R}}(S_t^* \cup v_i) - F_{\mathcal{R}}(S_t^*)}$ 
4:   Insert  $v_i$  into  $S_t^*$ 
5: end while
6: return  $S_t^*$ 

```

4.3.2 Node Selection Phase

Algorithm 4 presents our node selection phase. After θ is calculated, the algorithm also follows the standard greedy approach for maximum coverage problem [59]. Note that the stopping condition in Line 2 is $n \cdot F_{\mathcal{R}}(S_t^*) < (t - \beta) \cdot n$. It is due to the complexity of theoretical analysis in terms of accuracy and efficiency if the algorithm terminates when $n \cdot F_{\mathcal{R}}(S_t^*) < (t \cdot n)$ is met. Goyal et al. [28] [29] follow a similar pattern to set a tunable parameter.

To prove the accuracy of Algorithm 4, we first introduce Lemma 4.14 to demonstrate the margin gain of adding a certain node.

Lemma 4.14 *Let S_t^o be the subset of V with minimum size such that $\mathbb{E}[I(S_t^o)] \geq t \cdot n$. For any $S \in V$ with $\mathbb{E}[I(S)] \leq t \cdot n$, there exists a node $v \in V \setminus S$ satisfying $\mathbb{E}[I(S \cup \{v\})] - \mathbb{E}[I(S)] \geq \frac{t \cdot n - \mathbb{E}[I(S)]}{|S_t^o|}$*

Proof: Assume $\forall v \in V \setminus S$, and $\mathbb{E}[I(S \cup \{v\})] - \mathbb{E}[I(S)] < \frac{t \cdot n - \mathbb{E}[I(S)]}{|S_t^o|}$. Let $S' = S_t^o \setminus S$.

We have

$$\begin{aligned}
\mathbb{E}[I(S_t^o \cup S)] &\leq \mathbb{E}[I(S)] + \sum_{v \in S'} (\mathbb{E}[I(S \cup v)] - \mathbb{E}[I(S)]) \\
&< \mathbb{E}[I(S)] + |S_t^o| \cdot \frac{t \cdot n - \mathbb{E}[I(S)]}{|S_t^o|} \\
&= t \cdot n.
\end{aligned}$$

It contradicts the fact that $\mathbb{E}[I(S_t^o)] \geq t \cdot n$. ■

In *IMM*, we estimate the influence $I(S)$ based on $F_{\mathcal{R}}(S)$. For ease, We name this kind of influence estimator as an *oracle*. Theorem 4.8 presents the theoretical upper bound of S_t^* given an oracle.

Theorem 4.8 *Let tunable parameters $\beta > 0$, $\phi > 0$. If $t \cdot n \geq 8$, our oracle returns the estimation of influence within the $\varepsilon_0 \cdot tn$ absolute error with the probability as least $\delta/S(t)$, we have $|S_t^*| \leq \alpha |S_t^o|$ with at least δ probability, where $\alpha = \left\lceil \log \left(\frac{tn}{\beta} \right) (1 + \phi) \right\rceil$, $\varepsilon_0 \leq \frac{\beta(tn)^{\phi'} - 1}{(tn)^{1+\phi'}}$ and $\phi' = \frac{\phi \cdot n - (\beta - 1)}{n + (\beta - 1) + \log(tn)}$.*

Proof: Since for each any seed set S , we have $|n \cdot F_{\mathcal{R}}(S) - \mathbb{E}[I(S)]| \leq \varepsilon_0 n$ with at least $\delta/S(t)$ probability, which can be written in

$$\mathbb{E}[I(S)] - \varepsilon_0 tn \leq n \cdot F_{\mathcal{R}}(S) \leq \mathbb{E}[I(S)] + \varepsilon_0 tn,$$

and

$$n \cdot F_{\mathcal{R}}(S) - \varepsilon_0 tn \leq \mathbb{E}[I(S)] \leq n F_{\mathcal{R}}(S) + \varepsilon_0 tn.$$

First, we consider a set S_l in l -th iteration, satisfying $n \cdot F_{\mathcal{R}}(S_l) \geq (1 + \varepsilon_0)(tn - \beta)$ and $n \cdot F_{\mathcal{R}}(S_{l-1}) < (tn - \beta)$. We compute the difference between $n \cdot F_{\mathcal{R}}(S_i)$ and $n \cdot F_{\mathcal{R}}(S_{i-1})$ for all $1 \leq i \leq l - 1$. By $S_0 = \emptyset$,

$$\begin{aligned} \mathbb{E}[I(S_{i-1})] &\leq n \cdot F_{\mathcal{R}}(S_{i-1}) + \varepsilon_0 tn \\ &\leq (1 + \varepsilon_0)(tn - \beta) + \varepsilon_0 tn \\ &= tn + (-(1 + \varepsilon_0)\beta + 2\varepsilon_0 tn) \\ &\leq tn + \left(-\frac{1}{2}(1 - (1 + \varepsilon_0)t - \varepsilon_0 t) + 2\varepsilon_0 tn\right) \\ &\leq tn + 2(-(1 + t) + \varepsilon_0(t + 4tn)) \\ &\leq tn \end{aligned}$$

Let $t_i = t - \mathbb{E}[I(S_i)]/n$ and $t_i^* = t - F_{\mathcal{R}}(S_i)$. By Lemma 4.14, there exists a node $x \in V \setminus S_{i-1}$ satisfying $\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})] \geq \frac{t_{i-1} \cdot n}{|S_{i-1}^o|}$. Let $\zeta = 1/(1 + \varepsilon_1)$. When

$\theta' \geq 2n(\log S(t) + \delta)/\varepsilon_1^2$ holds,

$$\begin{aligned}
 & \Pr[n \cdot F_{\mathcal{R}}(S_{i-1} \cup \{x\}) - n \cdot F_{\mathcal{R}}(S_{i-1}) \leq \zeta(\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])] \\
 = & \Pr[\theta'(F_{\mathcal{R}}(S_{i-1} \cup \{x\}) - F_{\mathcal{R}}(S_{i-1})) - \theta'(\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])/n \\
 \leq & -(1 - \zeta)\theta'(\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])/n] \\
 \leq & \exp\left(-\frac{(1 - \zeta)^2}{2n}\theta'(\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])\right) \\
 \leq & \exp\left(-\frac{(1 - \zeta)^2}{2n}\theta'\right) \\
 \leq & \delta/S(t).
 \end{aligned}$$

Thus, we have $n \cdot F_{\mathcal{R}}(S_{i-1} \cup \{x\}) - n \cdot F_{\mathcal{R}}(S_{i-1}) \geq 1/(1 + \varepsilon_1)(\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])$

with at least $\delta/S(t)$ probability. It leads to

$$\begin{aligned}
 & n \cdot F_{\mathcal{R}}(S_i) - n \cdot F_{\mathcal{R}}(S_{i-1}) \\
 \geq & n \cdot F_{\mathcal{R}}(S_{i-1} \cup \{x\}) - n \cdot F_{\mathcal{R}}(S_{i-1}) \\
 \geq & (\mathbb{E}[I(S_{i-1} \cup \{x\})] - \mathbb{E}[I(S_{i-1})])/(1 + \varepsilon_1) \\
 \geq & \frac{tn - \mathbb{E}[I(S_{i-1})]}{(1 + \varepsilon_1)|S_t^o|} \\
 \geq & \frac{tn - n \cdot F_{\mathcal{R}}[I(S_{i-1})] - \varepsilon_0 tn}{(1 + \varepsilon_1)|S_t^o|} \\
 \geq & \frac{t_{i-1}^*}{(1 + \varepsilon_1)|S_t^o|} - \frac{\varepsilon_0 tn}{(1 + \varepsilon_1)|S_t^o|}.
 \end{aligned}$$

Then we have

$$t_{i-1}^* - t_i^* = F_{\mathcal{R}}(S_i) - F_{\mathcal{R}}(S_{i-1}) \geq \frac{t_{i-1}^*}{(1 + \varepsilon_1)|S_t^*|} - \frac{\varepsilon_0 t}{(1 + \varepsilon_1)|S_t^o|},$$

which leads to

$$t_i^* \leq (1 - \frac{1}{(1 + \varepsilon_1)|S_t^*|})t_{i-1}^* + \frac{\varepsilon_0 t}{(1 + \varepsilon_1)|S_t^o|}.$$

Since $n \cdot F_{\mathcal{R}}(S_{l-1}) < (tn - \beta)$, $t_{l-1}^* \geq \frac{\beta}{n}$ holds. Let $a = 1 - \frac{1}{(1+\varepsilon_1)|S_t^*|}$ and $b = \frac{\varepsilon_0 t}{(1+\varepsilon_1)|S_t^o|}$.

We have

$$\begin{aligned} t_{l-1}^* &\leq a \cdot t_{l-2}^* + b \\ &\leq a^{l-1}t + \frac{1 - a^{l-1}}{1 - a}b \\ &< a^{l-1}t + \frac{1}{1 - a}b. \end{aligned}$$

Thus, we have

$$\frac{\beta}{n} < a^{l-1}t + \frac{1}{1 - a}b.$$

With the fact that $1 + x \leq e^x$ holds $\forall x \geq 0$, we have

$$\begin{aligned} \frac{\beta}{n} &\leq \exp\left(-\frac{l-1}{(1+\varepsilon_1)|S_t^*|}\right)t + \varepsilon_0 t \\ l &< (1+\varepsilon_1)|S_t^o| \log\left(\frac{t}{\beta/n - t\varepsilon_0}\right) + 1 \\ l &\leq \left\lceil (1+\varepsilon_1)|S_t^o| \log\left(\frac{t}{\beta/n - t\varepsilon_0}\right) \right\rceil \\ |S_l| &\leq \left\lceil \log\left(\frac{tn}{\beta - t\varepsilon_0 n}\right) \right\rceil (1+\varepsilon_1)|S_t^o|. \end{aligned}$$

Since $\varepsilon_0 \leq \frac{\beta(tn)^{\phi'} - 1}{(tn)^{1+\phi'}}$, we have $|S_l| \leq \lceil \log(tn) \rceil (1+\phi')(1+\varepsilon_1)|S_t^o|$. When we set $\varepsilon_1 = t\varepsilon_0$, we have $\theta = \theta_1 > 2n(\log S(t) + \ell \log n)/\varepsilon_1$. Finally we have a $\left\lceil \log(tn) (1+\phi')(1 + \frac{\beta(tn)^{\phi'} - 1}{(tn)^{\phi'+1}}) \right\rceil$ -approximate solution.

Since we need at most $t \cdot n$ seeds to have $t \cdot n$ nodes influenced, by the union bound of all sets with size from 1 to $t \cdot n$, $|S_t^*| \leq \alpha |S_t^o| + 1$ holds at least δ probability.

To get a more succinct expression, let $(1 + \phi')(1 + t \cdot \frac{\beta(tn)^{\phi'} - 1}{(tn)^{\phi'+1}}) \leq (1 + \phi)$. Since this expression is hard to compute, we utilize Taylor expansion to get an approximate solution. Let $x = 1 + \phi'$, $f(x) = x(1 + t \cdot \frac{\beta(tn)^{x-1} - 1}{(tn)^x})$ and $a = 1$. Expand $f(x)$ at $x = a$, which leads to

$$f(x) = \frac{f(a)}{0!} + \frac{f'(a)}{1!}(x - a) + R_1(x).$$

And $R_1(x)$ is the first order remainder. We have

$$\begin{aligned} f'(x) &= 1 + t \cdot \frac{\beta(tn)^{x-1} - 1}{(tn)^x} + t \cdot x \left(\frac{\beta(tn)^{x-1} \log(tn) \cdot (tn)^x - (\beta(tn)^{x-1} - 1) \cdot (tn)^x \log(tn)}{tn^{2x}} \right) \\ &= 1 + t \cdot \frac{\beta(tn)^{x-1} - 1}{(tn)^x} + t \cdot x \cdot \frac{\log(tn)}{(tn)^x}, \end{aligned}$$

And

$$\begin{aligned} f''(x) &= t \cdot \frac{\log(tn)}{(tn)^x} + t \cdot \frac{\log(tn)}{(tn)^x} - t \cdot x \cdot \frac{\log^2(tn)}{(tn)^x} \\ &= t \cdot \frac{\log(tn)}{(tn)^x} (2 - x \cdot \log(tn)). \end{aligned}$$

$$\begin{aligned} f(x) &= \frac{f(a)}{0!} + \frac{f'(a)}{1!} (x - a) + R_1(x) \\ &= \left(1 + \frac{\beta - 1}{n}\right) + \left(\left(1 + \frac{\beta - 1}{n}\right) + \frac{\log(tn)}{n} \right) (x - 1) + R_1(x) \end{aligned}$$

By using Lagrange remainder, we could have $R_1(x) = \frac{f''(\xi)}{2!} (x - a)^2$, where ξ is a number between a and x . With the assumption that $tn \geq 8$, we have $2 - \xi \cdot \log(tn) \leq 0$, which leads to $R_1(x) \leq 0$.

Let $(1 + \frac{\beta-1}{n}) + ((1 + \frac{\beta-1}{n}) + \frac{\log(tn)}{n})(x-1) = (1 + \phi)$, we get $\phi' = x - 1 = \frac{\phi - \frac{\beta-1}{n}}{(1 + \frac{\beta-1}{n}) + \frac{\log(tn)}{n}} = \frac{\phi \cdot n - (\beta-1)}{n + (\beta-1) + \log(tn)}$. With this setting, we will have $(1 + \phi')(1 + \frac{\beta(tn)^{\phi'} - 1}{(tn)^{\phi'+1}}) = (1 + \phi) + R_1(1 + \phi') \leq (1 + \phi)$.

Thus, the theorem is proved. ■

4.4 Comparison

In this section, we present the theoretical analysis of Goyal et al.'s method given the oracle in their paper. And we modify their oracle to present a more practical solution.

Let $f(S)$ be a function we are aiming to estimate and $f'(S)$ be the estimation result of a certain oracle. We denote it as an absolute oracle if it returns a solution in the form of $(1 - \varepsilon_1)f(S) \leq f'(S) \leq (1 + \varepsilon_2)f(S)$, where ε_1 and ε_2 are non-negative parameters known to us before estimation. Otherwise, the oracle is a relative oracle. In other words, a relative oracle involves at least one undetermined value, like OPT in the influence maximization problem. In Goyal et al.'s paper, the oracle they use does not return a solution with absolute error ε_0 . Instead a relative error oracle is employed in their algorithm, which is unpractical because we do not know the value of $|S_t^o|$.

In experiment, Goyal et al. estimate influence by R times via Monte Carlo simulations. And R is set to 10000 according to Kempe et al.'s work [39]. For ease of writing, we denote κ as $|S_t^o|$. By following the proof of Theorem 2 in their paper, we build following lemma for a relative oracle:

Lemma 4.15 *Given ϕ and β , Goyal et al.'s method reaches a $(1 + \phi)(1 + \log(tn/\beta))$ -approximate solution, if the oracle could give the estimation of influence satisfying $(1 - \varepsilon_0)f(S_i) \leq f'(S_i) \leq f(S_i)$ and $\varepsilon_0 \leq \frac{(tn/\beta)^\phi - 1}{\kappa((tn/\beta)^{1+\phi} - 1)}$.*

Proof: Let $x = tn/\beta$ and $X = \frac{x^\phi - 1}{x^{1+\phi} - 1}$, we have

$$\begin{aligned}
 \varepsilon_0 &\leq \frac{X}{\kappa} \\
 \varepsilon_0 &\leq \frac{X}{\kappa(1 + X/\kappa - X)} \\
 \varepsilon_0 &\leq X/\kappa + X\varepsilon_0 - X\varepsilon_0/\kappa \\
 \varepsilon_0 &\leq X - X(1 - \varepsilon_0)(1 - 1/\kappa) \\
 \frac{\varepsilon_0}{(1 - (1 - \varepsilon_0)(1 - 1/\kappa))} &\leq X.
 \end{aligned}$$

As proven in Goyal et al.'s paper, their algorithm returns a $(1 + \phi)(1 + \log(tn/\beta))$ -approximate solution if $\frac{\varepsilon_0}{(1-(1-\varepsilon_0)(1-1/\kappa))} \leq \frac{x^\phi - 1}{x^{1+\phi} - 1}$. $\blacksquare$

Note that in the proof of Theorem 2 of Goyal et al.'s paper [28], their method requires the estimation of the influence $f'(S_i)$ satisfying $(1 - \varepsilon_0)f(S_i) \leq f'(S_i) \leq f(S_i)$ to reach a $(1 + \phi)(1 + \log(tn/\beta))$ -approximate solution. This requirement could not be satisfied unless we have unlimited number of Monte Carlo simulations. Therefore, the accuracy declared by Goyal et al. cannot be achieved in their experiments. A more practical way to implement is to apply an oracle satisfying $(1 - \varepsilon_0)f(S_i) \leq f'(S_i) \leq (1 + \varepsilon_0)f(S_i)$. Thus, we can obtain Lemma 4.16 by modify their oracle to an counterpart with absolute error.

Lemma 4.16 *Given ϕ and β , Goyal et al.'s method returns a $\frac{(1+\phi)}{1-2\varepsilon_0}(1 + \log(tn/\beta))$ -approximate solution, if the oracle gives the estimation of influence satisfying $(1-\varepsilon_0)f(S_i) \leq f'(S_i) \leq (1 + \varepsilon_0)f(S_i)$ and $\varepsilon_0 \leq \frac{(tn/\beta)^\phi - 1}{2\kappa((tn/\beta)^{1+\phi})}$.*

Proof: We have

$$(1 - \varepsilon_0)f(S) \leq f'(S) \leq (1 + \varepsilon_0)f(S)$$

holds for all node sets S in G . Without loss of generality, let x be the node providing maximum margin gain and y be the one selected in greedy algorithm. We have

$$(1 - \varepsilon)f(S_i \cup \{x\}) \leq f'(S_i \cup \{x\}) \leq f'(S_i \cup \{y\}).$$

Let $t_i^* = tn - f'(S_i)$. According to Lemma 1, we have

$$\begin{aligned} \frac{tn - f(S_i)}{\kappa} &\leq f(S_i \cup \{x\}) - f(S_i) \\ \frac{tn - f'(S_i)/(1 - \varepsilon_0)}{\kappa} &\leq \frac{f'(S_i \cup \{y\})}{1 - \varepsilon_0} - \frac{f'(S_i)}{1 + \varepsilon_0} \\ \frac{t_i^*}{\kappa} &\leq \frac{tn - t_{i+1}^*}{1 - \varepsilon_0} - (tn - t_i^*)\left(\frac{1}{1 + \varepsilon_0} + \frac{\varepsilon_0}{\kappa(1 - \varepsilon_0)}\right) \end{aligned}$$

$$t_{i+1} \leq t_i^* \left(\frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{1-2\varepsilon_0}{\kappa} \right) + tn \left(1 - \frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{\varepsilon_0}{\kappa} \right).$$

Let $a = \left(\frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{1-2\varepsilon_0}{\kappa} \right)$ and $b = tn \left(1 - \frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{\varepsilon_0}{\kappa} \right)$, we have

$$\begin{aligned} t_{i+1} &\leq at_i^* + b \\ &\leq a^{i+1}tn + \frac{1-a^{i+1}}{1-a}b \\ &\leq a^{i+1}tn + \frac{1}{1-a}b. \end{aligned}$$

Suppose the greedy algorithm takes l iterations. Let $\delta = \frac{b}{(1-a)tn}$. Then,

$$\begin{aligned} t_l &\leq a^l tn + \frac{1}{1-a}b \\ t_l &\leq \left(\frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{1-2\varepsilon_0}{\kappa} \right)^l tn + \frac{1}{\left(\frac{2\varepsilon_0}{1+\varepsilon_0} + \frac{1-2\varepsilon_0}{\kappa} \right)} tn \left(1 - \frac{1-\varepsilon_0}{1+\varepsilon_0} - \frac{\varepsilon_0}{\kappa} \right) \\ t_l &\leq \left(1 - \frac{1-2\varepsilon_0}{\kappa} \right)^l tn + tn\delta \\ t_l &\leq \exp\left(-\frac{(1-2\varepsilon_0)l}{\kappa}\right) + tn\delta \\ l &\leq \frac{k}{1-2\varepsilon_0} \log\left(\frac{tn}{t_l - tn\delta}\right) \\ l &\leq \frac{k}{1-2\varepsilon_0} \log\left(\frac{tn}{\beta - tn\delta}\right). \end{aligned}$$

Let $x = tn/\beta$. We want $x^{1+\phi} \geq x \frac{1}{(1-\delta x)}$, i.e., $\delta \leq \frac{x^\phi - 1}{x^{\phi+1}}$. Assume $k \geq 1 + \varepsilon_0$ holds. And let $X = \frac{x^\phi - 1}{x^{\phi+1}}$. If $\varepsilon_0 \leq \frac{X}{2k}$, we have

$$\begin{aligned} \varepsilon_0 &\leq \frac{X}{2\kappa} \\ \varepsilon_0 &\leq \frac{X}{2\kappa(1 - X + X/\kappa)} \\ 2\varepsilon_0 &\leq 2X\varepsilon_0 + \frac{X}{\kappa} - \frac{2X}{\kappa}\varepsilon_0 \end{aligned}$$

	Goyal et al.'s	GSM
Oracle type	relative	absolute
Oracle accuracy		
In lemma 4.15	$\varepsilon_0 \leq \frac{(tn/\beta)^\phi - 1}{\kappa((tn/\beta)^{1+\phi} - 1)}$	$\varepsilon_0 \leq \frac{\beta(tn)^{\phi'} - 1}{(tn)^{1+\phi'}}$ and $\phi' = \frac{\phi \cdot n - (\beta - 1)}{n + (\beta - 1) + \log(tn)}$
In lemma 4.16	$\varepsilon_0 \leq \frac{(tn/\beta)^\phi - 1}{2\kappa((tn/\beta)^{1+\phi} - 1)}$	
Approximation		
In experiment	$\lceil \log(tn/\beta) \rceil$	$\lceil (1 + \phi) \log(tn) \rceil$
In lemma 4.15	$\lceil (1 + \phi) \log(tn/\beta) \rceil$	
In lemma 4.16	$\left\lceil \frac{(1+\phi)}{1 - 2 \frac{(tn/\beta)^\phi - 1}{\kappa((tn/\beta)^{1+\phi} - 1)}} \log(tn/\beta) \right\rceil$	

Table 4.2: Comparison with Goyal et al's method

$$\begin{aligned}
 X &\geq \frac{1}{2\varepsilon_0 + \frac{1-2\varepsilon_0}{\kappa}} \left(\frac{2\varepsilon_0}{1} \right) \\
 X &\geq \frac{1}{\left(\frac{2\varepsilon_0}{1+\varepsilon_0} + \frac{(1-2\varepsilon_0)}{\kappa(1+\varepsilon_0)} \right)} \left(\frac{2\varepsilon_0}{1+\varepsilon_0} \right) \\
 X &\geq \frac{1}{\left(\frac{2\varepsilon_0}{1+\varepsilon_0} + \frac{1-2\varepsilon_0}{\kappa} \right)} \left(\frac{2\varepsilon_0}{1+\varepsilon_0} \right) \\
 X &\geq \frac{1}{\left(\frac{2\varepsilon_0}{1+\varepsilon_0} + \frac{1-2\varepsilon_0}{\kappa} \right)} \left(\frac{2\varepsilon_0}{1+\varepsilon_0} - \frac{\varepsilon_0}{\kappa} \right) \\
 X &\geq \delta.
 \end{aligned}$$

Thus, the lemma is proved. ■

In Table 4.4, we elaborate the requirements of oracles and approximations for both methods. We can see Goyal et al.'s oracle is not feasible in practical usage, since κ is unknown. In comparison, our *GSM* algorithm requires an absolute oracle independent from the value of κ . To compare fairly, we calculate the computing overheads of Goyal et al.'s method as shown in Lemma 4.15 and 4.16. Both of expressions are inferior to our $\lceil (1 + \phi) \log(tn) \rceil$ time complexity.

Chapter 5

Experiments

5.1 Influence Maximization via Martingale

5.1.1 Experimental Settings

All of our experiment are conducted on a 64-bit Ubuntu 14.04.5 system with an 8-core 2.4GHz Intel Xeon E5-2650 CPU and 64 GB memory.

Datasets. We use five real social networks in our experiments, as shown in Table 5.1. Among them, *Pokec*, *LiveJournal*, *Orkut*, and *Twitter* are the largest datasets ever used in the literature of influence maximization. All datasets are publicly available: *Pokec*, *LiveJournal*, and *Orkut* can be obtained from [3], while *NetHEPT* and *Twitter* can be obtained from [4] and [5], respectively.

Diffusion Models. We consider three well-adopted diffusion models in our experiments, namely, the *independent cascade (IC)* model [39], the *linear threshold (LT)* [39] and the *continuous time dependent cascade (CTIC)* model [46, 13, 25]. For the IC model, we set the propagation probability $p(e)$ of each edge e to $1/i$, where i denotes the number of edges that share the same ending point with e . This setting is adopted in previous work [32, 14, 38]. For the CTIC model, we set the length distribution of each edge e to be a *Weibull distribution* [42], with the following pdf:

$$f(x) = \frac{a}{b} \cdot \left(\frac{x}{b}\right)^{a-1} \cdot e^{-(x/b)^a},$$

Table 5.1: Datasets.

Name	n	m	Type	Corresponding Site
<i>NetHEPT</i>	15.2K	31.4K	undirected	arxiv.org
<i>Pokec</i>	1.6M	30.6M	directed	pokec.azet.sk
<i>LiveJournal</i>	4.8M	69.0M	directed	www.livejournal.com
<i>Orkut</i>	3.1M	117.2M	undirected	www.orkut.com
<i>Twitter</i>	41.7M	1.5G	directed	twitter.com

where a and b are sampled from $[0, 10]$ uniformly at random for each edge e . In addition, the threshold parameter t of the CTIC model is varied from 1 to 10 in our experiments. These settings follow from the experiments in [23] for evaluating *ConTinEst* [23].

Recall from Section 3.2.3 that the LT model is a special case of the triggering model where each sample from a node v 's triggering distribution $\mathcal{T}(v)$ is a unit set containing only one incoming neighbor of v . Following previous work [13], we construct $\mathcal{T}(v)$ as follows. First, for each of v 's incoming neighbors u , we assign to u a number selected from $[0, 1]$ uniformly at random. Then, we normalize the numbers assigned to all incoming neighbors, so that they sum up to one. Finally, we set the normalized number of each incoming neighbor as the probability that it is sampled from $\mathcal{T}(v)$.

Algorithms. We compare *IMM* with five algorithms, namely, *TIM* [58], *TIM*⁺ [58], *IRIE* [38] and *SIMPATH* [32]. In particular, *TIM* and *TIM*⁺ are the state-of-the-art techniques that return $(1 - 1/e - \varepsilon)$ -approximations under the triggering model. We implement *IMM* with C++, and we adopt the C++ implementations of the other algorithms made available by their inventors.

Parameter Settings. For *IMM*, *TIM*, *TIM*⁺, we set $\varepsilon = 0.5$, $\ell = 1$, and $k = 50$ by default. For *IRIE* and *SIMPATH*, we set their internal parameters as recommended in [38] and [32], respectively. In each of our experiments, we run each method 5 times and report the average measurements. In addition, to compare the quality of the seed sets

returned by each method, we estimate the expected influence of any seed set S by taking its average influence in 10000 simulations of the diffusion process.

5.1.2 Results under the IC Model

In our first set of experiments, we compare four methods that support the IC models, namely, *IMM*, *TIM*, *TIM*⁺, and *IRIE*. Figure 5.1a shows the computational costs of *IMM* and *TIM*⁺ on *Twitter* (i.e., our largest dataset), with k varying from 1 to 50. We omit *TIM* and *IRIE*, since the former incurs prohibitive cost on *Twitter*, while the latter requires more than 64G memory to process the data. Observe that *IMM* consistently outperforms *TIM*⁺ in terms of running time, and is faster than *TIM*⁺ by three orders of magnitude when $k = 1$. The reason that *TIM*⁺ is inferior to *IMM* is two-fold. First, as we point out in Section 3.3, the node selection phase of *TIM*⁺ requires generating a large number of RR sets, due to the facts that (i) it cannot reuse RR sets produced in the parameter estimation phase, and (ii) it makes sub-optimal choices when setting its internal parameters. Second, the parameter estimation phase of *TIM*⁺ (denoted as *TIM*⁺-*PE*) incurs significant overheads when k is small. The rationale is that, when k decreases, the optimal expected influence *OPT* also tends to decrease, in which case it is more challenging to derive a reasonable lower bound of *OPT*. To illustrate *TIM*⁺-*PE*'s computational overhead, we plot it in we plot it in Figure 5.1 along with *IMM* and *TIM*⁺. Observe that *TIM*⁺ runs even slower than *IMM*. Furthermore, when $k = 1$, *TIM*⁺-*PE* accounts for 97% of *TIM*'s computation time. In contrast, *IMM* does not suffer from this issue, since it adopts a more effective statistical test in its sampling phase.

Figure 5.1b illustrates the running time of *IMM* and *TIM*⁺ on *Twitter* as a function of ε . Regardless of the value of ε , the processing cost of *IMM* is lower than that of *TIM*⁺ by an order of magnitude. The computation time of both *IMM* and *TIM* decreases with ε , since a larger ε leads to a less stringent theoretical guarantee, which is easier to achieve.

Figure 5.1: Running time and expected influence on *Twitter* under the IC model ($\ell = 1$ for *IMM* and TIM^+).

Figure 5.2: Running time vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and TIM^+).

Figure 5.3: Expected influence vs. k under the IC model ($\varepsilon = 0.5$ and $\ell = 1$ for *IMM*, *TIM*, and TIM^+).

The overhead of TIM^+-PE is insensitive to ε , which is consistent with the fact that its time complexity is independent of ε . Figure 5.1c (resp. Figure 5.1d) shows the expected influence of *IMM* and TIM^+ when k (resp. ε) varies. In all cases, *IMM* and TIM^+ yield similar results.

Figure 5.2 plots the running time of *IMM*, *TIM*, TIM^+ , and *IRIE* against k for the four smaller datasets. Again, *IMM* consistently outperforms TIM^+ , especially when k is small.

Furthermore, the parameter estimation phase of TIM^+ (i.e., TIM^+-PE) alone incurs a higher cost than IMM in most cases. On the other hand, TIM^+ is more efficient than TIM , due to the heuristic optimization that it adopts to reduce computational overhead of its node selection phase (see Section 3.2.3). $IRIE$ outperforms IMM on *NetHEPT* when $k \leq 5$, but is inferior to IMM in all other cases. In particular, the running time of $IRIE$ is around two orders of magnitude larger than that of IMM on *Pokec*, *LiveJournal*, and *Orkut*, when $k \geq 20$.

Figure 5.3 shows the expected influence of each method on the four smaller datasets. IMM , TIM , and TIM^+ produce comparable results in all scenarios. Meanwhile, on *LiveJournal*, the quality of seed sets returned by $IRIE$ are noticeably worse than those returned by IMM , TIM , and TIM^+ . The reason is that IMM , TIM , and TIM^+ all provide asymptotic guarantees in terms of the accuracy of their solutions, whereas $IRIE$ is a heuristic method that does not offer any worst-case assurance.

Remark. We have also evaluated the effect of ε on the performance of IMM , TIM , and TIM^+ on the four smaller datasets. We vary ε from 0.1 to 0.5, and find that IMM consistently surpasses both TIM and TIM^+ in terms of efficiency, while achieving similar expected influence. We omit the results due to the space constraint.

5.1.3 Results under the LT model

Our second set of experiments compares four methods that support the LT model, i.e., IMM , TIM , TIM^+ , and $SIMPATH$. Figure 5.4 shows the running time and expected influence of IMM and TIM^+ on *Twitter*, omitting TIM and $SIMPATH$ since they entail prohibitive costs on the dataset. As with the case under the model, IMM considerably outperforms TIM^+ in all cases, and is even more efficient than the parameter estimation phase of TIM^+ (denoted as TIM^+-PE). At the same time, there is no noticeable difference between the quality of the results returned by IMM and TIM^+ .

Figure 5.4: Running time and expected influence on *Twitter* under the LT model ($l = 1$ for *IMM* and TIM^+).

Figure 5.5: Running time vs. k under the LT model ($\varepsilon = 0.5$ and $l = 1$ for *IMM*, *TIM*, and TIM^+).

Figure 5.5 (resp. Figure 5.6) illustrates the computational overheads (resp. expected influence) of all methods on the four smaller datasets. The relative performance of *IMM*, *TIM*, and TIM^+ is qualitatively similar to that in Figures 5.2 and 5.3, with *IMM* surpassing *TIM* and TIM^+ in terms of efficiency. On the other hand, *SIMPATh* incurs a much larger overheads than all other and its result accuracy on *Pokec* and *LiveJournal* is considerably lower than that of *IMM*, *TIM*, and TIM^+ .

5.1.4 Results under the CTIC Model

Our last set of experiments compares *IMM* with *ConTinEst* under the CTIC model. We first consider the case when both *IMM* and *ConTinEst* return $(1 - 1/e - \varepsilon)$ -approximate solutions with $\varepsilon \leq 0.5$. Towards this end, we tune *ConTinEst*'s parameters c and d as follows. First, we set $d = 5$, following the experiments in [23]. Then, we let c be the

Figure 5.6: Expected influence vs. k under the LT model ($\varepsilon = 0.5$ and $\ell = 1$ for IMM , TIM , and TIM^+).

Figure 5.7: Running time of ***IMM*** and ***ConTinEst*** under the CTIC model.

smallest value that satisfies Equation Eq. 3.25. Note that this setting is to the advantage of *ConTinEst* because, as we analyze in Section 3.4.2, *ConTinEst* may require a much larger c to ensure $(1 - 1/e - \varepsilon)$ -approximations.

Figure 5.7 a illustrates the computation time of *IMM* and *ConTinEst* on *NetHEPT*, when $\varepsilon = 0.5$, $t = 10$, and k varies from 1 to 5. (Recall that t is the threshold parameter in the CTIC model.) We use only *NetHEPT* in this experiment, since *ConTinEst* incurs prohibitive overheads on the larger datasets. Observe that the running time of *ConTinEst* increases significantly with k . The reason is that, when the approximation ratio of *ConTinEst* is fixed, its computational cost is roughly proportional to k^3 , as we analyze in Section 5.1.4. In contrast, *IMM* scales much better with k , and it is three orders of magnitude faster than *ConTinEst* when $k = 5$.

Figure 5.7 b shows the overheads of *IMM* and *ConTinEst* when $\varepsilon = 0.5$, $k = 3$, and t varies from 1 to 10. The running time of *IMM* increases with t . To explain, recall that

under the CTIC model, *IMM* generates each RI set by invoking the Dijkstra’s algorithm on a weighted version of G to identify nodes that are within t distance to a given node v . When t increases, there exists a larger number of nodes in G that are within t distance to v , and hence, the construction of RI sets incurs a larger overhead. This renders *IMM* less efficient. In contrast, the computational cost of *ConTinEst* is insensitive to t . That said, even when $t = 10$, *IMM* is still more than 250 times faster than *ConTinEst*. Finally, Figure 5.7c plots the processing costs of *IMM* and *ConTinEst* when $k = 3$, $t = 10$, and ε varies from 0.1 to 0.5. The costs of *IMM* and *ConTinEst* both decrease with the increase of ε , but *IMM* is significantly more efficient than *ConTinEst* in all cases.

5.2 Greedy Seed Minimization

5.2.1 Experimental Settings

We conduct several experiments to evaluate the time performance of *GSM* and Goyal et al.’ *MINTSS* algorithm [29, 28]. Both methods are compiled by g++ 4.84 with C++ 11 standard, and conducted on a 64-bit Ubuntu 14.04.5 system with an 8-core 2.4GHz Intel Xeon E5-2650 CPU and 64 GB memory. Unless otherwise specified, we set $\beta = 0.1 \times t$ in our experiment.

Datasets. Since *MINTSS* follows the settings of Kempe et al.’s work [39] that evaluates the influence spread by 10000 Monte Carlo samples, it can only be implemented on small scale graph. Thus, we use *NetHEPT* [4] to assess *GSM* and *MINTSS*. And the performance of *GSM* further is evaluated on large networks, *Pokec*, *LiveJournal*, *Orket* from SNAP [3] and *Twitter* [5] with 1.4 billion edges. Experiments on varying β are also conducted on the dataset *Twitter* given $t \cdot n = 1000$. Other benchmarks incur prohibitive computing overhead on this network.

Figure 5.8: Running time vs. t under the IC model for GSM and $MINTSS$. Figure 5.9: Running time vs. t and vs. β under the IC model for GSM .

5.2.2 Results under the IC Model

Figure 5.8 a illustrates the running time of GSM and $MINTSS$ on database of *NetHEPT*, when $\beta = 0.1 \times t$. GSM runs orders of magnitude faster compared to $MINTSS$, especially when tn is set to a small number. It is because $MINTSS$ follows the traditional greedy framework. Even for a small dataset *NetHEPT*, $MINTSS$ needs at least hundreds of seconds processing the query. Note that, unlike the running of IMM is almost irrelevant to k , GSM is sensitive to the value of tn . As k and tn increase, both IMM and GSM incur more iterations to find the seed set, but the sampling set size θ in IMM also decreases.

We conduct more experiments of GSM on larger graphs *Pokec*, *LiveJournal*, and *Orkut* as shown in Figure 5.8 b, c and d. Other benchmarks cannot handle graphs like this with millions of edges in 12 hours. In comparison, GSM is able to return corresponding results within at most two minutes.

To explore the limits of GSM , we further apply it on the *Twitter* with a billion of edges. As demonstrated in Figure 5.9 a, the algorithm is able to return the result within

 Figure 5.10: Running time vs. t under the LT model for *GSM* and *MINTSS*.

 Figure 5.11: Running time vs. t and vs. **beta** under the LT model for *GSM*.

half a hour, even if tn is set to a large number. Figure 5.9 b illustrates the running time of different β setting given $tn = 1000$. As we can see, with a larger β , *GSM* terminates faster at the sacrifice of its approximation. If we tune β to a very small number, the running time can be prohibitive.

5.2.3 Results under the LT Model

The experimental results on LT model of *GSM* and *MINTSS* show similar pattern of the IC counterparts, as shown in Figure 5.10 a. Under the setting $\beta = 0.1 \times t$, *GSM* can return the result of seed minimization query on *NetHEPT* dataset at least three orders of magnitude faster than *MINTSS*. Since the computation on the LT model is more easily implemented than the one on the IC model, both *GSM* and *MINTSS* decrease their running time compared to the IC results with the same setting. In particular, for a small tn setting of 200, *GSM* is able to terminate in 0.01 second.

We also evaluate the performance of *GSM* on larger datasets. Due to the same

reason, *MINTSS* cannot finish a seed minimization query on $\beta = 0.1 \times t$ within 12 hour on the LT model. Figure 5.10 b, c and d indicate that *GSM* can easily handle millions of edges and return the result in few minutes. The experimental results of *GSM* on LT model also slightly outperform IC model, due to the easy calculation of the LT propagation.

And similarly, running time results vs t and β are experimentally evaluated on a large dataset *Twitter*. Figure 5.11 a and b show the result on different t and β settings, respectively. *GSM* terminates slightly faster on various settings on the LT model compared to their IC counterparts.

Chapter 6

Conclusion and Future Work

In this chapter, we first present our conclusion based on our previous works. Next, we show our future work in this area.

6.1 Conclusions

To conclude, this thesis studies two problems, i.e., the influence maximization and the seed minimization problems on large scale real world social networks.

First we propose *IMM*, an efficient influence maximization that runs in $O((k + \ell)(n + m) \log n / \varepsilon^2)$ expected time. In theoretical side, it returns $(1 - 1/e - \varepsilon)$ -approximate solutions with at least $1 - n^{-\ell}$ probability. In practical side, we experimentally test our algorithm on large networks with up to 1.4 billion edge, and it outperforms various state-of-the-art techniques under most scenarios. We further extend *IMM* to the time-continuous model.

Then, we study seed minimization problem, and present a greedy algorithm *GSM*. *GSM* utilizes the superior estimation tool to achieve an efficient performance. And we prove that our algorithm returns a $\lceil (1 + \phi) \log(tn) \rceil$ -approximate solution with an absolute oracle. The experimental results illustrate that *GSM* has a smaller time consumption compared to the benchmarks in various settings.

6.2 Future Works

One prerequisite of *IMM* is to store the whole graph into the main memory. Thus, the high space consumption hinders it from processing a larger network exceeding 64GB. It inspires us to further expand the usage of *IMM* in distributed system, or partitioned graphs.

IMM also requires the graph to be a static one. In a dynamic graph, nodes and edges can be removed or inserted. If we can update RR sets for each modification of the dynamic graph, we are able to apply the influence maximization problem on dynamic graphs.

And the most of existing influence maximization algorithms are non-adaptive, i.e., the k -size seed set is selected before we evaluate the influence spread of these k nodes. Under an adaptive setting, we select k seed sets in batches, each of which is of equal size b . And the choice i -th batch is determined after we obtain the result of $(i - 1)$ -th batch. The adaptive problem is derived from the scenario where a company sends out free samples in batches, and the choice of selected users in a certain batch depends on how influenced nodes are distributed in the graph in previous batches.

These directions of future works can also fit in the seed minimization problems. And except it, influence spread on social networks has various applications simulating real world scenarios. By designing a more sophisticated weight functions and limiting conditions, we are able to explore other perspectives of the seed minimization problem.

References

- [1] <https://en.wikipedia.org/wiki/Twitter>.
- [2] <https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-w>
- [3] <https://snap.stanford.edu/data/>.
- [4] <http://research.microsoft.com/en-us/people/weic/graphdata.zip>.
- [5] <http://an.kaist.ac.kr/traces/WWW2010.html>.
- [6] E. Bakshy, I. Rosenn, C. Marlow, and L. Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.
- [7] N. Barbieri, F. Bonchi, and G. Manco. Topic-aware social influence propagation models. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 81–90. IEEE, 2012.
- [8] S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. In *WINE*, pages 306–311, 2007.
- [9] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier. Maximizing social influence in n-early optimal time. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 946–957. SIAM, 2014.
- [10] N. Chen. On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415, 2009.
- [11] S. Chen. Flashlogging: exploiting flash devices for synchronous logging performance. In *SIGMOD Conference*, pages 73–86, 2009.

- [12] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang. Online topic-aware influence maximization. *Proceedings of the VLDB Endowment*, 8(6):666–677, 2015.
- [13] W. Chen, W. Lu, and N. Zhang. Time-critical influence maximization in social networks with time-delayed diffusion process. In *AAAI*, 2012.
- [14] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, pages 1029–1038, 2010.
- [15] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, pages 199–208, 2009.
- [16] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*, pages 88–97, 2010.
- [17] F. R. K. Chung and L. Lu. Concentration inequalities and martingale inequalities: A survey. *Internet Mathematics*, 3(1):79–127, 2006.
- [18] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Syst. Sci.*, 55(3):441–453, 1997.
- [19] E. Cohen, D. Delling, T. Pajor, and R. F. Werneck. Sketch-based influence maximization and computation: Scaling up with guarantees. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 629–638. ACM, 2014.
- [20] G. Cordasco, L. Gargano, and A. A. Rescigno. Influence propagation over large scale social networks. In *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1531–1538, Aug 2015.
- [21] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [22] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 57–66, New York, NY, USA, 2001. ACM.

- [23] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, pages 3147–3155, 2013.
- [24] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.
- [25] M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *ICML*, pages 561–568, 2011.
- [26] M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Modeling information propagation with survival theory. In *ICML*, pages 666–674, 2013.
- [27] M. Gomez-Rodriguez and B. Schölkopf. Influence maximization in continuous time diffusion networks. In *ICML*, 2012.
- [28] A. Goyal, F. Bonchi, L. V. Lakshmanan, and S. Venkatasubramanian. Approximation analysis of influence spread in social networks. *arXiv preprint arXiv:1008.2005*, 2010.
- [29] A. Goyal, F. Bonchi, L. V. Lakshmanan, and S. Venkatasubramanian. On minimizing budget and time in influence propagation over social networks. *Social network analysis and mining*, 3(2):179–192, 2013.
- [30] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, pages 241–250, 2010.
- [31] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 5(1):73–84, 2011.
- [32] A. Goyal, W. Lu, and L. V. S. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *WWW*, pages 47–48, 2011.
- [33] A. Goyal, W. Lu, and L. V. S. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *ICDM*, pages 211–220, 2011.

- [34] M. Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.
- [35] A. Guille, H. Hacid, C. Favre, and D. A. Zighed. Information diffusion in online social networks: A survey. *ACM Sigmod Record*, 42(2):17–28, 2013.
- [36] J. Guo, P. Zhang, C. Zhou, Y. Cao, and L. Guo. Personalized influence maximization on social networks. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 199–208. ACM, 2013.
- [37] K. Huang, S. Wang, G. Bevilacqua, X. Xiao, and L. V. Lakshmanan. Revisiting the stop-and-stare algorithms for influence maximization. *Proceedings of the VLDB Endowment*, 10(9):913–924, 2017.
- [38] K. Jung, W. Heo, and W. Chen. Irie: Scalable and robust influence maximization in social networks. In *ICDM*, pages 918–923, 2012.
- [39] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.
- [40] D. Kempe, J. M. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *ICALP*, pages 1127–1138, 2005.
- [41] J. Kim, S.-K. Kim, and H. Yu. Scalable and parallelizable processing of influence maximization for large-scale social networks. In *ICDE*, pages 266–277, 2013.
- [42] J. F. Lawless. *Statistical Models and Methods for Lifetime Data*. Wiley-Interscience, 2002.
- [43] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.
- [44] Y. Li, J. Fan, Y. Wang, and K.-L. Tan. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

- [45] Y. Li, D. Zhang, and K.-L. Tan. Real-time targeted influence maximization for online advertisements. *Proceedings of the VLDB Endowment*, 8(10):1070–1081, 2015.
- [46] B. Liu, G. Cong, D. Xu, and Y. Zeng. Time constrained influence maximization in social networks. In *ICDM*, pages 439–448, 2012.
- [47] Q. Liu, B. Xiang, E. Chen, H. Xiong, F. Tang, and J. X. Yu. Influence maximization over large-scale social networks: A bounded linear approach. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 171–180. ACM, 2014.
- [48] C. Long and R. C. Wong. Minimizing seed set for viral marketing. In *2011 IEEE 11th International Conference on Data Mining*, pages 427–436, Dec 2011.
- [49] W. Lu, S. Chen, K. Li, and L. V. Lakshmanan. Show me the money: Dynamic recommendations for revenue maximization. *Proceedings of the VLDB Endowment*, 7(14):1785–1796, 2014.
- [50] H. T. Nguyen, T. N. Dinh, and M. T. Thai. Cost-aware targeted viral marketing in billion-scale networks. In *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*, pages 1–9. IEEE, 2016.
- [51] H. T. Nguyen, M. T. Thai, and T. N. Dinh. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *Proceedings of the 2016 International Conference on Management of Data*, pages 695–710. ACM, 2016.
- [52] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [53] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 61–70, New York, NY, USA, 2002. ACM.
- [54] T. C. Schelling. *Micromotives and macrobehavior*. WW Norton & Company, 2006.

- [55] L. Seeman and Y. Singer. Adaptive seeding in social networks. In *FOCS*, pages 459–468, 2013.
- [56] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 807–816. ACM, 2009.
- [57] Y. Tang, Y. Shi, and X. Xiao. Influence maximization in near-linear time: A martingale approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1539–1554. ACM, 2015.
- [58] Y. Tang, X. Xiao, and Y. Shi. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 75–86. ACM, 2014.
- [59] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.
- [60] X. Wang, Y. Zhang, W. Zhang, X. Lin, and C. Chen. Bring order into the samples: A novel scalable method for influence maximization. *IEEE Transactions on Knowledge and Data Engineering*, 29(2):243–256, 2017.
- [61] Y. Wang, G. Cong, G. Song, and K. Xie. Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *KDD*, pages 1039–1048, 2010.
- [62] P. Wicks. The als ice bucket challenge—can a splash of water reinvigorate a field?, 2014.
- [63] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.
- [64] P. Zhang, W. Chen, X. Sun, Y. Wang, and J. Zhang. Minimizing seed set selection with probabilistic coverage guarantee in a social network. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1306–1315. ACM, 2014.

Publication Lists

- **Influence Maximization in Near-Linear Time: A Martingale Approach**
Youze Tang, **Yanchen Shi**, Xiaokui Xiao
Proceedings of the ACM International Conference on Management of Data (**SIGMOD**),
pages 1539-1554, 2015.
- **Influence Maximization: Near-Optimal Time Complexity Meets Practical Efficiency.**
Youze Tang, Xiaokui Xiao, **Yanchen Shi**
Proceedings of the ACM International Conference on Management of Data (**SIGMOD**),
pages 75-86, 2014.