SOCIAL NETWORK ARCHITECTURES AND APPLICATIONS

A Dissertation
Submitted to
THE TEMPLE UNIVERSITY GRADUATE BOARD

in Partial Fulfillment
of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

by

Huanyang Zheng
December, 2017

Examining Committee Members:

Dr. Jie Wu, Dept. of Computer and Information Sciences (Advisor)
Dr. Bo Ji, Dept. of Computer and Information Sciences
Dr. Jamie Payton, Dept. of Computer and Information Sciences
Dr. Wei-Shih Yang, Dept. of Mathematics (External reader)

ABSTRACT

SOCIAL NETWORK ARCHITECTURES AND APPLICATIONS

by

Huanyang Zheng

Rather than being randomly wired together, the components of complex network systems are recently reported to represent a scale-free architecture, in which the node degree distribution follows power-law. While social networks are scale-free, it is natural to utilize their structural properties in some social network applications. As a result, this dissertation explores social network architectures, and in turn, leverages these architectures to facilitate some influence and information propagation applications.

Social network architectures are analyzed in two different aspects. The first aspect focuses on the node degree snowballing effects (i.e., degree growth effects) in social networks, which is based on an age-sensitive preferential attachment model. The impact of the initial links is explored, in terms of accelerating the node degree snowballing effects. The second aspect focuses on Nested Scale-Free Architectures (NSFAs) for social networks. The scale-free architecture is a classic concept, which means that the node degree distribution follows the power-law distribution. 'Nested' indicates that the scale-free architecture is preserved when low-degree nodes and their associated connections are iteratively removed. NSFA has a bounded hierarchy.

Based on the social network structure, this dissertation explores two influence propagation applications for the Social Influence Maximization Problem (SIMP). The first application is a friend recommendation strategy with the perspective of social influence maximization. For the system provider, the objective is to recommend a fixed number of new friends to a given user, such that the given user can maximize

his/her social influence through making new friends. This problem is proved to be NP-hard by reduction from the SIMP. A greedy friend recommendation algorithm with an approximation ratio of $1 - e^{-1}$ is proposed. The second application studies the SIMP with the crowd influence, which is NP-hard, monotone, non-submodular, and inapproximable in general graphs. However, since user connections in Online Social Networks (OSNs) are not random, approximations can be obtained by leveraging the structural properties of OSNs. The modularity, denoted by Δ, is proposed to measure to what degree this problem violates the submodularity. Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1 - e^{-1/(\Delta+1)}$, respectively.

Beside the influence propagation applications, this dissertation further explores three different information propagation applications. The first application is a social network quarantine strategy, which can eliminate epidemic outbreaks with minimal isolation costs. This problem is NP-hard. An approximation algorithm with a ratio of 2 is proposed through utilizing the problem properties of feasibility and minimality. The second application is a rating prediction scheme, called DynFluid, based on the fluid dynamics. DynFluid analogizes the rating reference among the users in OSNs to the fluid flow among containers. The third application is an information cascade prediction framework: given the social current cascade and social topology, the number of propagated users at a future time slot is predicted. To reduce prediction time complexities, the spatiotemporal cascade information (a larger size of data) is decomposed to user characteristics (a smaller size of data) for subsequent predictions. All these three applications are based on the social network structure.

Keywords: network architecture, preferential attachment, nested scale-free, friend recommendation, social influence maximization, social network quarantine, rating prediction, information cascade.

To my beloved parents

ACKNOWLEDGEMENTS

For the past several years in the Center for Networked Computing (CNC) of Temple University CIS department, I have benefited from the collaborations with brilliant researchers. I always enjoyed the excellent working environment and harmonious atmosphere in this department.

This research could not have been finished without the guidance of my advisory committee members, support from my family, and the help of friends. I would like to express my sincere gratitude to my advisor, Dr. Jie Wu, for his selfless help, immeasurable caring, strict guidance, and patience. He has taught me how to think comprehensively and critically, shown me the importance of persistence and hardworking to success, and provided me with solid supports. I take him as my role model, and I will never forget his guidance.

I would like to thank my committee members, Dr. Bo Ji , Dr. Jamie Payton, and Dr. Wei-Shih Yang, for their sincere help, extensive knowledge, and valuable comments to complete this dissertation.

My special thanks go to Dr. Paul Lafollette, Dr. John Fiore, and Dr. Avinash Srinivasan. It was my pleasure to work with them as a teaching assistant. I deeply appreciate their help and support. I also would like to thank my lab-mates and colleagues for the precious time that we spent together.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

CHAPTER 1

INTRODUCTION

1.1 Motivation

In past decades, scientists tacitly assumed components of complex network systems (such as the society and the Internet) to be randomly wired together, which is impractical for real networks. The small-world theory has shown that the real networks have much larger clustering coefficients than the random networks, and have much smaller network diameters than the grid networks. Recently, an avalanche of research has shown that many real networks, independent of their age, function, and scope, converge to similar architectures. These architectures are abstracted as the scale-free network architectures, in which the node degree distribution follows power-law. Examples of scale-free networks include social networks, citation networks, product review networks, autonomous systems, and so on. While social networks are generally scale-free, it is natural to utilize their structural properties in some social network applications. As a result, this dissertation explores social network architectures, and in turn, leverages these architectures to facilitate some influence and information propagation applications for social networks.

1.2 Challenges

This dissertation involves numerous challenges for social network architectures. The first challenge is to understand the social network architecture in a time-evolving manner. Although the preferential attachment model has been proposed

[1] to describe the evolving behaviors of social networks, it is overly simplified. The preferential attachment model assume that the degree growth potential of a node in the social network only depend on the current degree of that node, i.e., independent of other node properties. The second challenge is to further explore the social network architecture in addition to the scale-free property [2], which means that the node degree distribution follows power-law. Although the scale-free property has been found for a long time, no property is found in addition to it.

The influence propagation applications are challenging. Currently, the influence propagation model is mainly applied in the virtual marketing field. Can we apply the influence propagation model to other fields, such as the friend recommendation? This dissertation explores such applications in depth. One step further, almost all influence propagation models, such as the independent cascade and linear threshold, are submodular [3]. Submodularity means that the marginal influence gain of a person has diminishing return effects with respect to existing people. Few results [3] are provided when the influence propagation model even slightly violates the submodularity. Can we explore some non-submodular models?

The information propagation applications are also challenging. This is because there exists multiple types of information for social networks. Moreover, different type of information has a different way of propagation. This dissertation involves three types of information: epidemic (disease in real life or virus in computer networks), trust (trustor-trustee relationship among people), and cascade (post or forward information in online social network). What are the similarities and differences among different types of information during propagations? This dissertation has a fundamental exploration with respect to different types of information.

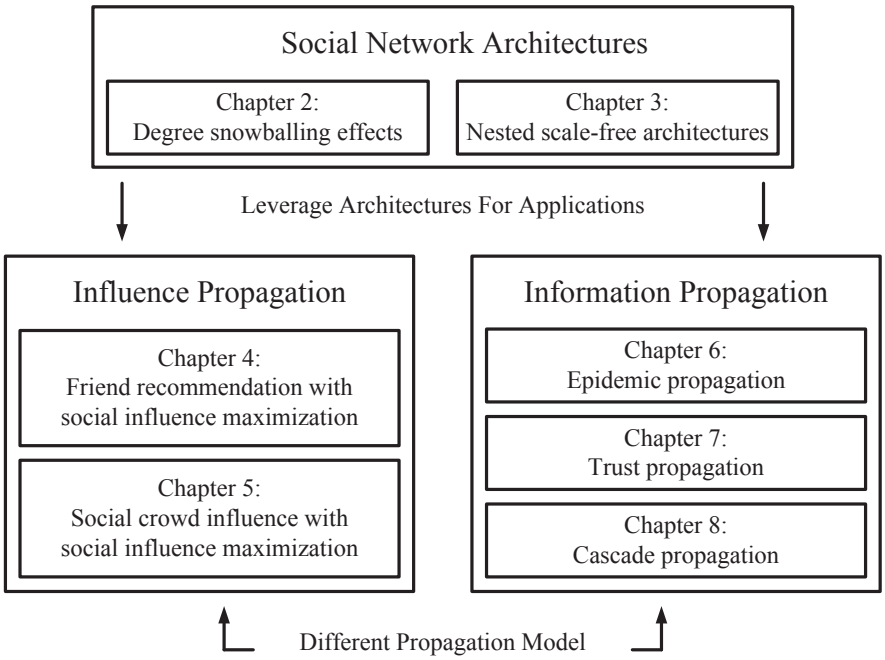


Figure 1.1: The dissertation overview.

1.3 Overview

The dissertation overview is shown in Fig. 1.1, which includes three parts. The first part, which includes Chapters 2 and 3, explores social network architectures. All the remaining chapters are social network applications that leverage these architectures in some way. The second part, which includes Chapters 4 and 5, focuses on the influence propagation applications with the independent cascade model. In contrast, the third part, which includes Chapters 6, 7, and 8, focuses on the information (i.e., epidemic, trust, and cascade) propagation applications. Note that the second and third parts have different types of propagation models for social networks.

Chapters 2 and 3 start with the social network architectures. Chapter 2 studies the node degree snowballing effects (i.e., degree growth effects) in social networks through an age-sensitive preferential attachment model. In this model, nodes are iteratively added one by one to a growing network. Upon entering the network, each new node connects to a suitably chosen set of existing nodes, while the attachment

probability for an existing node to get connected depends on both its node degree and age difference. The impact of the initial links is explored, in terms of accelerating the node degree snowballing effects. If a new node enters the growing network with more initial links (a larger degree), it could attract many more links from the later nodes, and thus, its degree snowballs faster. This dissertation finds that the initial links are only impactful when neither the node degree nor the age difference dominates the attachment probability. In that case, the relationship between the ratio of the additional initial link and the gain ratio of the eventual node degree is shown to include two stages (linear stage and diminishing return stage). Chapter 3 studies a scalable publish/subscribe system for social networks, which are shown to have Nested Scale-Free Architectures (NSFAs). The scale-free architecture is a classic concept, which means that the node degree distribution follows the power-law distribution. 'Nested' indicates that the scale-free architecture is preserved when low-degree nodes and their associated connections are removed. NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies. By leveraging the NSFA's hierarchy, the proposed publish/subscribe system achieves a competitive tradeoff among the event routing efficiency, system robustness, and overhead. For a network with $|V|$ nodes, the number of routing hops for the event deliveries in the proposed system is expected to be $O(\ln \ln |V|)$. For the topological information, each node only needs to maintain an overhead with a constant size, $O(1)$. Node arrival, departure, and failure can be handled within a message complexity of $O(\ln \ln |V|)$.

Chapters 4 and 5 focus on the influence propagation applications for the Social Influence Maximization Problem (SIMP), which is one of the most fundamental problems for social networks. The SIMP aims to select k initially-influenced seed users to maximize the number of eventually-influenced users. Under the independent cascade model, the SIMP has been proved to be NP-hard, monotone, and submodular.

Therefore, a naive greedy algorithm that maximizes the marginal gain obtains an approximation ratio of $1-e^{-1}$. Chapter 4 proposes a friend recommendation strategy with the perspective of social influence maximization. In online social networks, sometimes people make new friends to maximize their social influences. For example, business page owners on Facebook want to influence as many people as possible for commercial advantages. For the system provider, the objective is to recommend a fixed number of new friends to a given user, such that the given user can maximize his/her social influence through making new friends. This problem is proved to be NP-hard by reduction from the SIMP. A greedy friend recommendation algorithm with an approximation ratio of $1 - e^{-1}$ is proposed, based on the submodular property. It involves a sub-problem of computing the influence spread. Through utilizing the social network structure, a novel method is proposed to solve this sub-problem with a tradeoff between the accuracy and the time complexity. Chapter 5 extends the SIMP by considering the crowd influence. This is because the crowd influence surpasses the combination of the independent influence from each person in the crowd. This problem is proved to be NP-hard and monotone, but not submodular. It is proved to be inapproximable within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$ in general graphs. However, since user connections in Online Social Networks (OSNs) are not random, approximations can be obtained by leveraging the structural properties of OSNs. The modularity, denoted by Δ, is proposed to measure to what degree this problem violates the submodularity. Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1 - e^{-1/(\Delta+1)}$, respectively.

Chapters 6, 7, and 8 focus on the information propagation applications, in terms of the epidemic propagation, trust propagation, and cascade propagation, respectively. Chapter 6 describes a social network quarantine application for diseases. Note that the notion of diseases has been extended from real human diseases to general epidemic information propagations, such as the rumors in distributed systems. Controlling the

spread of a disease is usually done through quarantine, where people that have, or are suspected to have, a disease are isolated from having interactions with others. As a tradeoff, normal human interactions are inevitably degraded by the quarantine. Therefore, a robust quarantine strategy that can eliminate epidemic outbreaks with minimal isolation costs. This problem is shown to be NP-hard. A bounded algorithm with an approximation ratio of 2 is proposed, through utilizing the feasibility and minimality properties. Chapter 7 describes a rating prediction application for OSNs through trust propagations. In trust-based recommendation systems, if a user is predicted to have a high rating of a product, then this product is recommended to that user for shopping potential. Therefore, rating predictions are critical for qualified recommendations. Based on the fluid dynamics theory, a novel rating prediction scheme called DynFluid is proposed. The key observation is that the rating of a user depends on his/her user experience, as well as the ratings of other users. For example, users may refer to friends' ratings upon rating a product, themselves. DynFluid analogizes the rating reference among the users to the fluid flow among containers: each user is represented by a container; the rating of a user is mapped to be the fluid temperature in the corresponding container. Two user characteristics, persistency and persuasiveness, are also incorporated into DynFluid. Chapter 8 describes a cascade prediction application for OSNs. Information cascades occur when people observe the actions of others (followees) and then make the same choices that the others have made (followers). Cascade predictions are important, since they can detect and help resist bad cascades. This dissertation focuses on photo cascade predictions in Flickr: given the current cascade and social topology, we want to predict the number of propagated users at a future time slot. Information cascades include a large amount of data that crosses both space and time. To reduce prediction time complexities, the idea is to decompose the spatiotemporal cascade information (a larger size of data) to user characteristics (a smaller size of data) for subsequent predictions. Space and

time matrices are introduced to record the cascade information. This dissertation introduces a set of new notions, persuasiveness and receptiveness (represented as two vectors for complexity reduction), to capture characteristics of followees and followers. Persuasiveness includes followees' abilities to propagate information, while receptiveness includes followers' willingness to accept information. A three-stage parallel prediction scheme is proposed as follows. (1) We map the spatiotemporal cascade information to a weighted matrix, in which the weights of space and time information are tuned. (2) Singular value decomposition is used to extract nodes' persuasiveness and receptiveness from the weighted matrix. (3) Predictions are conducted based on nodes' persuasiveness and receptiveness.

1.4 Contributions

The contributions of this dissertation can be summarized as follows:

- The node degree snowballing effects in social networks are analyzed based on an age-sensitive preferential attachment model. The impact of the initial links is explored in terms of the node degree snowballing effects.
- Nested Scale-Free Architectures (NSFAs) are proposed for scale-free social networks. Nested indicates that the scale-free architecture is preserved when low-degree nodes and their associated connections are iteratively removed.
- A friend recommendation strategy with the perspective of social influence maximization is proposed. A greedy friend recommendation algorithm with an approximation ratio of $1 - e^{-1}$ is proposed.
- The SIMP with the crowd influence is studied and is proved to be NP-hard, monotone, non-submodular, and inapproximable in general graphs. However, since OSNs are not general graphs, two approximation algorithms are proposed.

- The problem of eliminating epidemic outbreaks with minimal isolation costs is explored. It is proved to be NP-hard. A social network quarantine algorithm is proposed with an approximation ratio of 2.
- A rating prediction scheme, called DynFluid, is proposed based on the fluid dynamics. DynFluid analogizes the rating reference among the users in OSNs to the fluid flow among containers.
- A cascade prediction framework is proposed for OSNs. To reduce the time complexity, the spatiotemporal cascade information (a larger size of data) is decomposed to user characteristics (a smaller size of data).

CHAPTER 2

SNOWBALLING EFFECTS IN PREFERENTIAL ATTACHMENT: THE IMPACT OF THE INITIAL LINKS

Chapters 2 and 3 focus on the social network architectures. More specifically, this chapter studies the node degree snowballing effects (i.e., degree growth effects) in the age-sensitive preferential attachment model, where nodes are iteratively added one by one to a growing network. Upon entering the network, each new node connects to a suitably chosen set of existing nodes, while the attachment probability for an existing node to get connected depends on both its node degree and age difference. We are interested in accelerating the node degree snowballing effects through the impact of the initial links. If a new node enters the growing network with more initial links (a larger degree), it could attract many more links from the later nodes, and thus, its degree snowballs faster. We find that the initial links are only impactful when neither the node degree nor the age difference dominates the attachment probability. In that case, the relationship between the ratio of the additional initial link and the gain ratio of the eventual node degree is shown to include two stages (linear stage and diminishing return stage). Applications of our work involve citation networks and online social networks. For example, in citation networks, we answer the question that whether an author can attract additional citations through self-citations. Finally, real data-driven experiments verify the accuracies of our results, which cast some new light in real-world growing networks.

2.1 Introduction

One of the most impressive recent discoveries in the field of network evolution is the observation that a number of large growing networks are scale-free [1, 4, 2]. Their key feature is that the node degree distributions have a power-law form [5, 6]. Typical scale-free networks include the citation networks, the online social networks, the World Wide Web, and so on. The preferential attachment model is one of the most acknowledged models for explaining the formation of scale-free networks [7]. In this model, nodes are iteratively added one by one to a growing network (one new node per time unit). Upon entering the network, each new node connects to a suitably chosen set of existing nodes, while the attachment probability for an existing node to get connected is proportional to its degree. Therefore, the existing node with a large degree is preferentially attached, resulting in the *degree snowballing effect* (i.e., degree growth effect), in which the rich get richer.

We are interested in accelerating such degree snowballing effects through the impact of the initial links. The initial links of a node are the links set by that node at the time when it enters the growing network. If a new node enters the growing network with more initial links, it could attract many more links from the later nodes, and thus its degree snowballs faster. While the impact of the initial links remains unexplored, it has important applications as follows. (1) In citation networks, we are more likely to cite papers with high citations than that with low citations. Then, if an author cites his/her own papers (self-citation), is it possible for the papers of this author to gain extra citations at a later time by the snowballing effect? (2) In online social networks such as Facebook and Twitter, business pages want to attract more followers, as to propagate the product information for sales. Then, if a business page makes an advertisement (e.g., Facebook page promotion [8]), how many additional followers can this business page attract at a later time by the snowballing effect? The

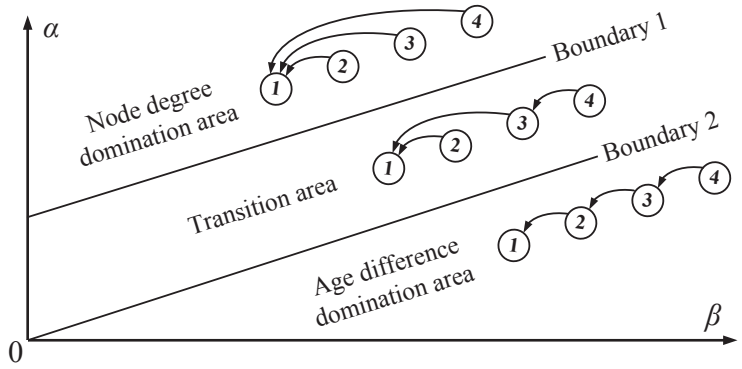


Figure 2.1: The age-sensitive preferential attachment.

impact of the initial links is explored in our study, which is critical for the development of the network science.

To be more realistic, here we study the degree snowballing effects in *age-sensitive preferential attachment models*, where the attachment probability is age-sensitive [9]. For example, in citation networks, we prefer to cite recent papers more than old papers. Specifically, we consider that the attachment probability for an existing node to get connected is proportional to $d^{\alpha} \cdot \Delta t^{-\beta}$. Here, d is the degree of the existing node, while Δt is the age difference (also entry time difference) between the new node and the existing node. α and β ($\alpha > 0$ and $\beta > 0$) are parameters obtained by existing estimators [10]. The age-sensitive model can reduce to the classic model when $\alpha = 1$ and $\beta = 0$. Then, in terms of the attachment probability, there exists a tradeoff between the attractiveness brought by the node degree and the repulsiveness brought by the age difference. Although older nodes have larger degrees, they may not attract more links from the new nodes, due to the larger age difference. An example is shown in Fig. 2.1, where the nodes enter the growing network one by one (following their IDs). Upon entering the network, the node connects to existing nodes according to the node degree and the age difference. It can be seen that the resulting network structure of the age-sensitive preferential attachment model depends on α and β (age difference domination area, transition area, and node degree domination area).

The snowballing effects in age-sensitive preferential attachment models are more intriguing and challenging. With respect to α and β, how does the impact of the initial links vary? Since those three areas in Fig. 2.1 result in different network structures, the impact of the initial links should be qualitatively different. Moreover, is the amount of the initial links important? While a small amount of the initial links leads to a limited change, a large amount of the initial links may lead to a big change.

Our results and contributions are summarized as follows:

- Percolation phenomena are found in the age-sensitive preferential attachment models. Boundaries 1 and 2 in Fig. 2.1 are $\alpha = \beta + 1.5$ and $\alpha = \beta$, respectively. We show that the initial links are not impactful in the node degree domination area and the age difference domination area.
- We show that the initial links are only impactful in the transition area of $\beta \leq \alpha \leq \beta + 1.5$. In that case, the impact of the initial links is found to have two stages (linear stage and diminishing return stage). We further show that the initial links are most impactful, when the corresponding growing network lies in the "middle" of the transition area.
- Accuracies of our theoretical results are verified. The degree snowballing effects are observed in the real-world citation network and online social network.

The remainder of this chapter is organized as follows. Section 2.2 is the related work. In Section 2.3, we set up the model and formulate the problem. In Section 2.4, the impact of the initial links is analyzed. Section 2.5 includes the experiments. Section 2.6 shows the conclusion. Proofs are presented in the Appendix.

2.2 Related Work

The classic preferential attachment model was proposed in [11] with a well-studied body of knowledge in the network science. The aging effects have been observed. For example, Wang et al. [12] studied the predictability of the citation patterns with respect to different time slots. Zhao et al. [13] explored the multi-scale dynamics of time-sensitive information propagations. Authors in [9, 14] studied the scale-free properties in the age-sensitive preferential attachment models, in terms of the degree distributions and the clustering properties. The aging effects are preliminarily explored in the citation networks [15], and then are found in the online social networks [16], the World Wide Web [17], the recommendation systems [18], and so on. These works mainly focus on the scale-free properties, where the node degree distribution follows power-law form. In contrast, we explore the degree snowballing effects in growing networks, which are completely novel.

The other existing findings that are highly related to the snowballing effects include the rich-get-richer phenomenon, the "Matthew effect" [19], and the cumulative advantage [20]. Although they have been empirically confirmed for a long time with respect to the economic market, quantitative studies have not been conducted for the citation networks and the online social networks. For example, Kumar et al. [21] studied the equilibrium states of two-sided market evolution through an empirical analysis on the cumulative capital advantage. Braha et al. [22] simulated the corporate competition in the preferential attachment model with respect to the snowball effect. Kas et al. [23] studied the structures and statistics of citation networks. However, they did not consider the impact of the initial links, which is explored in this chapter.

2.3 Model and Problem Formulation

2.3.1 Preferential Attachment Model

In the preferential attachment model [7], nodes are iteratively added one by one to a growing network (one new node per time unit). The node added at the time s is denoted as N_s, while the current time is denoted as t ($t \geq s$). The age of a node is its existing time in the growing network, i.e., the age of the node N_s is $t-s$. Upon entering the network, each new node connects to a suitably chosen set of existing nodes, while the attachment probability for an existing node to get connected is proportional to $d^{\alpha} \cdot \Delta t^{-\beta}$. Here, d is the degree of the existing node, while Δt is the age difference (also entry time difference) between the new node and the existing node. α and β ($\alpha > 0$ and $\beta > 0$) are parameters obtained by existing estimators [10]. The initial links of a node are the links set by that node at the time when it enters the growing network. We assume that each new node sets m new links to the existing nodes. The links are directional, while the node degree is the summation of its in-degree and out-degree. Let $d(s,t)$ denote the expected degree of the node s at the time t ($t \geq s$), while the initialization condition is $d(s,s) = m$.

Since an existing node will get attached by later nodes, the degree of an existing node snowballs with respect to the time. However, in terms of the snowballing speed, there exists a tradeoff between the attractiveness brought by the node degree (with a larger α being more attractive) and the repulsiveness brought by the age difference (with a larger β being more repulsive). Although older nodes have larger degrees, they may not attract more links from the new nodes, due to the larger age differences. As previously shown in Fig. 2.1, the resulting network structure of the age-sensitive preferential attachment model depends on α and β (age difference domination area, transition area, and node degree domination area).

2.3.2 Problem Formulation

This chapter studies the impact of the initial links in the age-sensitive preferential attachment models. While a normal node enters the growing network with only m links, we focus on a particular node that enters the network with additional m' links ($m + m'$ links in total), as to observe the impact of the initial links. Since a larger degree means a larger attachment probability, the additional initial links can accelerate the degree snowballing effects for the nodes in the growing network. Our study has important applications as follows.

- In citation networks, we are more likely to cite papers with a high number citations than that with a low number of citations. Then, if an author cites his/her own papers (self-citation), is it possible for the papers of this author to gain extra citations at a later time? In this scenario, m and m' represent the average paper citations and the number of self-citations, respectively.
- In online social networks such as Facebook and Twitter, business pages want to attract more followers, as to propagate the product information for sales. Then, if a business page makes an advertisement (e.g., Facebook page promotion [8]), how many additional followers can this business page attract at a later time by the snowballing effect? Here, m' can be interpreted as the number of followers attracted by the advertisement.

For the simplicity of the following analysis, we define the *initial rate* (r_i) as the ratio of the additional initial links to the normal initial links (i.e., $r_i = m'/m$). A larger initial rate means that the corresponding node has a larger initial degree.

If the node N_s enters the network with an additional m' links, then we use $d'(s,t)$ to denote its expected degree at the time t ($t \geq s$). Its initialization condition is $d'(s,s) = m + m'$. We are interested in the ratio of the node degree gain brought by the additional initial links, which is defined as the *gain rate* (denoted by r_g). In other

words, we have:

$$r_g = \frac{d'(s,t) - d(s,t)}{d(s,t)} \tag{2.3-1}$$

The objective of this chapter is to study *the relationship between the initial rate and the gain rate*, which represents the impact of the initial links in the growing networks. A larger initial rate should bring a non-smaller gain rate. We also want to study how this relationship changes with respect to the parameters α, β, s, and t. Note that, α and β indicate the attractiveness brought by the node degree and the repulsiveness brought by the age difference, respectively. Therefore, the values of α and β are also important for the initial links to be impactful in the corresponding growing network. Meanwhile, s indicates the time for introducing the additional initial links. Our analyses are shown in the next section.

2.4 Snowballing Effects

This section studies the relationship between the initial rate and the gain rate, as to understand the impact of the initial links. First, we review the classic preferential attachment model. Then, we look into the snowballing effects within the node degree domination area and the age difference domination area of the age-sensitive model, respectively. Finally, we show the snowballing effects within the transition area.

2.4.1 Classic Preferential Attachment

In the classic preferential attachment model [7], the attachment probability for an existing node to get connected is only proportional to its degree ($\alpha = 1$ and $\beta = 0$). Let us start with the case for a normal node that enters the network with m links. Then, when a new node enters the network at the time $t + 1$, the attachment

probability for the node N_s to get connected is:

$$\frac{d(s,t)}{\sum_{s=1}^{t} d(s,t)} = \frac{d(s,t)}{2mt} \tag{2.4-2}$$

The denominator $\sum_{s=1}^{t} d(s,t)$ is the total degree, which is the normalization factor in Eq. 2.4-2. The total degree is $2mt$, since there are t nodes in the network and each node has brought m links. We assume that the attachment processes for the m links are independent of each other, and thus the expected degree gain of the node N_s is $m \times \frac{d(s,t)}{2mt} = \frac{d(s,t)}{2t}$. In other words, we have the following equation:

$$d(s,t+1) = d(s,t) + m \times \frac{d(s,t)}{2mt} = \frac{2t+1}{2t} d(s,t) \tag{2.4-3}$$

If we do the recursion in Eq. 2.4-3, then we can get:

$$\begin{aligned} d(s,t) &= \frac{2t-1}{2t-2} \times \frac{2t-3}{2t-4} \times \cdots \times \frac{2s+1}{2s} \times d(s,s) \\ &= \exp\Big\{ \ln \frac{2t-1}{2t-2} + \cdots + \ln \frac{2s+1}{2s} \Big\} \times d(s,s) \\ &\approx \exp\Big\{ \frac{1}{2t-2} + \cdots + \frac{1}{2s} \Big\} \times d(s,s) \\ &\approx \exp\Big\{ \frac{1}{2} \ln \frac{t}{s} \Big\} \times d(s,s) = m\sqrt{\frac{t}{s}} \end{aligned} \tag{2.4-4}$$

In Eq. 2.4-4, we have used the approximations of $\ln \frac{2s+1}{2s} \approx \frac{1}{2s}$ and $\sum_{x=s}^{t-1} \frac{1}{2x} \approx \int_s^t \frac{1}{2x} \mathrm{d}x$. Eq. 2.4-4 implies that the node degree has a square-root growth with respect to the ratio of the current time to the node entry time. Similar to Eq. 2.4-4, if the node N_s enters the network with an additional m' links, we can get:

$$d'(s,t) \approx \exp\Big\{ \frac{1}{2} \ln \frac{t}{s} \Big\} \times d'(s,s) = (m+m')\sqrt{\frac{t}{s}} \tag{2.4-5}$$

Eqs. 2.4-4 and 2.4-5 mean that *the gain rate equals the initial rate* (i.e., $r_g = r_i$) in the classic preferential attachment model. Here we have assumed that the number

of additional initial links is small (i.e., $m' \ll 2mt$). However, the relationship of $r_g = r_i$ is uncommon in real-world growing networks, since the prerequisite that the attachment probability is only proportional to the degree may not be true.

In the following three subsections, we will discuss the snowballing effects in the age-sensitive preferential attachment model, where the attachment probability is determined by both the node degree and the age difference. As previously mentioned, the attachment probability is proportional to $d^{\alpha} \cdot \Delta t^{-\beta}$. The tradeoff between the attractiveness brought by the node degree and the repulsiveness brought by the age difference divides the resulting network structure into three areas (age difference domination area, node degree domination area, and transition area). Each of the three following subsections corresponds to one of those three areas.

2.4.2 Age Difference Domination Area

This subsection studies the snowballing effects in the age-sensitive preferential attachment model, in which the age difference dominates the attachment probability. In other words, the attractiveness brought by the node degree is much smaller than the repulsiveness brought by the age difference. To study the snowballing effects, we first need to clarify the boundary of this area, as shown in the following theorem:

Theorem 2.1. *When $\alpha < \beta$, the first node will attract a finite number of links, with respect to the network growth.*

The proof of Theorem 2.1 is shown in Appendix 2.7.1. The basic idea of the proof is to show that, when $\alpha < \beta$, the first node is much less attractive than a younger node for a new node to attach. The insight behind Theorem 2.1 is that the age difference dominates the attachment probability, where the new nodes are more intended to link to the younger nodes. At this time, even if an old node has a very high degree, it will not be further attached to by the new nodes. The resulting network structure

for this case is illustrated in Fig. 2.2(a), where the nodes connect to each other one by one following their entry times. As for the snowballing effects, we have:

Theorem 2.2. *When $\alpha < \beta$, for the node N_s that enters the growing network at the time s, it needs at least $\Omega((t-s)^{\beta/\alpha})$ additional initial links to keep its attractiveness for nodes that enter the growing network at the time t.*

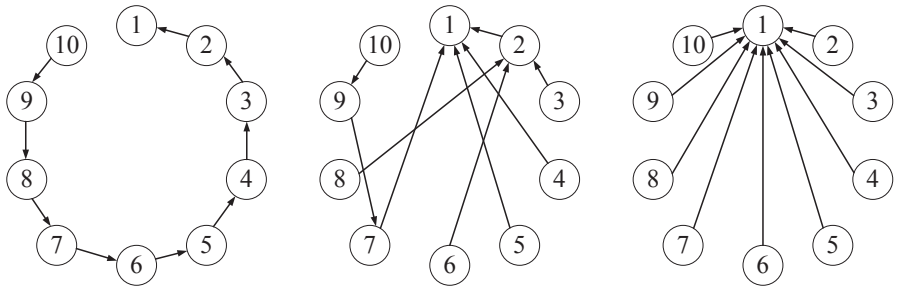
The proof of Theorem 2.2 is shown in Appendix 2.7.2. The basic idea of the proof is to show that, when $\alpha < \beta$, the node N_s needs many additional initial links to resist the dominated repulsiveness brought by the age difference. The insight behind Theorem 2.2 is that the initial links in the growing network with $\alpha < \beta$ are not impactful, since the initial links are wasted on resisting the dominated aging effects. In other words, the gain rate is close to zero, unless we have a very large initial rate (basically impossible for real-world growing networks).

2.4.3 Node Degree Domination Area

This subsection studies the snowballing effects in the age-sensitive preferential attachment model, in which the node degree dominates the attachment probability. In other words, the attractiveness brought by the node degree is much larger than the repulsiveness brought by the age difference. Similarly, we first need to clarify the boundary of this area, as shown in the following theorem:

Theorem 2.3. *When $\alpha > \beta + 1.5$, the first node will attract an infinite number of links, with respect to the network growth. The first node N_1 has a degree of $\Theta(t)$.*

The proof of Theorem 2.3 is shown in Appendix 2.7.3. The basic idea of the proof is to show that, when $\alpha > \beta + 1.5$, the first node is much more attractive than the remaining nodes for a new node to attach. The insight behind Theorem 2.3 is that the degree dominates the attachment probability, where the new nodes are more likely to attach to the oldest node. At this time, younger nodes will not be further attached



(a) $\alpha < \beta$ (b) $\beta \leq \alpha \leq \beta + 1.5$ (c) $\alpha > \beta + 1.5$

Figure 2.2: The percolation phenomena in the age-sensitive preferential attachment ($m = 1$ and $t = 10$). The ID of a node is its entry time.

by the new nodes, while the first node has a degree of $\Theta(t)$. In other words, the first node monopolizes the majority of the links. The resulting network structure for this case is illustrated in Fig. 2.2(c). Note that, the classic model with $\alpha = 1$ and $\beta = 0$ lies under the boundary of $\alpha = \beta + 1.5$, which has a qualitative difference with models in the node degree domination area. The first node only has a degree of $\Theta(\sqrt{t})$ when $\alpha = 1$ and $\beta = 0$. As for the snowballing effects, we have:

Theorem 2.4. *When $\alpha > \beta + 1.5$, for the node N_s that enters the growing network at the time s, it needs at least $\Omega(s^{\alpha-\beta})$ additional initial links to keep its attractiveness for later nodes.*

The proof of Theorem 2.4 is shown in Appendix 2.7.4. The basic idea of the proof is to show that, when $\alpha > \beta + 1.5$, the node N_s needs many additional initial links to compete with the node N_1, in terms of attracting the new attachments. Theorem 2.3 states that the first node N_1 has a degree of $\Theta(s)$ at the time s. The insight behind Theorem 2.4 is that a large number of additional initial links is needed to break the link monopoly of the node N_1. In other words, the gain rate is also close to zero, unless a very large initial rate is used. Therefore, the initial links in the growing network with $\alpha > \beta + 1.5$ are not impactful on the eventual node degree, the result of which is similar to that for the age difference domination area.

2.4.4 Transition Area

In the previous two subsections, Theorems 2.1 and 2.3 show that $\alpha = \beta$ and $\alpha = \beta + 1.5$ are two boundaries for the percolation phenomena in the age-sensitive preferential attachment models. When $\alpha < \beta$ or $\alpha > \beta + 1.5$, the resulting network structure turns out to be simplex. The resulting network structure for the transition area of $\beta \leq \alpha \leq \beta + 1.5$ is more complex. An example for the transition area is illustrated in Fig. 2.2(b). Meanwhile, Theorems 2.2 and 2.4 show that the initial links are not impactful in the age difference domination area and the node degree domination area, since the initial links are wasted to resist the dominated power.

This subsection discusses the snowballing effects in the transition area of $\beta \leq \alpha \leq \beta + 1.5$, based on [9]. Let us start with the case for a normal node that enters the network with m links. Similar to Eq. 2.4-3, we have:

$$d(s, t+1) = d(s, t) + m \times \frac{d(s, t)^{\alpha}(t-s)^{-\beta}}{\sum_{s=1}^{t} d(s, t)^{\alpha}(t-s)^{-\beta}} \tag{2.4-6}$$

When $\alpha = 1$ and $\beta = 0$, Eq. 2.4-6 is reduced to Eq. 2.4-3 (the classic model). Eq. 2.4-6 can also be written in the continuous form:

$$\frac{\partial d(s, t)}{\partial t} = m \times \frac{d(s, t)^{\alpha}(t-s)^{-\beta}}{\int_{1}^{t} d(s, t)^{\alpha}(t-s)^{-\beta} \mathrm{d}s} \tag{2.4-7}$$

Since Eq. 2.4-7 is very complex, we consider the node degree to be scaling ($d(s, t) \equiv d(s/t)$) [9]. In other words, the node degree is considered as a function of s/t . For notation simplicity, we set $\xi = s/t$. Then, Eq. 2.4-7 can be rewritten as:

$$\frac{1}{d(\xi)^{\alpha}} \times \frac{\mathrm{d}d(\xi)}{\mathrm{d}\xi} = \frac{-1}{\xi(1-\xi)^{\beta}} \frac{m}{\int_{0}^{1} d(\xi)^{\alpha}(1-\xi)^{-\beta} \mathrm{d}\xi} \tag{2.4-8}$$

If we do the integral in Eq. 2.4-8, we can get:

$$\frac{d(\xi)^{1-\alpha} - d(1)^{1-\alpha}}{1-\alpha} = \frac{m\int_1^{\xi} \frac{-1}{\xi(1-\xi)^{\beta}} \mathrm{d}\xi}{\int_0^1 d(\xi)^{\alpha}(1-\xi)^{-\beta}\mathrm{d}\xi} \tag{2.4-9}$$

When $\alpha \to 1$ and $\beta = 0$, Eq. 2.4-9 reduces to $\ln \frac{d(\xi)}{d(1)} = \frac{-1}{2}\ln \xi$ that is consistent with Eq. 2.4-4, i.e., $d(s,t) = m\sqrt{t/s}$. This is because $d(\xi)^{1-\alpha} \approx e^{(1-\alpha)\ln d(\xi)} \approx 1 + (1-\alpha)\ln d(\xi)$, when $\alpha \to 1$. The result in Eq. 2.4-9 can be rewritten as:

$$\begin{aligned} d(\xi) &= \left[m^{1-\alpha} + \frac{(1-\alpha)m\int_1^{\xi} \frac{-1}{\xi(1-\xi)^{\beta}} \mathrm{d}\xi}{\int_0^1 d(\xi)^{\alpha}(1-\xi)^{-\beta}\mathrm{d}\xi} \right]^{\frac{1}{1-\alpha}} \\ &\approx m \times [1 + (1-\alpha)C(\alpha,\beta,\xi)]^{\frac{1}{1-\alpha}} \end{aligned} \tag{2.4-10}$$

Eq. 2.4-10 is approximated by using $d(\xi) = m\xi^{-1/2}$ to calculate the normalization factor. This approximation is feasible, since it can represent the degree distribution in the transition area. Meanwhile, the $C(\alpha,\beta,\xi)$ in Eq. 2.4-10 is:

$$C(\alpha,\beta,\xi) = \frac{\int_1^{\xi} \frac{-1}{\xi(1-\xi)^{\beta}} \mathrm{d}\xi}{\int_0^1 \xi^{-\alpha/2}(1-\xi)^{-\beta}\mathrm{d}\xi} \tag{2.4-11}$$

Similar to Eq. 2.4-10, if the node N_s enters the network with m' additional links, then we can get:

$$\begin{aligned} d'(\xi) &= \left[(m+m')^{1-\alpha} + \frac{m\int_1^{\xi} \frac{-1}{\xi(1-\xi)^{\beta}} \mathrm{d}\xi}{\int_0^1 d(\xi)^{\alpha}(1-\xi)^{-\beta}\mathrm{d}\xi} \right]^{\frac{1}{1-\alpha}} \\ &\approx m \times [(1+\frac{m'}{m})^{1-\alpha} + (1-\alpha)C(\alpha,\beta,\xi)]^{\frac{1}{1-\alpha}} \end{aligned} \tag{2.4-12}$$

In Eq. 2.4-12, we have assumed that m' is small enough with respect to the normalization factor of $\int_0^1 d(\xi)^{\alpha}(1-\xi)^{-\beta}\mathrm{d}\xi$. Combining Eqs. 2.4-10, 2.4-11, and

2.4-12, the relationship between the initial rate and the gain rate can be obtained:

$$r_g = \left[\frac{(1+r_i)^{1-\alpha}+(1-\alpha)C(\alpha,\beta,\xi)}{1+(1-\alpha)C(\alpha,\beta,\xi)}\right]^{\frac{1}{1-\alpha}} - 1 \tag{2.4-13}$$

Note that, $C(\alpha,\beta,\xi)$ can be regarded as a constant with respect to r_i. Meanwhile, we have $C(1,0,\xi) = \frac{1}{2}\ln\xi$. Further analysis on Eq. 2.4-13 shows the following theorem:

Theorem 2.5. *When $|(1-\alpha)C(\alpha,\beta,\xi)| \gg 1$, the gain rate is close to zero (i.e., $r_g \approx 0$). When $|(1-\alpha)C(\alpha,\beta,\xi)| \ll 1$, the relationship between the initial rate and the gain rate satisfies $r_g \approx (1+r_i)e^{-C(\alpha,\beta,\xi)} - 1$.*

The proof of Theorem 2.5 is shown in Appendix 2.7.5. Theorem 2.5 shows three intriguing properties for the impact of the initial links, while the first property is that there is a prerequisite for the initial links to be impactful. This threshold results from the fact that either the dominated attractiveness brought by the node degree or the dominated repulsiveness brought by the age difference can weaken the impact of the initial links. This is similar to the case in the node degree domination area or the age difference domination area. If this threshold is satisfied, then the gain rate increases linearly with respect to the initial rate. However, note that Theorem 2.5 is derived under the assumption that m' is small enough with respect to the normalization factor in Eq. 2.4-7. If the initial rate is very large ($r_i \in \Omega(\int_0^1 \xi^{-\alpha/2}(1-\xi)^{-\beta}\mathrm{d}\xi)$), the gain rate has a diminishing return effect with respect to the initial rate. This is because the total links in the network are limited: A node cannot attract more than $2mt$ links, no matter how many additional initial links are given. Therefore, the relationship between r_g and r_i has two stages as shown in Fig. 2.3 (denoted as the linear stage and the diminishing return stage).

The second property revealed by Eq. 2.4-13 is that the initial links are most impactful when the attractiveness brought by the node degree and the repulsiveness brought by the age difference cancel each other out. This is because the threshold of

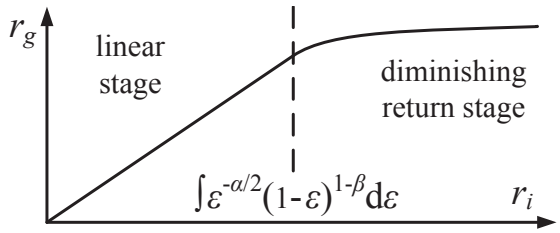


Figure 2.3: The two-stage relationship between the gain rate (r_g) and the initial rate (r_i), when $|(1-\alpha)C(\alpha,\beta,\xi)| \ll 1$.

$|(1-\alpha)C(\alpha,\beta,\xi)|$ will be small in such a case. The first node will not monopolize the majority of the links, while the aging effect will not prevent the new entering node from connecting to old nodes. The links from new entering nodes in the growing network will evenly connect to both the old and the young nodes. Actually, the classic model with $\alpha = 1$ and $\beta = 0$ is such a case (as previously shown in Eq. 2.4-5), where the gain rate is strictly linear with respect to the initial rate. This is because the threshold of $|(1-\alpha)C(\alpha,\beta,\xi)|$ becomes zero for $\alpha = 1$ and $\beta = 0$. Moreover, this threshold can be used to estimate whether the initial links are impactful in the corresponding growing network or not. The initial links are most impactful, when the corresponding growing network lies in the "middle" of the transition area.

The third property revealed by Eq. 2.4-13 is on the impact of the time period. Note that $-C(\alpha,\beta,\xi)$ decreases monotonously with respect to ξ ($\xi = s/t$). Meanwhile, the slope of the linear stage is approximately $e^{-C(\alpha,\beta,\xi)}$ (a smaller ξ brings a larger slope). The insight is that the initial links become more impactful with respect to a longer time period (the snowballing effect becomes more significant). Further explorations on the relationship between $C(\alpha,\beta,\xi)$ and ξ (representing the impact of the time) will be our future work.

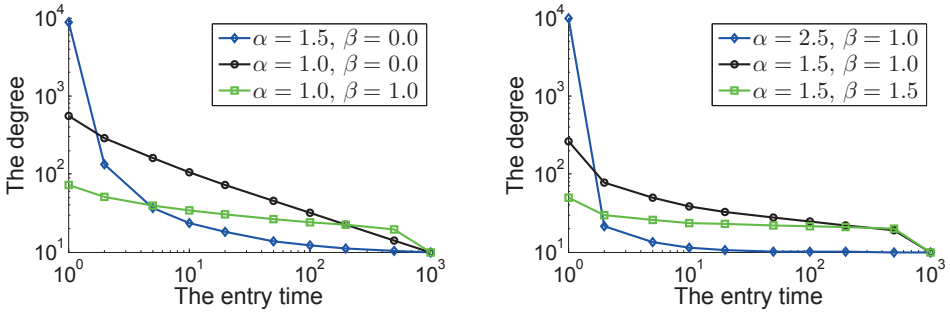
2.5 Experiments

This section first sets up the age-sensitive preferential attachment model to verify the accuracies of our theoretical results. Then, the snowballing effects are studied in several real-network datasets. The experimental results are shown from different perspectives to provide insightful conclusions.

2.5.1 Accuracy Verifications on Theoretical Results

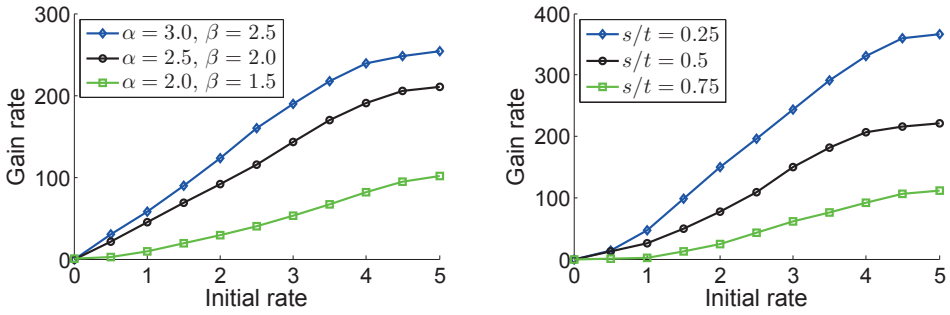
This subsection verifies the accuracy of our theoretical results for the age-sensitive preferential attachment model, which has a duration of 1,000 time slots (i.e., t =1,000). Upon each time slot, one new node will enter the network with 10 new connections to the existing nodes (i.e., $m = 10$). First, we check the expected node degree distributions with respect to the node entry time s. The results, which are averaged over 1,000 times, are shown in Fig. 2.4 as a log-log plot. Figs. 2.4(a) and 2.4(b) show two scenarios with different values of α and β. It can be seen that, when $\alpha = \beta + 1.5$, the first node attracts almost all the new links from the later nodes (there are $m \times t$ =10,000 links in total). At this time, the expected node degree decays quickly with respect to the node entry time, due to the overpowered attractiveness brought by the node degree. On the other hand, when $\alpha = \beta$, all the nodes tend to have a degree of $O(m)$, since the overpowered repulsiveness brought by the age difference only enables new nodes to connect to the most recent nodes. It can also be seen that, when neither the node degree nor the age difference dominates the attachment probability ($\beta \leq \alpha \leq \beta + 1.5$), the resulting network structure is more complex, due to the fact that the new nodes will connect to both old and young existing nodes. The experiment verifies the existence of the percolation phenomena in the age-sensitive preferential attachment model.

The snowballing effects in the transition area are also experimentally studied.



(a) Scenario 1. (b) Scenario 2.

Figure 2.4: The node degree distributions with respect to the node entry time.



(a) $s/t = 0.5$. (b) $\alpha = 2.5, \beta = 2.0$.

Figure 2.5: The node degree snowballing effects.

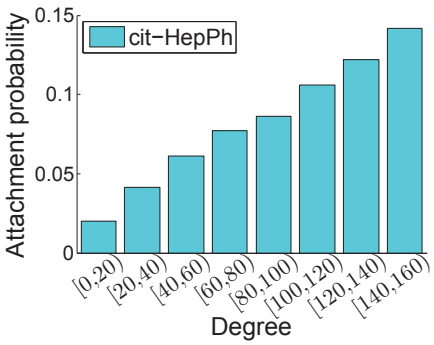
Fig. 2.5(a) shows the relationship between the initial rate and the gain rate for the node that enters the network at the 500^{th} time slot (i.e., $s/t = 0.5$), under three different settings of α and β. Each of the curves in Fig. 2.5(a) clearly has two stages (linear stage and diminishing return stage) as previously analyzed. Then, Fig. 2.5(b) shows the impact of the time, under $\alpha = 2.5$ and $\beta = 2.0$. It can be seen that a smaller s/t leads to a larger gain rate, since more nodes will enter the network after the node N_s. These experimental results confirm that our theoretical results in Eq. 2.4-13 are accurate. In the following two subsections, we will further verify the node degree snowballing effects in real data-driven experiments (the citation network and the online social network).

2.5.2 Citation Network

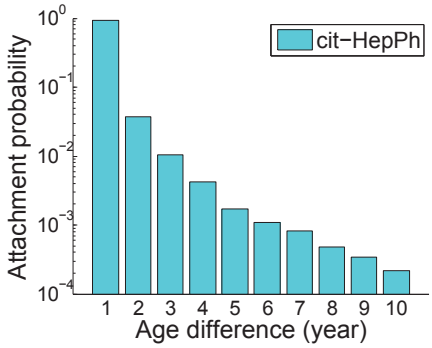
This subsection conducts real data-driven experiments to verify the snowballing effects in citation networks. Citation networks are classic growing networks, where papers serve as nodes in the network. New papers enter the network as time goes by. If a paper i cites a paper j, then the network contains a directed link from i to j. It is common sense that authors generally prefer to cite papers with a high number of citations (compared to papers with a low number of citations), as well as recent papers (compared to old papers). Currently, it is well-known that the classic preferential attachment model can explain the formation of citation networks [7].

Our experiments use the real dataset [24] of the Arxiv high energy physics phenomenology citation network (denoted as cit-HepPh). This dataset covers the papers published over the period from January 1993 to April 2003. 34,546 papers (nodes) and 421,578 citations (links) are involved in this dataset. On average, a new paper enters the growing network every 0.12 days (i.e., the time unit for adding one new node to the growing network). The relationships between the attachment probability and the node degree (or age difference) are shown in Fig. 2.6. It verifies the feasibility of the assumption that the attachment probability is proportional to $d^{\alpha} \cdot \Delta t^{-\beta}$. Then, we use the maximum likelihood to estimate the exponents α and β in the attachment probability. On average, we get $\alpha = 0.91$ and $\beta = 1.2 \times 10^{-3}$ as the exponents in the attachment probability. β is small, due to the ground truth that we sometimes cite a paper from 10 years ago (more than 30,000 time units).

To study the snowballing effects, papers published in the years 1995 and 1998 are analyzed. If a paper has more than an average number of citations in its first publication year, the additional portion is regarded as its initial rate. The final number of citations of that paper (in the year 2003) and the average number of citations are used to calculate the gain rate of that paper. The result is shown in

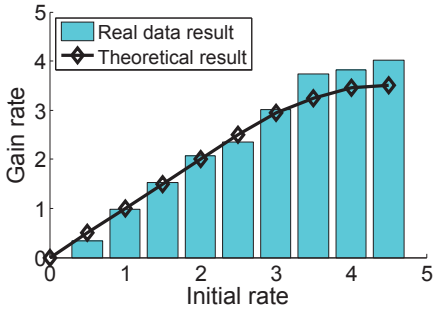


(a) The relationship between attachment probability and node degree.

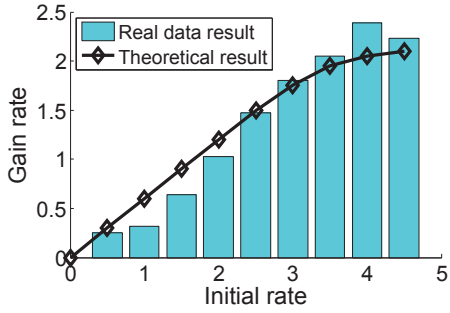


(b) The relationship between attachment probability and age difference.

Figure 2.6: The age-sensitive preferential attachment in cit-HepPh.



(a) Papers published in 1995.



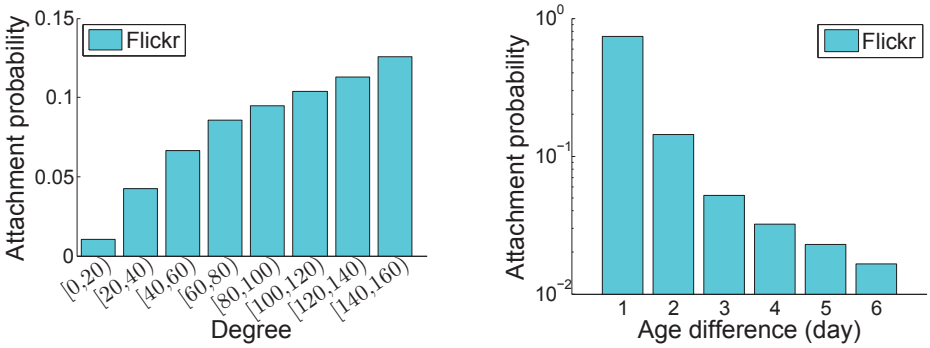
(b) Papers published in 1998.

Figure 2.7: The node degree snowballing effects in cit-HepPh.

Fig. 2.7, where we have $\alpha = 0.91$ and $\beta = 1.2 \times 10^{-3}$. It can be seen that the relationship between the initial rate and the gain rate has two stages in Fig. 2.7 (i.e., real data result). It is consistent with Eq. 2.4-13, which is denoted as the theoretical result in Fig. 2.7. It can also be seen that, the initial links are more impactful for earlier papers. For the same initial rates, papers published in 1995 have higher gain rates than did those in 1998.

2.5.3 Online Social Network

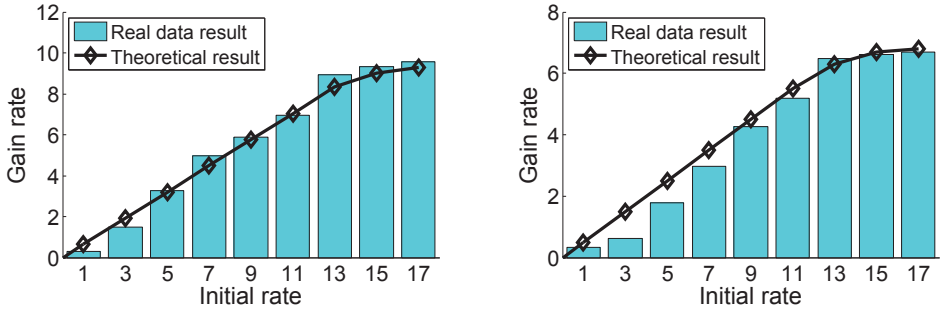
This subsection conducts real data-driven experiments to verify the snowballing effects in the online social networks, which are platforms for users to build social



(a) The relationship between attachment probability and node degree.

(b) The relationship between attachment probability and age difference.

Figure 2.8: The age-sensitive preferential attachment in Flickr.



(a) Users entered on Monday.

(b) Users entered on Wednesday.

Figure 2.9: The node degree snowballing effects in Flickr.

relations among other users. Users in the network share news, stories, and photos with each other. Social network sites are web-based services that allow individuals to create a public profile, to create a list of users with whom to share connections, and view and cross the connections within the system. Online social networks are growing networks, where users (i.e. nodes) enter the network one by one. If a new user i follows an existing user j, then the network contains a directed link from i to j. Users are more likely to follow popular users with high degrees, as well as contemporary users with smaller age differences. It is well-known that the classic preferential attachment model can also be applied to the online social networks [7].

In the experiments, we use the Flickr dataset [25]. Flickr is an online social network

for sharing photos. Key features of Flickr not initially present are tags, marking photos as favorites, group photo pools, and interestingness, for which a patent is pending. This dataset covers all new users in the period from November 2006 to May 2007, including 167,527 users (nodes) and 526,874 follower-followee relationships (links). On average, a new user enters the growing network every 0.04 days (i.e., the time unit for adding one new node to the growing network). The relationships between the attachment probability and the node degree (or age difference) are shown in Fig. 2.8. It again verifies the feasibility of the assumption that the attachment probability is proportional to $d^{\alpha} \cdot \Delta t^{-\beta}$. We also use the maximum likelihood to estimate α and β. On average, it turns out that we have $\alpha = 0.89$ and $\beta = 1.3 \times 10^{-4}$ in this dataset. Note that β is also small in this dataset, due to the ground truth that new users sometimes follow old users (one month is about 750 time units).

To study the snowballing effects, we focus on the users who entered the network in the first week of April. These users are selected, since they have complete records in the dataset (while the records of some other users may be missing). If a user has above average connections in his/her first day, the additional portion is regarded as its initial rate. Meanwhile, the final number of connections of that user (at the end of this week) and the average number of connections are used to calculate the gain rate of that user. The result is shown in Fig. 2.9, where we have $\alpha = 0.89$ and $\beta = 1.3 \times 10^{-4}$. It can be seen that the relationship between the initial rate and the gain rate should also have two stages, as shown in Fig. 2.9 (i.e., real data result). Although our theoretical result in Eq. 2.4-13 has a little overestimation in Fig. 2.9(b), it is basically accurate for this dataset. The initial links are also more impactful for earlier users in this dataset. For the same initial rates, users who entered on Monday have higher gain rates than do those who entered on Wednesday.

2.6 Summary

This chapter studies the node degree snowballing effects in the age-sensitive preferential attachment model, where the attachment probability depends on both the node degree and the age difference. We are interested in accelerating such degree snowballing effects through the impact of the initial links. Our study answers the question 'how many additional citations can an author obtain through self-citations?' The percolation phenomena are found in the age-sensitive preferential attachment model: the initial links are only impactful in the transition area, where neither the node degree nor the age difference dominates the attachment probability. In that case, we show that the relationship between the initial rate and the gain rate has two stages (linear stage and diminishing return stage). Real data-driven experiments in the citation network and the online social network verify the accuracies of our theoretical results, which cast some new light on the impact of the initial links in real-world growing networks.

2.7 Appendix

2.7.1 Proof of Theorem 2.1

The basic idea of the proof is that, when the node N_{t+1} enters the network, its probability of linking to the node N_1 is definitely smaller than that to the node N_t. The key observation is that N_1 has at most mt links at the time t (i.e., it attracts all the links of the later nodes). While the linking probability from N_{t+1} to N_t is proportional to $d^\alpha \cdot \Delta t^{-\beta} = m^\alpha$, the linking probability from N_{t+1} to N_1 is asymptotically bounded by $d^\alpha \cdot \Delta t^{-\beta} = (mt)^\alpha \cdot t^{-\beta} = m^\alpha \cdot t^{\alpha-\beta}$. If $\alpha < \beta$, then N_{t+1} is much more likely to link to N_t, instead of N_1 ($m^\alpha \gg m^\alpha \cdot t^{\alpha-\beta}$ when t is large). This implies that N_1 is no longer able to attract new links. N_1 attracts a finite number of links.

2.7.2 Proof of Theorem 2.2

The basic idea of the proof is that, when $\alpha < \beta$, the node N_s needs $\Omega((t-s)^{\beta/\alpha})$ additional links to resist the dominated repulsiveness brought by the age difference. The key observation is that the node N_s can attract more links if we ignore the existences of all the nodes older than N_s. In other words, the upper bound for the degree of the node N_s is the case, where it is regarded as the first node in the growing network. Similar to the proof of Theorem 2.1, let us focus on the attachment probability for the node that enters the growing network at the time t. While the linking probability from N_t to N_{t-1} is proportional to $d^\alpha \cdot \Delta t^{-\beta} = m^\alpha$, the linking probability from N_t to N_s is at most proportional to $d^\alpha \cdot \Delta t^{-\beta} = d^\alpha \cdot (t-s)^{-\beta}$. To keep the attractiveness of the node N_s, its degree should be larger than $m(t-s)^{\beta/\alpha}$, which can only be brought by its additional initial links. Therefore, at least $\Omega((t-s)^{\beta/\alpha})$ additional links are needed.

2.7.3 Proof of Theorem 2.3

By induction, we now prove that, when $\alpha > \beta + 1.5$, the node N_1 has a degree of at least $c \cdot t$ at the time t. Here, c is a certain constant. This declaration is true, when $t = 1$. Suppose this declaration holds when $t = T$, and then the node N_{T+1} enters the growing network. Note that, the linking probability from N_{T+1} to N_1 is proportional to $d^\alpha \cdot \Delta t^{-\beta} \geq c^\alpha \cdot T^{\alpha-\beta}$. Meanwhile, the linking probability from N_{T+1} to all the other nodes (i.e., N_2, N_3, ..., N_T) is at most proportional to $(2m-c)^\alpha \cdot \int_1^{T-1} \Delta t^{-\beta} \mathrm{d}\Delta t$. This upper bound is obtained by using the average degree of $(2m-c)$ to approximate this linking probability, since older nodes should have larger degrees than younger nodes. When $\beta \geq 0$, we have $\int_1^{T-1} \Delta t^{-\beta} \mathrm{d}\Delta t < T$. Therefore, the condition of $\alpha > \beta + 1.5$ indicates $T^{\alpha-\beta} \gg T$, meaning the node N_1 attracts the most links of the node N_{T+1}. If we set $c \leq m/2$, then the node N_1 has a degree of at least $c \cdot (T+1)$, when $t = T+1$.

By induction, the node N_1 has a degree of at least $c \cdot t$ at the time t, when $\alpha > \beta + 1.5$.

2.7.4 Proof of Theorem 2.4

When $\alpha > \beta + 1.5$, Theorem 2.3 states that the first node N_1 has a degree of $\Theta(s)$, when the node N_s enters the growing network at the time s. Since N_1 has a very large degree, N_s needs some additional initial links to compete with N_1 in the following attachment process. Let us consider the case when the node N_{s+1} enters the network at the time $s+1$. The linking probability from N_{s+1} to N_1 is proportional to $d^{\alpha} \cdot \Delta t^{-\beta} \in \Theta(s^{\alpha-\beta})$. Therefore, the node N_s needs at least $\Omega(s^{\alpha-\beta})$ additional initial links to break the link monopoly of N_1. Otherwise, N_s cannot attract the new links.

2.7.5 Proof of Theorem 2.5

When $|(1-\alpha)C(\alpha,\beta,\xi)| \gg 1$, Eq. 2.4-13 can be rewritten as:

$$\begin{aligned} r_g &= \left[\frac{(1+r_i)^{1-\alpha} + (1-\alpha)C(\alpha,\beta,\xi)}{1+(1-\alpha)C(\alpha,\beta,\xi)}\right]^{\frac{1}{1-\alpha}} - 1 \\ &\approx \left[\frac{(1-\alpha)C(\alpha,\beta,\xi)}{(1-\alpha)C(\alpha,\beta,\xi)}\right]^{\frac{1}{1-\alpha}} - 1 = 1 - 1 = 0 \end{aligned} \tag{2.7-14}$$

When $|(1-\alpha)C(\alpha,\beta,\xi)| \ll 1$, Eq. 2.4-13 can be rewritten as:

$$\begin{aligned} r_g &= \left[\frac{(1+r_i)^{1-\alpha} + (1-\alpha)C(\alpha,\beta,\xi)}{1+(1-\alpha)C(\alpha,\beta,\xi)}\right]^{\frac{1}{1-\alpha}} - 1 \approx \left[\frac{(1+r_i)^{1-\alpha}}{1+(1-\alpha)C(\alpha,\beta,\xi)}\right]^{\frac{1}{1-\alpha}} - 1 \\ &\approx \left[\frac{(1+r_i)^{1-\alpha}}{e^{(1-\alpha)C(\alpha,\beta,\xi)}}\right]^{\frac{1}{1-\alpha}} - 1 = (1+r_i)e^{-C(\alpha,\beta,\xi)} - 1 \end{aligned} \tag{2.7-15}$$

The above two equations complete the proof of Theorem 2.5.

CHAPTER 3

NSFA: NESTED SCALE-FREE ARCHITECTURE FOR SCALABLE PUBLISH/SUBSCRIBE OVER P2P NETWORKS

Chapters 2 and 3 focus on the social network architectures. More specifically, this chapter proposes a publish/subscribe system based on unstructured P2P networks, which are used for content distributions in online social networks. The network is shown to have Nested Scale-Free Architectures (NSFAs). The scale-free architecture is a classic concept, which means that the peer degree distribution follows power-law. 'Nested' indicates that the scale-free architecture is preserved when low-degree peers and their associated connections are removed. We find that NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies. By leveraging the NSFA's hierarchy, our publish/subscribe system achieves a competitive tradeoff among the event routing efficiency, system robustness, and overhead. For an unstructured P2P network with $|V|$ peers, the number of routing hops for the event deliveries in our system is expected to be $O(\ln \ln |V|)$. For the topological information, each peer only needs to maintain an overhead with a constant size, $O(1)$. Peer arrival, departure, and failure can be handled within a message complexity of $O(\ln \ln |V|)$. Finally, real data-driven experiments demonstrate the efficiency and effectiveness of the NSFA-based publish/subscribe system.

3.1 Introduction

Publish/Subscribe (pub/sub) systems are appealing abstractions for the Content Delivery Networks (CDNs). A pub/sub system involves three roles: *subscribers*, *publishers*, and *brokers*. Subscribers express their interests through *subscriptions* with the system, in order to receive *events* matching their subscriptions. Events are issued by publishers and are delivered to subscribers via brokers. Real-world pub/sub system applications include Yahoo Message Broker [26] which is integrated with web applications, Global Data Synchronization Network [27] that exchanges supply chain information among retailers, and SuperMontage [28] that disseminates financial data and orders among traders.

Most existing large-scale pub/sub systems are implemented as overlay networks on the Internet. They require hundreds of servers placed at strategic points across the globe to handle the load, as well as trained professionals to monitor these servers. For example, the infrastructure of Akamai's pub/sub system includes 56,000 dedicated servers among 950 networks in 70 countries [28]. However, such expensive infrastructure costs are not always feasible for many large corporations. Moreover, the scalabilities of these systems are inherently questionable, because all the events are handled on given servers in a centralized manner. As a result, building pub/sub systems on Peer-to-Peer (P2P) networks becomes an alternative option [29]. P2P networks are inexpensive and highly scalable, since all the peers contribute their machines to distributedly increase the computational and storage resources. In P2P-based pub/sub systems, peers are used to not only store events, but also route events to other peers with matching subscriptions.

P2P networks can be roughly classified as *unstructured* and *structured*. Unstructured P2P networks, such as Gnutella, Kazaa, and Bitcoin, are formed by peers that "randomly" connect to each other. Since there is not a globally-

(a) P2P network with all peers. (b) Top 50% peers.

Figure 3.1: NSFA in the Gnutella dataset.

imposed structure on peers, unstructured P2P-based pub/sub systems have very low construction overheads and are highly robust in terms of frequent churns (peer arrivals and departures). However, routing events from publishers to subscribers becomes extremely inefficient due to the lack of structure. For example, Sub-2-Sub [30] uses a flooding-based routing strategy that leads to a high amount of network traffic. For another example, SIENA [31] cannot provide a bounded routing performance, in terms of the routing hops and overheads. On the other hand, structured P2P networks, such as Chord, P-Grid, and Pastry, organize peers into specific topologies through Distributed Hash Tables (DHTs), and thus enable high-performance routings for pub/sub systems. However, as a tradeoff, structured P2P-based pub/sub systems have considerable construction overheads and are vulnerable to churns, due to the maintenance of large-size DHTs. For example, handling one peer churn in Terpstra [32] and PastryStrings [33] takes logarithmic messages with respect to the number of peers in the system.

Recent advances in network science show that peer connections in unstructured P2P networks are not truly random [34]. We propose that they share a *Nested Scale-Free Architecture* (NSFA). The scale-free architecture is a classic concept [34],

meaning that the peer degree distribution follows power-law. In such an architecture, a majority of the periphery peers are inactive with a small number of connections, while a minority of the core peers are active with a large number of connections. 'Nested' indicates that the scale-free architecture is preserved, when low-degree peers and their associated connections are removed. An example is shown in Fig. 3.1, where peers with more connections are shown as larger nodes. Fig. 3.1(a) depicts the largest strongly connected component formed by peers whose IDs are smaller than 500 in the Gnutella dataset [35]. It is scale-free. Then, we iteratively remove the peer with the smallest number of connections, until half of the peers remain. Fig. 3.1(b) shows the resulting network, which maintains scale-free by the nested property. We find that NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies [36, 37, 38, 39, 40]. For an unstructured P2P network with $|V|$ peers, the number of hierarchical levels in NSFA can be bounded by $\Theta(\ln|V|)$.

By leveraging the NSFA's hierarchy, this chapter proposes a novel distributed pub/sub system for unstructured P2P networks. While our system has very low construction overhead and is highly robust, its event routing efficiency is competitive with those in structured P2P-based pub/sub systems. Our event routing is hierarchical, and has two phases: the publisher first uploads the event to the network core, which in turn downloads the event to the subscriber. For an unstructured P2P network with $|V|$ peers, the number of routing hops can be $O(\ln\ln|V|)$ in our approach. Moreover, instead of using costly DHTs whose sizes scale up with the number of peers, each peer in the proposed system only needs to keep a constant-size overhead as the topological information to handle event routings and peer churns. Peer arrival, departure, and failure can be handled within a message complexity of $O(\ln\ln|V|)$. Our pub/sub system outperforms classic systems, in terms of better bounds of asymptotic performances.

Our main contributions are summarized as follows:

- We address a novel architecture of NSFA in unstructured P2P networks. A distributed labeling scheme is proposed to determine the hierarchical levels in NSFA.
- Based on NSFA's hierarchy, we propose a novel pub/sub system, which has low construction overhead, is highly robust in terms of peer churns, and is efficient for event routings. Our system performance is well-bounded.
- The state-of-the-art pub/sub systems over P2P networks are surveyed in a comparative manner with the proposed work, in terms of the event routing efficiency, system robustness, and overhead.
- Extensive real data-driven experiments on Gnutella and Bitcoin are conducted to evaluate the proposed pub/sub system. Experimental results are shown from different perspectives to provide insightful conclusions.

The remainder of this chapter is organized as follows. Section 3.2 formulates the problem. Section 3.3 presents the NSFA and its hierarchy. Section 3.4 describes the pub/sub system. Section 3.5 surveys the related works in a comparative manner with the proposed pub/sub system. Section 3.6 includes experiments. Finally, Section 3.7 concludes this chapter.

3.2 Problem Statement

The objective of this chapter is to build an advanced content-based pub/sub system for unstructured P2P networks. An unstructured P2P network is modeled by a directed graph $G = (V, E)$, where V is a set of peers (nodes), and $E \subseteq V^2$ is a set of directed connections among the peers (directed edges). The numbers of incoming and outgoing connections held by a peer are denoted as its indegree and outdegree, respectively. Each peer may publish events, which are conjunctions of attribute-value

pairs. For example, {(temperature, 30), (precipitation, 20)} is an event describing the weather conditions. Each peer has its own interests. Peers express their interests through subscriptions with the system, in order to receive interested events matching their subscriptions. Subscriptions are conjunctions of attribute-operator-value tuples. For example, {(temperature, $<$, 40), (precipitation, $>$, 10)} could be one subscription of a peer. A peer may have multiple subscriptions. A subscription matches an event if all the operators in this subscription are satisfied using the corresponding attributes and values of that event. Peers can also route events to other peers. Each peer can be a publisher, a subscriber, a broker, or an arbitrary combination of these three roles.

To evaluate the system performance, we focus on three metrics: *event routing efficiency*, *robustness*, and *overhead*. Event routing efficiency refers to the number of routing hops in event deliveries. Robustness refers to the number of messages used by the system to deal with peer arrivals, departures, and failures. Overhead refers to each peer's storage consumption on the topological information for event routings, peer churns, and peer failures. Classic unstructured P2P-based pub/sub systems are highly robust and have low overheads, but sacrifice routing efficiency due to the lack of structure. On the contrary, classic structured P2P-based pub/sub systems have a high routing efficiency, but perform poorly in terms of robustness and overhead. This is because they need to maintain large-size DHTs as the topological information. By contrast, our system obtains well-bounded results on all these three metrics by leveraging the NSFA of unstructured P2P networks.

We use a hierarchical approach. We start with the inherent hierarchy in unstructured P2P networks (the NSFA's hierachy in Section 3.3). The hierarchical level of each peer is determined. Based on the NSFA's hierarchy, our pub/sub system is described (Section 3.4). Then, a comparative survey of related works is given on their asymptotic performances (Section 3.5). Our system has a better bound than classic approaches.

3.3 Unstructured P2P Networks

3.3.1 Scale-Free Architecture

A P2P network is a distributed application that partitions tasks or work loads between peers. They can be roughly classified as unstructured (Gnutella, Kazaa, and Bitcoin) and structured (Chord, P-Grid, and Pastry), depending on whether particular topologies are specified or not by design. In previous decades, peers in unstructured P2P networks were considered to be "randomly" connected to each other. However, recent advances in network science show that peer connections are not truly random [34]. Bulut et al. [4] showed that unstructured P2P networks have the Scale-Free Architecture (SFA):

Definition 3.1. A network (i.e., G) satisfies SFA, if its node degree distribution follows power-law.

Since P2P networks are directional, the degree could be either indegree or outdegree. Let P_d denote the fraction of peers with a degree of d, the power-law means that:

$$P_d = \frac{\alpha - 1}{d_{\min}} \cdot \left(\frac{d}{d_{\min}}\right)^{-\alpha} \tag{3.3-1}$$

Here, α is a constant parameter ranging from 2 to 3 [34]. $d_{\min}$ is also a constant parameter from where the power-law degree distribution holds. Eq. 3.3-1 indicates that majority peers hold only a few connections, while minority peers hold lots of connections. Consequently, SFA can produce the following network hierarchy by degree rankings:

Definition 3.2. SFA's network hierarchy is defined by ranking hierarchical levels based on peer degrees. Peers with smaller degrees have lower hierarchical levels.

Algorithm 1 is proposed as a distributed labeling scheme that determines peers'

Algorithm 1 Distributed labeling scheme for SFA

Input: The peer v and its neighbors.
Output: The hierarchical level of v in SFA.

1: Initialize v as unlabeled.
2: **repeat** in a round-by-round manner **do**
3: **if** v has the smallest degree (including the tie) among all its unlabeled neighbors **then**
4: Set v's label to be the largest label among its labeled neighbors plus one (in the event that v does not have a labeled neighbor, v's label is one).
5: **return** v's label as its hierarchical level.

hierarchical levels in SFA. It works through synchronous peer iterations (a round-by-round manner in line 2). The label of a peer indicates its hierarchical level. Since unstructured P2P networks are directional, Algorithm 1 has an *indegree version* and an *outdegree version*, depending on the degree used in it. For indegree and outdegree versions, the neighbors in Algorithm 1 (lines 3 and 4) refer to the incoming and outgoing neighbors, respectively.

SFA's hierarchy has been widely acknowledged in many existing works [36, 37, 38, 39, 40]. One of the most famous example includes BubbleRap [37], which presents a routing scheme for delay tolerant networks with social features. A message in BubbleRap is iteratively forwarded to nodes with larger degrees to seek the message destination. However, the number of routing hops in BubbleRap is not well-bounded even if the network satisfies SFA. The number of hierarchical levels in SFA could be $\Theta(|V|)$, where $|V|$ is the number of nodes in the network [40]. As a result, hierarchical routings in SFA may perform poorly under certain scenarios. On the other hand, SFA is not sufficient to describe the ad-hoc nature of unstructured P2P networks, where peers arrive and depart dynamically. Further explorations are conducted.

Algorithm 2 Distributed labeling scheme for NSFA

Input: The peer v and its neighbors.

Output: The hierarchical level of v in NSFA.

Same as Algorithm 1, except the subtle change on the if statement in line 3: **if** v has the smallest *effective degree* (including the tie) among all its unlabeled neighbors

3.3.2 Nested Scale-Free Architecture

Differing from structured P2P networks that have specified topologies by design, unstructured P2P networks are formed by peers that arrive and depart dynamically. When new peers arrive, they connect to existing peers as new peripheries of the existing network. Such time-evolving peer dynamics create an *onion-like architecture*, which is defined as the NSFA:

Definition 3.3. Let G_s denote the set of subgraphs generated by iteratively removing the lowest-degree node and its connections in a network, G. We claim that G satisfies NSFA, if (i) G and all the subgraphs in G_s satisfy SFA, and (ii) the standard deviation of their power-law exponents, α, is $o(1)$.

An intuitive explanation of NSFA is that, G satisfies NSFA, if it satisfies SFA and its power-law exponent (i.e., α in Eq. 3.3-1) has a limited variation when the current lowest-degree node in G is *iteratively* removed. Subgraphs in G_s, which are overly small to depict SFA, can be ignored by setting up a threshold. Clearly, NSFA is a "nested" extension of SFA. We claim that unstructured P2P networks satisfy NSFA (verified later). NSFA is our novel contribution, and is the key idea of this chapter. NSFA produces the following network hierarchy:

Definition 3.4. NSFA's network hierarchy is defined by ranking hierarchical levels by iteratively removing the set of peers that have the lowest degrees among their neighbors. Peers that are removed earlier have lower hierarchical levels.

The difference between Definitions 3.3 and 3.4 is that NSFA's hierarchy is defined by removing all the *local* lowest-degree peers in one iteration, while NSFA is defined

by removing the *global* lowest-degree peer in one iteration. Clearly, through local approximations, NSFA's hierarchy can reveal the structural properties of networks that satisfy NSFA. Algorithm 2 is proposed as a distributed labeling scheme to determine peers' hierarchical levels in NSFA. It involves the following concept: the *effective degree* of a peer is defined as its number of connections to its unlabeled neighbors. Algorithm 2 also works through synchronous peer iterations (round-by-round manner in line 2). The label of a peer indicates its hierarchical level. The only difference between Algorithms 1 and 2 is line 3 (the if statement), which shows the prerequisite for the peer labeling. However, it leads to a fundamental insight difference: Algorithm 2 can reveal the nested property, while Algorithm 1 cannot reveal this property. This is because labeling peers in a round of Algorithm 2 is essentially peeling off the current outermost layer of the onion-like network. Once a peer labels itself, its connections are no longer counted in the effective degrees of unlabeled peers. Peers are iteratively peeled off by the labeling process to reveal the onion-like architecture.

To better illustrate the differences between Algorithms 1 and 2, an example of their indegree versions is shown in Fig. 3.2. A node represents a peer. The numbers within the nodes are their hierarchical levels. Directed arrows are directed connections among peers. In Fig. 3.2, the two peers with the largest indegrees have the maximal hierarchical level in Algorithm 1, but they do not have the maximal hierarchical level in Algorithm 2. This is because their connections are peeled off in the first round. Each round of Algorithm 2 peels off the outermost layer of the onion-like network, which is composed of the remaining unlabeled peers. Our next claim is that, as the most important property and the greatest advantage of NSFA, the expected number of rounds for Algorithm 2 to terminate is bounded:

Theorem 3.5. *Suppose an unstructured P2P network has $|V|$ peers and satisfies NSFA. Then, Algorithm 2 is expected to terminate within $\Theta(\ln|V|)$ rounds of*

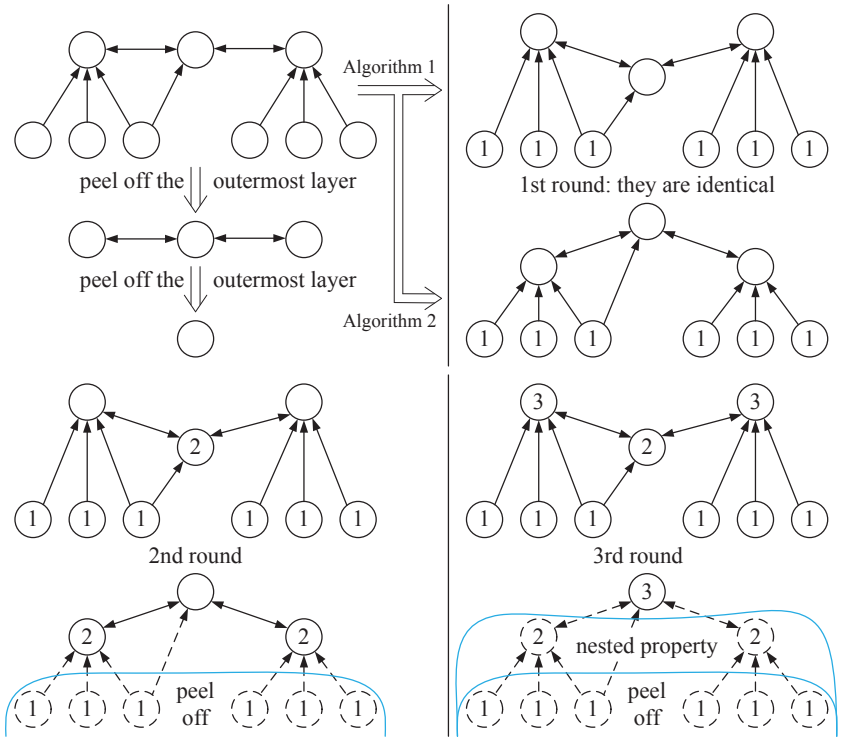


Figure 3.2: Difference between Algorithms 1 and 2 (indegree version).

synchronous peer iterations. The maximal peer label is also $\Theta(\ln |V|)$.

Proof: The key idea is to show that a constant percentage of unlabeled peers are expected to label themselves in each round of Algorithm 2. We start with the upper bound. Initially, all the peers are unlabeled. Let us consider the probability that an arbitrary peer (say v) labels itself. Suppose v's degree is d_v. Based on Eq. 3.3-1, the probability that a neighbor of v has a higher degree than v is:

$$\int_{d_v}^{\infty} \frac{\alpha - 1}{d_{\min}} \cdot \left(\frac{d}{d_{\min}}\right)^{-\alpha} \mathrm{d}d = \left(\frac{d_v}{d_{\min}}\right)^{1-\alpha} \tag{3.3-2}$$

The node v labels itself when all of its d_v neighbors have larger effective degrees than v. Since all peers are initially unlabeled, the probability that v labels itself is $(d_v/d_{\min})^{(1-\alpha)\times d_v}$. Therefore, the expected percentage of peers that will label

themselves in the first round of Algorithm 2 is:

$$\begin{aligned}
&\int_{d_{\min}}^{\infty} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)\times d_v} \cdot \frac{\alpha-1}{d_{\min}} \cdot \Big(\frac{d_v}{d_{\min}}\Big)^{-\alpha} \mathrm{d}d_v \\
&> \int_{d_{\min}}^{2d_{\min}} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)\times d_v} \cdot \frac{\alpha-1}{d_{\min}} \cdot \Big(\frac{d_v}{d_{\min}}\Big)^{-\alpha} \mathrm{d}d_v \\
&> \int_{d_{\min}}^{2d_{\min}} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)\times 2d_{\min}} \cdot \frac{\alpha-1}{d_{\min}} \cdot \Big(\frac{d_v}{d_{\min}}\Big)^{-\alpha} \mathrm{d}d_v \\
&= \frac{1}{2d_{\min}+1}\Big[1 - \frac{1}{2^{(\alpha-1)(2d_{\min}+1)}}\Big] = c
\end{aligned} \tag{3.3-3}$$

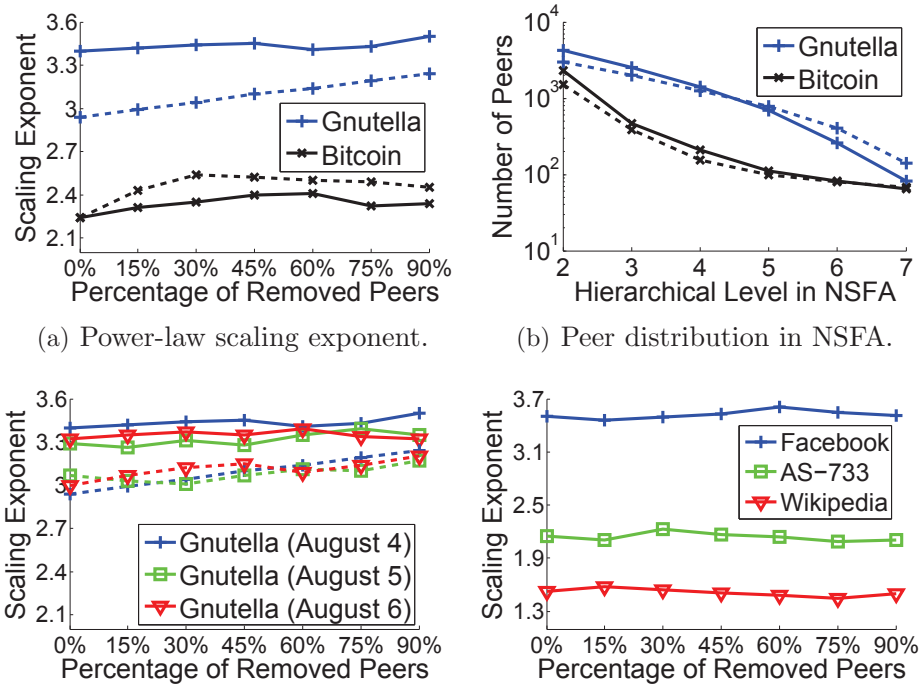
Let c denote the result in Eq. 3.3-3. Note that α is a constant that ranges from 2 to 3 [34]. $d_{\min}$ is also a constant parameter. Therefore, c is a positive constant, meaning that more than a constant percentage of unlabeled peers will label themselves in the first round of Algorithm 2. Then, in the second round, these labeled peers are peeled from the remaining network. This is because their connections are not counted in the effective degrees of unlabeled peers. With the nested property, we consider that the effective degree distribution for the unlabeled peers remains the same. This is because peers labeled in one iteration are not adjacent to each other, and the remaining network belongs to G_s by Definition 3.3. We ignore the limited variation of α. Through the same arguments in Eqs. 3.3-2 and 3.3-3, more than a constant percentage of unlabeled peers will label themselves in the second round of Algorithm 2. By induction, we conclude that *more than* a constant percentage of unlabeled peers will label themselves in each round of Algorithm 2. Since $|V| \times c^{-\log_c |V|} = 1$, Algorithm 2 is expected to terminate within $-\log_c |V|$ rounds. c is a positive constant that is smaller than one, and thus $-\log_c |V|$ belongs to $O(\ln |V|)$. Therefore, the number of rounds for Algorithm 2 to terminate is expected to be $O(\ln |V|)$. The proof of the lower bound is similar: *less than* a constant percentage of unlabeled peers will label

themselves in each round of Algorithm 2. Similar to Eq. 3.3-3, we have:

$$\begin{aligned}
&\int_{d_{\min}}^{\infty} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)\times d_v} \cdot \frac{\alpha-1}{d_{\min}} \cdot \Big(\frac{d_v}{d_{\min}}\Big)^{-\alpha} \mathrm{d}d_v \\
&< \frac{\alpha-1}{d_{\min}} \int_{d_{\min}}^{\infty} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)\times d_v} \mathrm{d}d_v \\
&= \frac{\alpha-1}{d_{\min}} \int_{d_{\min}}^{\infty} e^{(1-\alpha)\times d_v \times \ln(d_v/d_{\min})} \mathrm{d}d_v \\
&< \frac{\alpha-1}{d_{\min}} \Big[\int_{d_{\min}}^{ed_{\min}} \Big(\frac{d_v}{d_{\min}}\Big)^{(1-\alpha)d_{\min}} \mathrm{d}d_v + \int_{ed_{\min}}^{\infty} e^{(1-\alpha)d_v} \mathrm{d}d_v \Big] \\
&= \frac{\alpha-1}{d_{\min}} \Big[\frac{e^{(1-\alpha)d_{\min}+1}-1}{(1-\alpha)d_{\min}+1} d_{\min} - \frac{e^{(1-\alpha)ed_{\min}}}{1-\alpha} \Big] = c^* \qquad (3.3\text{-}4)
\end{aligned}$$

Let c^* denote the result in Eq. 3.3-4. Clearly, c^* is also a constant. Through the same argument, the number of rounds for Algorithm 2 to terminate is expected to be $\Omega(\ln|V|)$. Combining the upper and lower bounds, we conclude that Algorithm 2 is expected to terminate within $\Theta(\ln|V|)$ rounds of synchronous peer iterations. In each round, the maximal peer label increases by one (line 4 in Algorithm 1, which is also used by Algorithm 2). Therefore, the expected maximal peer label in NSFA is also $\Theta(\ln|V|)$. ■

Theorem 3.5 shows that *the number of hierarchical levels in NSFA can be logarithmically bounded*. By contrast, classic hierarchies ranked by the node connectivity [38, 40], social centrality [37, 39], or organizational roles [36] cannot guarantee such a bound. This is because they do not require structural similarities among different hierarchical layers [41]. Moreover, NSFA's hierarchy can be constructed distributedly with low overheads, since each node only takes local neighborhood information. Such great advantages of NSFA enable bounded high-performance routings. Furthermore, the number of routing hops in NSFA can be even asymptotically smaller than $\Theta(\ln|V|)$, since some hierarchical levels can be skipped. We will discuss this property later in Section 3.4 (Theorem 3.7).



(a) Power-law scaling exponent.

(b) Peer distribution in NSFA.

(c) Peer arrival and departure.

(d) Other NSFA networks.

Figure 3.3: NSFA verifications in unstructured P2P networks.

3.3.3 Verify the Existence of NSFA

To verify the existence of NSFA, real data-driven experiments on Gnutella [35] and Bitcoin [42] are conducted. The Gnutella dataset has 10,876 peers with 39,994 connections on August 4, 2002 (peer arrivals and departures are recorded for the following days). The Bitcoin dataset has 4,579 peers with 18,667 connections (partial dataset for comparisons with Gnutella). Verification results are shown in Fig. 3.3, where solid and dashed lines are the results of indegree and outdegree versions, respectively. Fig. 3.3(a) shows the variation of the power-law exponent (i.e., α), when the current lowest-degree peer is iteratively removed. α has a limited variation, even if a large percentage of peers are removed. Based on Definition 3.3, Gnutella and Bitcoin satisfy NSFA. Then, Fig. 3.3(b) shows the distribution of peers in NSFA's hierarchy. The number of peers decreases exponentially with respect to the hierarchical level. As a result, the number of hierarchical levels in NSFA is logarithmically bounded.

Table 3.1: Compare Hierarchies

Dataset	SFA indegree hierarchy		NSFA indegree hierarchy	
	# of levels	# of LMPs	# of levels	# of LMPs
Gnutella	20	1,715	10	120
Bitcoin	41	204	19	76
Dataset	SFA outdegree hierarchy		NSFA outdegree hierarchy	
	# of levels	# of LMPs	# of levels	# of LMPs
Gnutella	19	1,430	10	676
Bitcoin	36	167	17	43

Fig. 3.3(c) shows the scenario of peer arrivals and departures in Gnutella (three days from August 4 to 6). It can be seen that NSFA naturally holds when peers arrive and depart. The nested architecture actually results from peer arrivals and departures, in which SFA is satisfied. This phenomenon reveals that peer connections are not truly random [34]. Real data-driven experiments validate that unstructured P2P networks satisfy NSFA. Moreover, NSFA is not unique for unstructured P2P networks, i.e., NSFA exists in other types of networks. Experiments are conducted in three undirected networks [35]: Facebook (online social networks), AS-733 (autonomous systems), and Wikipedia dataset (website networks). Fig. 3.3(d) shows that these networks also satisfy NSFA, i.e., the variation of the power-law exponent is limited when the lowest-degree node is iteratively removed.

3.3.4 Compare Hierarchies

This subsection experimentally compares SFA's hierarchy and NSFA's hierarchy (Algorithms 1 and 2) in unstructured P2P networks. We have:

Definition 3.6. If a peer has a higher hierarchical level than all of its neighbors, it is a Local Maximum Peer (LMP).

We have two claims: (i) NSFA's hierarchy has fewer hierarchical levels than SFA's hierarchy, and (ii) NSFA's hierarchy in has fewer LMPs than SFA's hierarchy. The first claim means that the maximal peer label in Algorithm 2 is smaller than that

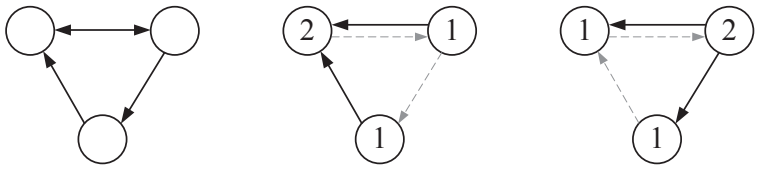
in Algorithm 1. Algorithm 2 terminates faster than Algorithm 1. The second claim means that NSFA's hierarchy is a more concentrated. To verify our claims, experiments on Gnutella [35] and Bitcoin [42] are conducted, based on the same settings as in the previous subsection. The result is shown in Table 3.1, where the NSFA's hierarchy has significantly fewer levels than SFA's hierarchy (basically half in both datasets). NSFA's hierarchy has many fewer LMPs than does SFA's hierarchy, especially in the indegree version (fewer than 10% for Gnutella). The above results demonstrate our claims.

NSFA's hierarchy has key advantages over SFA's hierarchy in terms of hierarchical event deliveries in pub/sub systems. This event delivery has two phases. In the first phase, a peer uploads its event to the network core, which is formed by the peers with the highest hierarchical levels. In the second phase, peers in the network core download the event to the subscribed peers. Since NSFA's hierarchy has fewer levels, events can be uploaded to the network core with fewer routing hops. Meanwhile, NSFA's hierarchy has fewer LMPs, meaning that there exist fewer local extrema for the event uploads and downloads. More details are described in the next section.

3.4 Publish/Subscribe in NSFA

3.4.1 System Design Overview

In a genenral pub/sub system, each peer may publish events and receive its own interested events. Each peer maintains an event filter to determine whether an event should be received or not. Each peer also needs some overhead to maintain the topological information for event routings, peer churns, and peer failures. In the proposed pub/sub system, such topological overheads of a peer only include its hierarchical level in both the indegree and outdegree versions of NSFA. Consequently, the storage consumption of a peer is a constant that does not scale up with the network



(a) Network. (b) Indegree. (c) Outdegree.

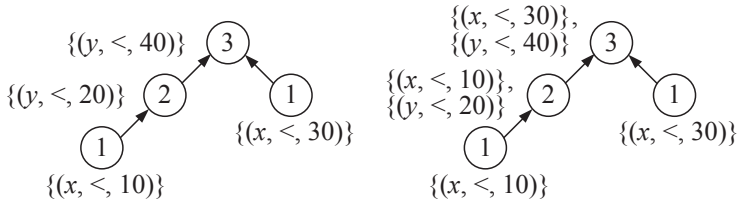
Figure 3.4: Examples of NSFA-based forests (indegree and outdegree versions).

size. In other words, the size of the overhead per peer is $O(1)$.

The event routing is accomplished through NSFA's hierarchy. This hierarchy abstracts *a forest of rooted trees* from the unstructured P2P network. The roots of these trees are LMPs. For a non-LMP peer, its parent in the tree is the neighbor that has the maximal hierarchical level. Since Algorithm 2 has two versions, the resultant forest has both indegree and outdegree versions. Fig. 3.4 shows such an example. Fig. 3.4(a) shows a network. The corresponding forests of indegree and outdegree versions are shown in Figs. 3.4(b) and 3.4(c), respectively. The numbers within the nodes represent their hierarchical levels in NSFA. Connections, which are not in the forest, are gray and dashed. We claim that the maximal tree depth in an NSFA-based forest can be bounded:

Theorem 3.7. *The maximal tree depth in an NSFA-based forest is expected to be* $O(\ln \ln |V|)$.

Proof: Let $l_{\max}$ denote the maximal peer label in NSFA. We start with an arbitrary peer (say v) in an arbitrary tree of the NSFA-based forest. Let l_v denote the label of this peer. Suppose the labels of v's neighbors are uniform-randomly distributed from 1 to $l_{\max}$. Then, for v's neighbors whose labels are larger than l_v, they are expected to have labels of $l_v + (l_{\max} - l_v)/2$. Since v's parent has the maximal label among v's neighbors, its label is expected to be larger than $l_v + (l_{\max} - l_v)/2$. In other words, the label difference between the current label and the maximal label is expected to be at least halved, when we move from an arbitrary peer to its parent. Since $2^{\log_2 l_{\max}} = l_{\max}$, it takes $O(\ln l_{\max})$ steps to move from a leaf to a root in a tree.



(a) Peer interests. (b) Subscriptions.

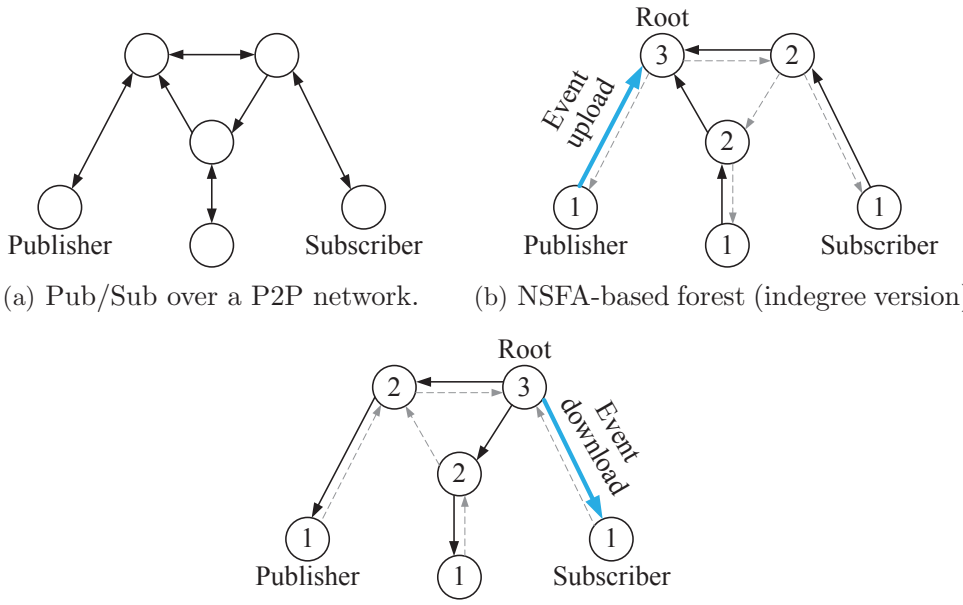
Figure 3.5: An example of peer interests and the corresponding subscriptions.

Since Theorem 3.5 has stated that $l_{\max} \in \Theta(\ln |V|)$, the maximal tree depth in an NSFA-based forest is expected to be $O(\ln \ln |V|)$. ■

The insight of Theorem 3.7 is that a peer may have a parent with a hierarchical level that is much higher than its own, leading to a double logarithmic tree depth. Theorems 3.5 and 3.7 show that connections in unstructured P2P networks are not truly random, in terms of NSFA. The maximal tree depth of an NSFA-based forest has a better bound than classic forests [43, 44]. Moreover, our NSFA-based forest can be constructed in a distributed manner. Our key idea is to utilize the NSFA-based forest as the network backbone to deliver events among peers with a bounded performance and a limited overhead.

3.4.2 Subscription

Peers express their interests through subscriptions with the system. Such subscriptions are captured by the event filter, which is maintained by each peer to determine whether an event should be received or not. The NSFA-based forest of the indegree version is used to collect these subscriptions. A peer may not only receive its own interests, but also may receive other peers' interests for the purpose of event deliveries. In our approach, a peer collects all the events interesting to itself, along with the interests of its descendants in the NSFA-based forest. Hence, a peer notifies all of its precedents in terms of its interests. An example is shown in Fig. 3.5, where x and y are two attributes. Fig. 3.5(a) shows the interests of each peer (attribute-



(a) Pub/Sub over a P2P network. (b) NSFA-based forest (indegree version).

(c) NSFA-based forest (outdegree version).

Figure 3.6: An illustration for the two-phase hierarchical event delivery in the proposed pub/sub system.

operator-value tuples). Fig. 3.5(b) shows the resulting subscriptions. The root will receive all the events matching that x is smaller than 30 or y is smaller than 40.

A peer subscription takes $O(\ln \ln |V|)$ messages. This is because this peer needs to notify all of its precedents, while the maximal tree depth is bounded by Theorem 3.7. The unsubscription process is similar and also takes $O(\ln \ln |V|)$. Note that peers with higher hierarchical levels tend to have more descendants, and thus they tend to collect more events. Hence, their event matchings are more time-consuming and can be accelerated by some existing works [45, 46].

3.4.3 Event Delivery

The proposed pub/sub system uses a two-phase hierarchical event delivery scheme. The first phase is the upload phase, which utilizes the NSFA-based forest of the indegree version. Once a peer wants to publish an event, it will upload this event to the root of its tree, through recursively forwarding this event to the parent. Once

the root receives an event in the upload phase, the second phase, or download phase, begins. This phase utilizes the NSFA-based forest of the outdegree version. Once a peer receives a matched event, it will forward this event to all of its children in the tree. This is because a peer collects all the events interesting to itself and its descendants through subscriptions. If the received event is not matched, the peer will drop it. In summary, the publisher first uploads the event to the network core (peers with highest hierarchical levels), which in turn downloads the event to the subscriber.

A potential problem is that multiple roots may exist in the NSFA-based forest. Meanwhile, the roots in the NSFA-based forest of the indegree version are not necessarily roots in the NSFA-based forest of the outdegree version. Hence, roots need to connect with each other before the event deliveries. Roots can register their addresses at an extra registration server. Since Table 3.1 shows that roots are few, the registration cost is ignorable. Through the registration server, roots can set up special connections to each other before the event deliveries. Similar techniques have been used in existing works [43, 44].

Fig. 3.6 illustrates an example for the event deliveries. The network topology is shown in Fig. 3.6(a). The publisher and subscriber are the peers who publish and subscribe to the event, respectively. Fig. 3.6(b) shows the upload event delivery phase in the NSFA-based forest of the indegree version. For simplicity, this forest includes only one tree. The numbers within the nodes (peers) are their hierarchical levels. Connections without the forest are gray and dashed. The event is uploaded to the root, by recursively being forwarded to the parent in the tree. Once the root receives the event, the upload phase terminates. The root sends this event to the other roots through special connections that are set up in advance of the event deliveries. Then, the download event delivery phase begins at the root in the NSFA-based forest of the outdegree version. As shown in Fig. 3.6(c), the peer will forward matched events to all of their children, but drop non-matched events.

Since Theorem 3.7 claims that the tree depth is $O(\ln \ln |V|)$, the number of routing hops for the event deliveries is also $O(\ln \ln |V|)$. The upload phase takes $O(\ln \ln |V|)$ to deliver the event from the publisher to the root, while the download phase also takes $O(\ln \ln |V|)$ to deliver the event from the root to the subscriber. The proposed upload-and-download routing scheme is based on classic hierarchical routing schemes [37, 43, 44, 47]. Our novel contribution is the NSFA. NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies. The outstanding performance of our event delivery comes from the bound of NSFA's hierarchy. Our work is asymptotically compared with classic approaches in Table 3.2 (later in Section 3.5).

3.4.4 Peer Arrival, Departure, and Failure

Our pub/sub system can handle the peer arrival, departure, and failure. As shown in Fig. 3.3(c), NSFA can naturally hold when peer arrives and departs. NSFA also holds if the failure is uniformly distributed. Therefore, we only need to adjust the hierarchical levels of peers. (i) When a new peer arrives at the pub/sub system, its hierarchical level is set to the lowest hierarchical level among all of its neighbors (or one in the case of no neighbor). Then, this peer will express its interests through subscriptions, taking $O(\ln \ln |V|)$ messages. (ii) If a non-root peer wants to depart, it will unsubscribe from the system. Its connections with its children can be transferred to its parent. If a root peer wants to depart, its neighbor with the highest hierarchical level will be selected as the new root. The connections and subscriptions of this root will be transferred to the new root. The new root will register its address within the network to set up connections with the other roots. (iii) If a peer finds a failing child, it examines the subscriptions of its children and then unsubscribe the failed child. If a peer finds a parent failure, it re-selects a parent to re-subscribe. Since Theorem 3.7 claims that the tree depth is $O(\ln \ln |V|)$, the peer arrival, departure, and failure take

$O(\ln\ln|V|)$ messages. Our pub/sub system can also run Algorithm 2 periodically to reset NSFA's hierarchy. Moreover, the following theorem shows that, when a peer's interests are popular, the number of messages for its arrival, departure, and failure can be reduced:

Theorem 3.8. *If a peer has the same interests as* $\Omega(\frac{1}{\ln\ln|V|})$ *fraction of peers in the system, then its arrival, departure, and failure are expected to take* $O(\sqrt{\ln\ln|V|})$ *messages.*

Proof: Suppose a peer, v, has the same interests as a fraction, f, of all peers in the system. Note that a peer collects all the events interesting to itself, along with the interests of its descendants in the NSFA-based forest. Hence, the subscriptions of peers with larger labels are more likely to include v's interests than those with smaller labels. For peers with labels of one, f fraction of their subscriptions include v's interests. These peers (a fraction, c, of total peers by Eq. 3.3-3) notify their parents of their subscriptions. For peers whose labels are larger than one, the fraction of their subscriptions that include v's interests is $1-(1-f)\times(1-\frac{c}{1-c}\times f)\approx(1+c)f$, assuming $f\ll 1$ and $c\ll 1$. Then, let us remove the peers with labels of one, and look at peers with larger labels. By induction, for peers with labels of $i+1$, the fraction of their subscriptions that include v's interests is $(1+ic)f$.

We start with the peer arrival. Once a new peer (say u) arrives, it needs to notify all of its precedents to express its interests. However, such a notification can terminate earlier, if the subscription of u's precedent already includes u's interests. Suppose u has a distance of L to the root. The expected number of notification messages is:

$$\sum_{i=1}^{L}\Big\{\prod_{j=0}^{i-1}\big[1-(1+jc)f\big]\Big\}\leq\sum_{i=1}^{L}\big[1-(1+\frac{i-1}{2}c)f\big]^{i} \tag{3.4-5}$$

In Eq. 3.4-5, $\prod_{j=0}^{i-1}\big[1-(1+jc)f\big]$ is the upper bound of the probability that the i-th message in the notification process is sent. When $f=0$, L messages are sent

for the subscription. When $f \in \Theta(\frac{1}{L})$ and $i \in \Theta(\sqrt{L})$, $[1-(1+\frac{i-1}{2}c)f]^i$ becomes a constant according to the definition of the Euler's number. When $f \in \Theta(\frac{1}{L})$, $\sum_{i=1}^{L}\left[1-(1+\frac{i-1}{2}c)f\right]^i \in O(\sqrt{L})$. This is because the terms with $i \in \Omega(\sqrt{L})$ can be ignored in the summation. Since Theorem 3.7 claims that $L \in O(\ln\ln|V|)$, the proof completes the part for peer arrivals.

The proofs for peer departures and failures are similar and are omitted due to the page limitation. The key idea is that, when a peer has the same interests as many existing peers in the system, its arrival, departure, and failure can be handled more locally instead of notifying all the precedents. ■

3.5 Related Works and Discussions

The pub/sub system is a well-known paradigm for CDNs with the objective of providing efficient event deliveries from the publishers to the subscribers [48]. Classic pub/sub systems are usually implemented as overlay networks on the Internet, using hundreds of servers to deliver content to clients [28]. Since traditional pub/sub systems are infrastructure-dependent, infrastructure-free P2P-based pub/sub systems become another option [32]. Early pub/sub systems were built on unstructured P2P networks due to their low construction overheads and inherent robustness. However, event routing is inefficient [30], since peers are "randomly" connected to each other. Therefore, state-of-the-art pub/sub systems are built on structured P2P networks to improve the routing performance, through using DHTs to organize peers into specific topologies [49]. Nevertheless, the maintenance of DHTs lead to high construction overheads, and degrades the system robustness. We focus on an asymptotical approach. Therefore, some classic pub/sub systems (e.g., SIENA [31], PADRES [50], and Hermes [51]) are not compared, since their performances are not bounded.

Table 3.2 compares the existing systems and the NSFA-based pub/sub system, in

Table 3.2: Comparisons among Existing Systems and Our NSFA-based System.

Metrics	Structured P2P-based Pub/Sub		
	Terpstra [32]	Meghdoot [52]	PastryStrings [33]
Event routing	$O(\ln \lvert V \rvert)$	$O(\tau \lvert V \rvert^{\frac{1}{\tau}})$	$O(\log_{\mu} \lvert V \rvert)$
System robustness	$O(\ln \lvert V \rvert)$	$O(\tau \lvert V \rvert^{\frac{1}{\tau}})$	$O(\log_{\mu} \lvert V \rvert)$
Overhead	$O(\ln \lvert V \rvert)$	$O(\tau)$	$O(\mu \log_{\mu} \lvert V \rvert)$

Metrics	Unstructured P2P-based Pub/Sub			
	Sub-2-Sub [30]	Vitis [43]	Poldercast [44]	NSFA-based
Event routing	$O(\lvert V \rvert)$	$O(\ln^2 \lvert V \rvert)$	$O(\ln \lvert V \rvert)$	$O(\ln \ln \lvert V \rvert)$
System robustness	$O(1)$	N/A	$O(\ln \lvert V \rvert)$	$O(\ln \ln \lvert V \rvert)$
Overhead	$O(1)$	$O(1)$	$O(1)$	$O(1)$

terms of the event routing efficiency (number of routing hops for event deliveries), system robustness (message complexity for peer churns and failures), and overhead (storage consumption per peer). $|V|$ is the number of peers in the pub/sub system. Terpstra [32], Meghdoot [52] and PastryStrings [33] are structured P2P-based pub/sub systems. Terpstra organizes peers as a Chord [53]. The topology of Meghdoot is a Cartesian space with a dimensionality of τ. Compared with Terpstra, Meghdoot has a smaller overhead at the cost of a worse event routing efficiency and a worse system robustness. PastryStrings has a prefix-based routing through string trees, where μ is a pre-specified digit base. It sacrifices the overhead to improve the event routing efficiency. Sub-2-Sub [30], Vitis [43], and Poldercast [44] are unstructured P2P-based pub/sub systems. Sub-2-Sub considers that connections among peers are random, and uses broadcasts to deliver events. Sub-2-Sub has a very poor routing performance, but is highly robust. Vitis and Poldercast also consider that peer connections are random. While Vitis uses a clustering-based event delivery protocol, Poldercast chains all the peers to a ring network with shortcuts for efficient event routings (based on small worlds).

By contrast, our pub/sub system has the best event routing efficiency by leveraging the inherent NSFA of unstructured P2P networks, where peer connections are not

truly random [34]. NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies [36, 37, 38, 39, 40]. The overhead of the NSFA-based pub/sub system is asymptotically optimal (a constant size per peer), since each peer only needs to maintain its hierarchical level as the topological information. The system robustness is competitive, and can be further reduced for scenarios in which peers share the same interests (Theorem 3.8). A notable point for the proposed system is that peers with higher hierarchical levels tend to have larger loads than peers with lower hierarchical levels (the load is not balanced). We argue that peers with more connections could have more computational and storage resources to share, i.e., "with great power comes great responsibility."

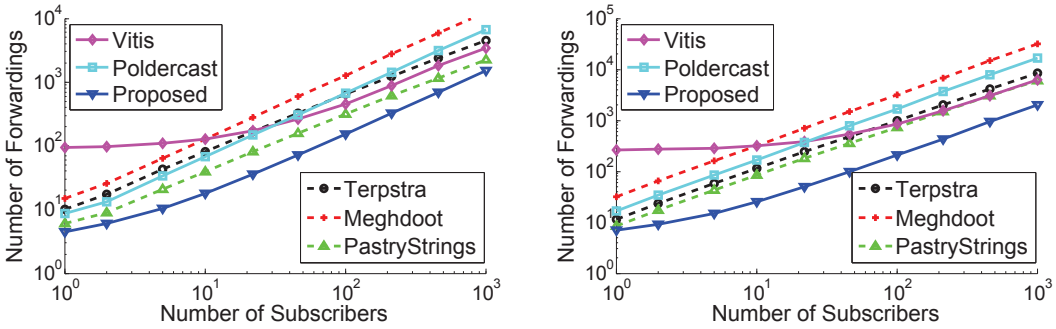
3.6 Experiments

3.6.1 Real Data-driven Experiments

This section conducts real data-driven experiments based on Gnutella [35] and Bitcoin [42]. To reveal the asymptotic performance gap, we use the complete Bitcoin dataset, which includes 6,336,769 peers and 37,450,461 connections. Our pub/sub system on Gnutella can facilitate large-size content deliveries, such as high-definition movie sharing services and high-volume enterprise data distributions. Our pub/sub system on Bitcoin can facilitate financial order disseminations among traders and payroll transactions among workers.

3.6.2 Pub/Sub System Evaluations

The proposed pub/sub system is evaluated through comparisons with Terpstra, Meghdoot, PastryStrings, Vitis, and Poldercast. These systems have been introduced in Section 3.5. Sub-2-Sub is not included, since it has an extreme design with the worst



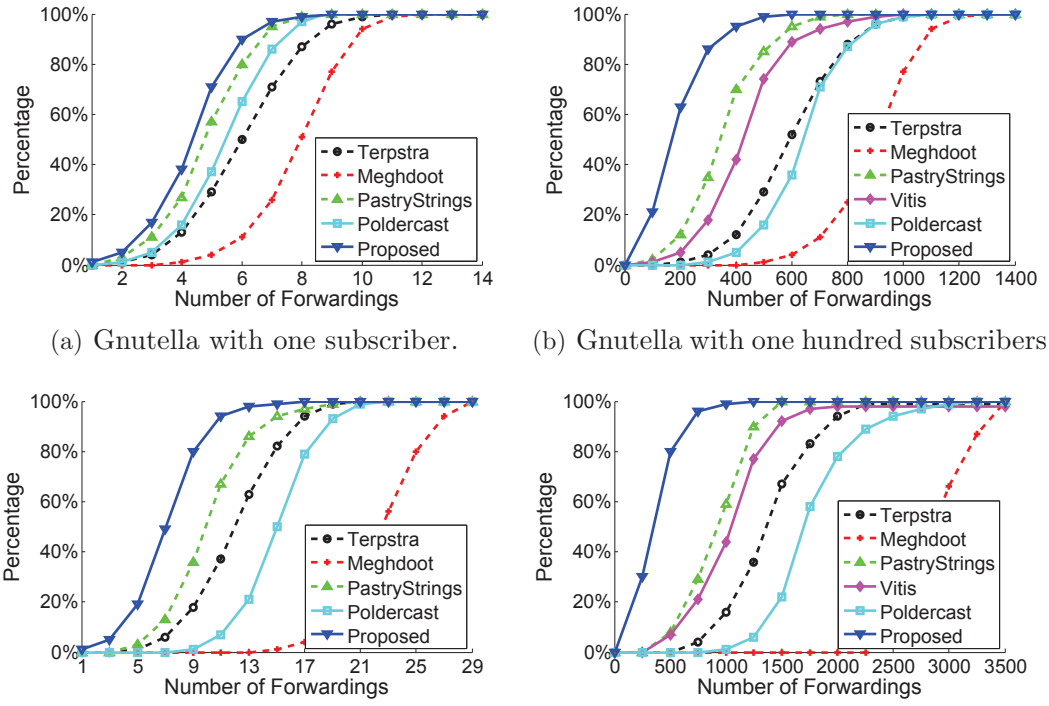
(a) Gnutella. (b) Bitcoin.

Figure 3.7: Event routing efficiency.

event routing efficiency, the best system robustness, and the smallest overhead. While unstructured P2P-based pub/sub systems are tested directly on the Gnutella and Bitcoin datasets, structured P2P-based pub/sub systems are tested with their own topologies that have the same number of peers with the datasets. In Meghdoot, we set $\tau = 8$ to minimize $\tau|V|^{\frac{1}{\tau}}$ for $|V| = 10,876$ in the Gnutella dataset. In PastryStrings, we use $\mu = 4$ as its digit base to encode event routings.

To evaluate the event routing efficiency, one publisher and several subscribers are uniform-randomly selected. The result of the total number of forwardings from the publisher to all subscribers in an event delivery is shown in Fig. 3.7. For smoothness, it is averaged over 100,000 times. The NSFA-based pub/sub system outperforms the others, since it has the best asymptotical bound of $O(\ln \ln |V|)$. Vitis has the worst performance when subscribers are few, since it has a bad asymptotical bound of $O(\ln^2 |V|)$. Meghdoot performs poorly due to the same reason. A notable point is that the performance gap between the proposed system and the other system is larger in Bitcoin than that in Gnutella. This is because Bitcoin has many more peers than Gnutella. The Bitcoin dataset has 6,336,769 peers and 37,450,461 connections. The asymptotical performance gap is revealed, when the network scales up.

Fig. 3.8 shows the Cumulative Distribution Function (CDF) with respect to the total number of forwardings in Fig. 3.7. Left and right subfigures are the results



(a) Gnutella with one subscriber.

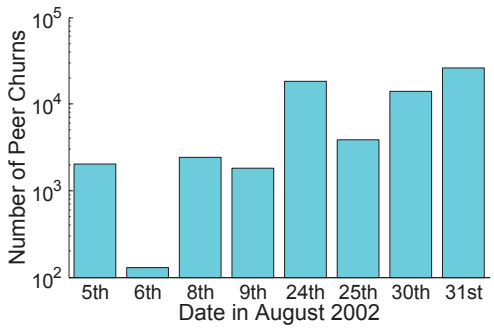
(b) Gnutella with one hundred subscribers.

(c) Bitcoin with one subscriber.

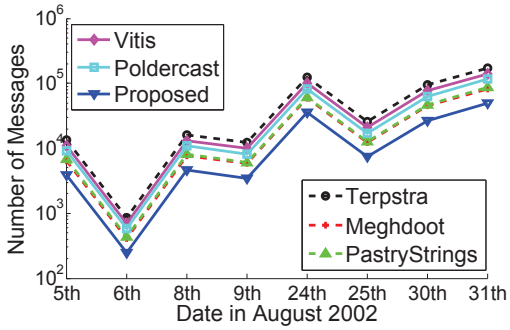
(d) Bitcoin with one hundred subscribers.

Figure 3.8: The CDF for the number of event forwardings.

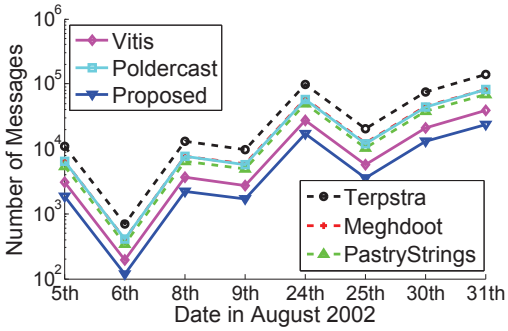
with one subscriber and one hundred subscribers, respectively. Vitis is not included in Figs. 3.8(a) and 3.8(c), since it needs a much larger number of forwardings when subscribers are few. Figs. 3.8(a) and 3.8(b) show that, when there are more subscribers, the proposed system has a more significant advantage over the other systems due to the lower asymptotical bound. Fig. 3.8(a) also shows that the NSFA-based system is not likely to deliver the event from a publisher to a subscriber within 3 forwardings, due to the event upload phase. On the other hand, most event deliveries in our system can be finished within 7 forwardings (Gnutella with one subscriber). When we have 100 subscribers, as shown in Fig. 3.8(b), most event deliveries can be finished within 500 forwardings, which is less than $7 \times 100 = 700$ forwardings. This is because the upload phases of different subscribers are shared. Figs. 3.8(c) and 3.8(d) present similar results for Bitcoin. The proposed pub/sub system performs significantly better than the other systems in Bitcoin, due to the



(a) Peer churn statistics.



(b) Message complexity (no shared interest).

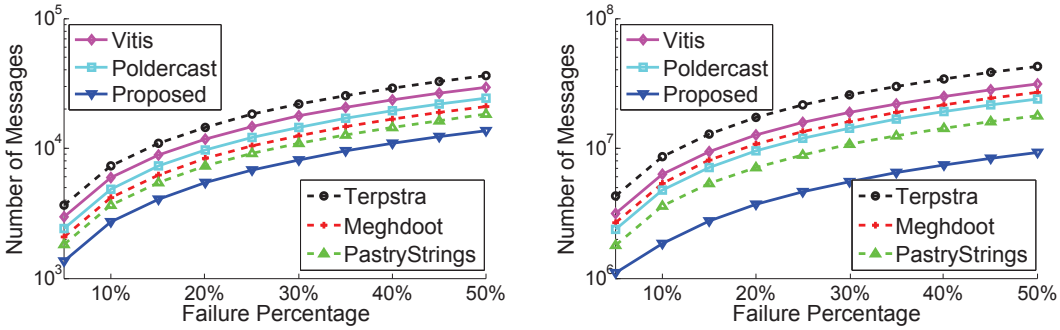


(c) Message complexity (shared interests).

Figure 3.9: System performance with respect to the peer churns in Gnutella.

asymptotical performance advantage. In Fig. 3.8(d), the number of forwardings used by our system is basically one third of PastryStrings and Vitis.

To evaluate the system robustness, the Gnutella dataset is further explored, since it has the peer churn records. However, the Bitcoin dataset does not include the peer churn records. Fig. 3.9(a) shows the available peer churn statistics in Gnutella on August 2002. August 24 and 31 have the largest numbers of peer churns. Note that the number of peer churns cannot be ignored with respect to the total number of peers in Gnutella. As shown in Fig. 3.3(c), NSFA holds when peer churns. Fig. 3.9(b) shows the number of messages dealing with peer churns, when peers do not share an interest. The number of messages scales up with the number of peer churns. Our NSFA-based pub/sub system uses the smallest number of messages to deal with peer churns. By contrast, the number of messages used by Terpstra is nearly quadrupled.



(a) Gnutella. (b) Bitcoin.

Figure 3.10: System performance with respect to the peer failures.

To study the message complexity when peers have common interests, we set up a special scenario where the interest of each peer is uniform-randomly chosen from ten given interests. The result is shown in Fig. 3.9(c). Our system and Vitis have lower message complexities since they can handle peer churns more locally.

The number of messages used for peer failures is shown in Fig. 3.10, where a given percentage of peers are randomly chosen to fail. Our NSFA-based system uses the smallest number of messages to handle the peer failures. By contrast, the number of messages used by Terpstra is almost tripled. Experiments are not conducted with respect to the overhead, since the overhead of our system is $O(1)$. Instead, we study the forwarding load distribution of root peers for our system in Gnutella and Bitcoin. 10,000 publisher-subscriber pairs are uniform-randomly selected for event deliveries. Fig 3.11 shows the cumulative distribution function of root peer forwardings. None of the root peers have more than 700 forwardings within 10,000 event deliveries.

3.7 Summary

This chapter proposes a scalable pub/sub system based on unstructured P2P networks, which are shown to have NSFAs. We propose that NSFA's hierarchy can be distributedly constructed, and has a better bound than classic hierarchies. By

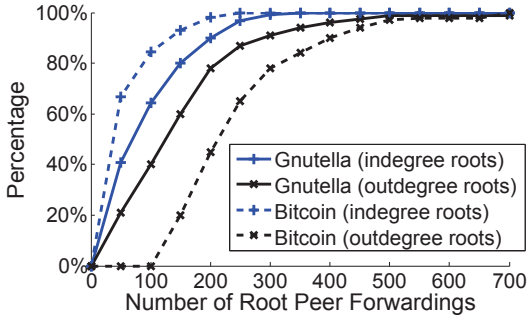


Figure 3.11: The CDF of root peer forwardings.

leveraging NSFA's hierarchy, the proposed pub/sub system achieves a competitive tradeoff among the event routing efficiency, system robustness, and overhead. The number of routing hops for event deliveries is $O(\ln \ln |V|)$. Each peer only maintains an overhead of a constant size as the topological information. Peer arrival, departure, and failure can be handled within a message complexity of $O(\ln \ln |V|)$. Experiments demonstrate the outstanding performance of the proposed pub/sub system.

CHAPTER 4

FRIEND RECOMMENDATION IN ONLINE SOCIAL NETWORKS: PERSPECTIVE OF SOCIAL INFLUENCE MAXIMIZATION

Chapters 4 and 5 focus on the influence propagation applications for social networks. More specifically, this chapter studies a friend recommendation strategy with the perspective of social influence maximization. This is because people may want to make new friends to maximize their social influences in online social networks. For example, business page owners on Facebook want to influence as many people as possible for commercial advantages. For the system provider (e.g., Facebook), the objective is to recommend a fixed number of new friends to a given user, such that the given user can maximize his/her social influence through making new friends. Our problem is proved to be NP-hard. A greedy friend recommendation algorithm with an approximation ratio of $1 - \frac{1}{e}$ is proposed, according to the submodular property. It involves a sub-problem of computing the influence spread. A novel method, which considers the multipath effect, is proposed to compute the influence spread. Experiments demonstrate the efficiency and effectiveness of our algorithms.

4.1 Introduction

Online Social Networks (OSNs) mainly focus on building social relations among users who share interests, activities, backgrounds, stories, and real-life connections. They are valuable tools used by many people to extend their daily contacts. Most

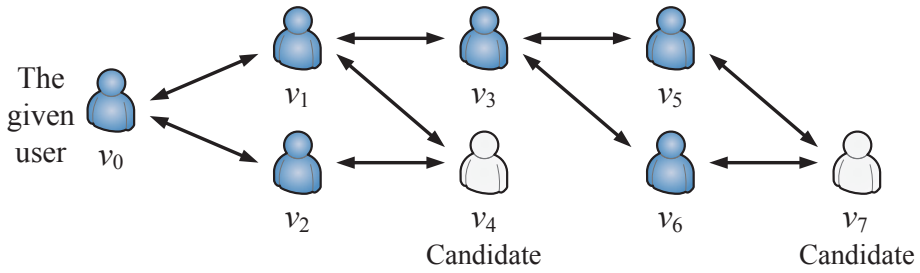


Figure 4.1: The tradeoff in the friend recommendation strategy.

OSNs are web-based and provide means for users to interact with each other over the Internet. OSNs are not only a way to keep in touch, but also a way of life. Existing OSNs such as Facebook, Twitter, and VK account for three of the top ten most-visited web sites in the world [54]. As of January 2014, 74% of online adults use OSNs [55].

As one of the essential components in OSNs, the friend recommendation system aims to seek appropriate people with whom users can make new friends. Classic approaches make recommendations according to the social proximities among the users, hypothesizing that people with close social circles are potential friends. For instance, the Facebook "People You May Know" feature recommends people to connect with each other, according to a friend-of-a-friend strategy [56]. In other words, if two unconnected users share many common friends, then they are recommended to become new friends. Friend recommendations on Facebook prioritize friends-of-friends over strangers (i.e., people who are not friends or friends-of-friends). Content-based and location-based recommendation strategies have also been proposed. In these approaches, people who share similar contents, or are geographically nearby, are recommended to connect with each other [57, 58].

We observe that people may want to make new friends with the objective of maximizing their social influences. The world-famous best-selling book, "How to Win Friends and Influence People," considers making friends and influencing people to be closely interrelated [59]. Consequently, people can use OSNs to advance their

careers and businesses. For example, Facebook provides business page services [60] for their owners to influence other people for commercial advantages (selling and advertising). Twitter also provides page promotion services for salesmen to attract high-value followers as potential customers. A successful story is that of Drew Ressler, who dramatically gained 1,300 new followers while targeting audiences interested in music and photography [61].

This chapter focuses on the friend recommendation strategy with the perspective of social influence maximization. For the system provider, the objective is to recommend a fixed number of new friends to a given user, such that the given user can maximize his/her social influence through new friends. Our problem is proved to be NP-hard. New challenges arise from the tradeoff between the *friend acceptance probability* and the *propagation capability*. The friend acceptance probability is self-explanatory. The propagation capability measures the influence spread beyond the given user that is brought by the recommended friend. A motivational example is shown in Fig. 4.1, where v_0 wants to maximize his/her social influence through making a new friend (candidates are v_4 and v_7). Arrows in Fig. 4.1 are bidirectional friendships. In terms of the friend acceptance probability, v_4 is a friend-of-a-friend of v_0, but v_7 is a stranger. Hence, v_4 is more likely to accept the friend request from v_0 and then propagate v_0's influence. However, in terms of the propagation capability, v_4 cannot further propagate v_0's influence to the users v_5 and v_6 (since they do not connect with each other). On the other hand, v_7 is influential and can propagate v_0's influence to v_5 and v_6, but v_7 is not likely to accept the friend request from v_0. This is because v_0 and v_7 are not socially proximal to each other. The tradeoff between the friend acceptance probability and the propagation capability should be investigated.

A greedy friend recommendation strategy is proposed to balance the above tradeoff. Moreover, it is an extension of the classic social influence maximization problem [62]. It involves an NP-hard sub-problem of the influence spread computation [63]: given

the OSN topology and a specified user, how many people can this user influence? Since OSNs are typically large, the scalability issue challenges classic solutions [64]. Through leveraging the structural properties of OSNs, we propose a novel method to efficiently compute the influence spread, based on the multipath effect.

Our main contributions are summarized as follows:

- We address a novel friend recommendation problem with the perspective of social influence maximization. Our problem is proved to be NP-hard. A greedy approximation algorithm with a ratio of $1 - \frac{1}{e}$ is discussed.
- We propose a novel influence spread computation method to support greedy friend recommendations. The multipath effect is explored.
- Extensive real data-driven experiments are conducted to evaluate the proposed algorithms. Evaluation results are shown from different perspectives to provide insightful conclusions for real-world applications.

The remainder of this chapter is organized as follows. Section 4.2 surveys related works. Section 4.3 formulates the problem. Section 4.4 describes the greedy recommendation algorithm. Section 4.5 computes the influence spread. Section 4.6 includes experiments. Finally, Section 4.7 concludes this chapter.

4.2 Related Work

Social Influence Propagation. Independent cascade is one of the most classic models for describing social influence propagations [62, 65]. It starts with a set of initially-influenced people, and then executes a probabilistic rule to propagate influences. The number of eventually-influenced people is denoted as the influence spread, the computation of which is NP-hard [63]. Several time-efficient methods were proposed to estimate the influence spread. Chen et al. [63, 64] simplified

the influence spread computation by restricting the influence to propagate along the maximum influence path. Borgs et al. [66] studied the influence spread by sampling methods, assuming that the statistical properties of an OSN is stable. This chapter uses the existing independent cascade model to address a novel friend recommendation problem. Our contributions also include a novel influence spread computation method.

Friend Recommendation. The friend recommendation system is an essential component of an OSN. Authors in [56, 67] hypothesized that people with close social circles are potential friends. Bu et al. [57] proposed that users who like the same music should be recommended to become friends with each other. Zhang et al. [68] considered that users make friends by sharing the same profiles. As for location-based approaches [58, 69], geographically nearby users are recommended to connect with each other. Differing from previous approaches, our friend recommendation strategy focuses on the perspective of maximizing the social influence of a specified user.

4.3 Preliminaries and Problem Formulation

4.3.1 Independent Cascade

The scenario of this chapter is an OSN, which is modeled as a directed weighted graph $G = (V, E)$. Here, V is a set of nodes (users), and $E \subseteq V^2$ is a set of weighted edges (friendships between users). An edge from user v to user u is denoted by e_{vu} with the edge weight of w_{vu}. Edge weights serve as probabilities for influence propagations, depending on the friendship closeness and the content popularity [70]. We use existing works [62, 65, 63, 64] to determine edge weights. The *independent cascade model* is used to simulate influence propagations. It starts with a set of initially-influenced nodes (called seed nodes). The other nodes are initially-uninfluenced. The influence propagation process unfolds in discrete time steps according to the

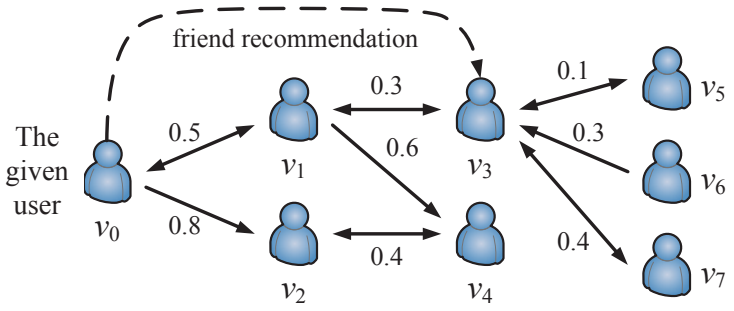


Figure 4.2: An example for the friend recommendation.

following probabilistic rule [62]. When a node v first becomes influenced in a time step, it has a single chance to influence each currently uninfluenced neighbor u. Then, the probability that u is influenced by v depends on w_{vu}. If u has multiple newly-influenced neighbors, their influence propagations can be sequenced in an arbitrary order. Once u is influenced, it will influence its neighbors in the next time step; however, u does not propagate any further influences in subsequent time steps. The above process terminates until no more uninfluenced nodes are influenced. We have:

Definition 4.1. The expected number of nodes eventually-influenced by seed nodes is defined as the *influence spread*.

Unfortunately, the computation of the influence spread is known as NP-hard [63]. Currently, it could be computed by time-consuming Monte-Carlo simulations, information-lossy heuristics, or sampling methods with strong assumptions.

4.3.2 Problem Formulation

This chapter studies the friend recommendation strategy with the perspective of social influence maximization (i.e., *maximizing influence spread*). The motivation is that people may want to make new friends to maximize their social influence. People, such as political party leaders, film stars, and business salesmen, sometimes combine making friends and influencing people as a lifestyle. Therefore, OSNs have a strong potential for adopting our friend recommendation strategy. For example, a

salesman on Facebook or Twitter would like to influence as many people as possible for commercial advantages [71, 72].

This chapter designs a friend recommendation strategy for an OSN system provider (e.g., Facebook). The objective is to recommend a fixed number (denoted by k) of new friends to a given user (denoted by v_0), such that v_0 can maximize his/her social influence spread through making new friends. The independent cascade model [62] is adopted as the influence cascade model, where v_0 is initially-influenced and the other nodes are initially-uninfluenced. The existing friends of v_0 are eliminated in the recommendations. We want to maximize the influence spread of v_0 through k new friendships (new edges). The friend acceptance probabilities depend on the social proximities between v_0 and the recommended people. This is because a stranger is less likely to accept the friend request from v_0 than a friend-of-friend of v_0.

An example is shown in Fig. 4.2, where edges are directed and the numbers on edges are their weights. If v_3 is recommended to v_0, a new edge of $e_{v_0v_3}$ may be added, depending on the friend acceptance probability that is estimated by their social proximity. Similar to the other edge weights, the weight, $w_{v_0v_3}$, of the new edge is also determined by existing works [62, 65, 63, 64], according to the friendship closeness and the content popularity. The challenge comes from the tradeoff between the friend acceptance probability and the propagation capability. Although influential strangers can effectively propagate v_0's influence, they are less likely to accept the friend request from v_0. On the other hand, friends-of-friends are likely to accept the friend request from v_0, but may not effectively propagate v_0's influence. This tradeoff should be explored.

4.4 Combining Friend Recommendation and Social Influence Maximization

4.4.1 Friend Acceptance Probability

This subsection describes the friend acceptance probability. If v is recommended to v_0 as a new friend, v_0 may influence more people through forming a new friendship with v. Let f_{v_0v} denote the friend acceptance probability that v accepts the friend request from v_0 (i.e., a new edge of e_{v_0v} is formed). Strangers are people who are not friends or friends-of-friends of v_0. We have the following justifications for f_{v_0v}:

- f_{v_0v} can be different for different strangers of v. This is because v_0 could judge their friendship potential from multiple fields such as common interests, school of graduation, place of work, and so on.
- Compared to the event in which v is a stranger of v_0, f_{v_0v} should be larger if v is a friend-of-a-friend of v_0. This is because people are more likely to accept a friend request from friends-of-friends than from strangers.

These two unique properties are the major differences between our metric and existing classic metrics (e.g., Jaccard's coefficient and Katz coefficient) [62]. Based on above justifications, f_{v_0v} is formally defined as follows:

$$f_{v_0v} = \alpha_v \times \frac{|N(v_0) \cap N'(v)|}{|N(v_0) \cup N'(v)|} + \beta_v \tag{4.4-1}$$

In Eq. 4.4-1, $N(v)$ and $N'(v)$ denote the sets of outgoing and incoming neighboring nodes of v, respectively. The fraction of $\frac{|N(v_0)\cap N'(v)|}{|N(v_0)\cup N'(v)|}$ is the common neighbor similarity from v_0 to v. It measures the number of common friends of v_0 and v. α_v and β_v are coefficients. To guarantee $0 \leq f_{v_0v} \leq 1$, we have $0 \leq \alpha_v \leq 1$, $0 \leq \beta_v \leq 1$, and $0 \leq \alpha_v + \beta_v \leq 1$. While α_v measures the impact of friends-of-friends, β_v measures the

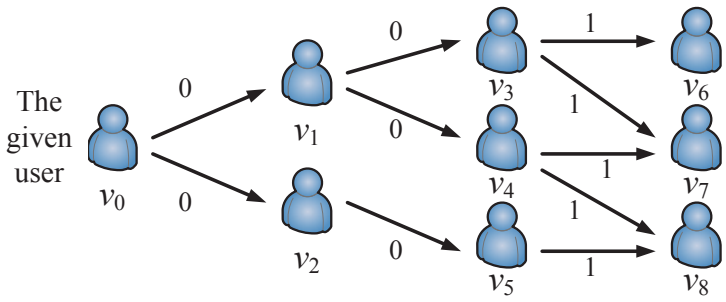


Figure 4.3: Proof of NP-hardness.

probability that people form a friendship with a stranger. α_v and β_v can vary among different people. When v is a friend-of-a-friend of v_0, f_{v_0v} is no smaller than β_v. This is because $N(v_0) \cap N'(v) \neq \emptyset$. On the other hand, when v is a stranger to v_0, f_{v_0v} reduces to β_v, which is its minimum value. An example is shown in Fig. 4.2. If we have $\alpha_{v_3} = 0.5$ and $\beta_{v_3} = 0.1$, then $f_{v_0v_3} = 0.5 \times \frac{1}{5} + 0.1 = 0.2$.

Note that f_{v_0v} does not depend on the edge weight w_{v_0v}. This is simply because the edge weight measures the influence propagation probability rather than the friendship closeness. Eq. 4.4-1 can be improved by considering the friendship closeness. Another notable point is that we use existing works [62, 65, 63, 64] to determine the weights of edges, including the weights of new edges that results from friend recommendations.

4.4.2 NP-hardness, Submodularity, and Greedy Approximation

This subsection explores the friend recommendation problem. Let R denote the set of users that are recommended to v_0 for making new friends. The constraint is $|R| \leq k$, and $|R|$ is the set cardinality of R. Once a user (say v) is recommended to v_0, the probability that v accepts the friend request from v_0 is f_{v_0v}. The independent cascade model is adopted stimulate influence propagations. In our problem, only v_0 is initially-influenced, and the other nodes are initially-uninfluenced. Let $\sigma(R)$ denote the influence spread. Based on Definition 4.1, $\sigma(R)$ is the expected number of nodes eventually-influenced by v_0 with friend recommendations in R. Note that,

when $R = \emptyset$, $\sigma(R)$ is not necessarily 0, since v_0 can still influence existing friends. We have the following theorem:

Theorem 4.2. *Our friend recommendation problem, which selects R to maximize $\sigma(R)$, is NP-hard.*

Proof: The proof is done by reducing the *Maximum Coverage Problem* (MCP) to a special case of our problem. The MCP is NP-hard [73], and is based on sets of elements. Given a number of k, its objective is to select k sets, such that the number of covered elements are maximized. If a set is selected, its elements are covered.

The reduction is done by mapping sets and elements to 2-hop and 3-hop friends of v_0, respectively. Edge weights in our problem are specially designed. Only weights of edges from 2-hop friends to 3-hop friends of v_0 are 1, and the others are 0. As a special case of our problem, let G be a directed acyclic graph with the source of v_0. Fig. 4.3 shows such an example: set $\{v_6, v_7\}$ is mapped to node v_3, set $\{v_7, v_8\}$ is mapped to node v_4, and set $\{v_8\}$ is mapped to node v_5. We use $f_{v_0v} = 1$ and $w_{v_0v} = 1$ for all $v \in R$. At this time, the optimal recommendation will only recommend 2-hop friends of v_0, since 3-hop friends can be influenced through 2-hop friends. For example, the optimal recommendation will not recommend v_6 and v_7, since the recommendation of v_3 is better. Hence, people, who are influenced by v_0 in the optimal recommendation, are composed of exactly k 2-hop friends and some 3-hop friends that correspond to the optimally-covered elements in the MCP. Therefore, the MCP reduces to a special case of our problem. Since the MCP can be reduced from the set cover problem that is NP-complete [73], our friend recommendation problem is NP-hard. ■

The idea of our proof is to weaken the tradeoff between the friend acceptance probability and the propagation capability by using $f_{v_0v} = 1$ and $w_{v_0v} = 1$ for all $v \in R$. We have:

Definition 4.3. $\sigma(R)$ is submodular, if it satisfies a natural diminishing returns

Algorithm 3 Greedy Friend Recommendation (GFR)

Input: The graph, G, and the given user, v_0;
Output: The set of recommended people, R;

1: Initialize $R = \emptyset$;
2: **for** $i = 1$ to k **do**
3: Select $v = \arg\max_u \left[\sigma(R \cup \{u\}) - \sigma(R)\right]$;
4: $R = R \cup \{v\}$;
5: **return** R;

property: the marginal gain from adding a node to the set R is at least as high as the marginal gain from adding the same node to a superset of R.

Theorem 4.4. *$\sigma(R)$ is submodular with respect to R.*

Proof: Let $p_R(v)$ denote the probability that the node v is eventually-influenced by v_0 with R. Clearly, we have $\sigma(R) = \sum_{v \in V \setminus \{v_0\}} p_R(v)$. We show that $p_R(v)$ is submodular with respect to R, through considering the influence propagation paths from v_0 to v. To see this, let R' denote an arbitrary superset of R, i.e., $R \subseteq R'$. Submodularity means that

$$p_{R \cup \{u\}}(v) - p_R(v) \geq p_{R' \cup \{u\}}(v) - p_{R'}(v) \tag{4.4-2}$$

where $u \in V \setminus R$. This inequality in Eq. 4.4-2 clearly holds, when the influence propagation paths from v_0 to v via u have some overlaps with those via $R' \setminus R$. An example of such overlaps is shown as u_1 and u_2 in Fig. 4.4, where dashed arrows are influence propagation paths. The equality in Eq. 4.4-2 holds, only when the influence propagation paths from v_0 to v via u are fully independent with those via $R' \setminus R$. An example is u_3 in Fig. 4.4. Therefore, for all v, $p_R(v)$ is submodular with respect to R. Considering that a non-negative linear combination of submodular functions is also submodular, we can conclude that $\sigma(R)$ is submodular with respect to R. ■

The insight behind Theorem 4.4 is very intuitive: the social influences brought by the people who are recommended later have potential overlaps with those who are

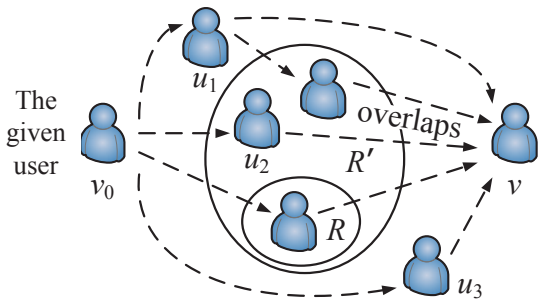


Figure 4.4: Proof of submodular property.

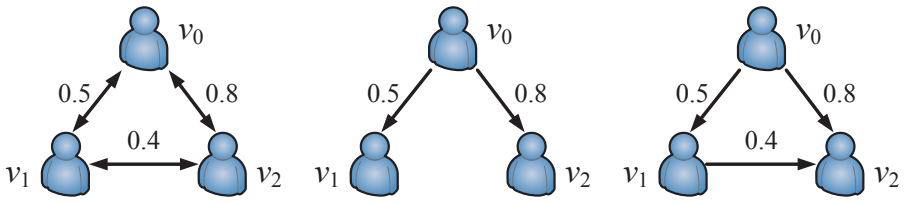
recommended earlier. Hence, the marginal gain of the influence spread satisfies the law of diminishing returns. According to [74], a greedy algorithm with a submodular objective function guarantees an approximation ratio of $1-\frac{1}{e}$ to the optimal algorithm. This algorithm is shown as Algorithm 3. It iteratively selects the user, who can maximize the marginal influence spread, into the recommendations. Note that the tradeoff between the friend acceptance probability and the propagation capability has been automatically considered in the computation of $\sigma(R)$. Existing friends of v_0 are eliminated in the friend recommendations.

A critical drawback of Algorithm 3 is that the computation of the influence spread is NP-hard [63]. Consequently, line 3 in Algorithm 3 cannot be optimally computed within a polynomial time complexity. State-of-the-art [64, 63, 75] cannot decently solve this problem. Hence, in the next section, a novel method is proposed to efficiently and accurately compute the influence spread. It serves as a sub-algorithm for line 3 in Algorithm 3, i.e., it computes $\sigma(R)$ for a given R.

4.5 Influence Spread Computation

4.5.1 Classic Approaches and Their Limitations

The influence spread computation is NP-hard [63], and it is usually done by Monte-Carlo simulations. The most classic approaches for this problem are proposed by Chen



(a) Original graph. (b) Tree reduction. (c) DAG reduction.

Figure 4.5: An example for the classic approaches and their limitations.

et al. [63, 64]. They simplified the influence spread computation by restricting the influence to propagate along the maximum influence path. In other words, the graph is reduced to an arborescence structure (called maximum influence arborescence model). Later, their method is improved by reducing the graph to a directed acyclic graph (DAG). An example is shown in Fig. 4.5. The original graph is shown in Fig. 4.5(a), where v_0 is initially-influenced and the other nodes are initially-uninfluenced. The probability that the node v is eventually-influenced is denoted by $p_R(v)$. We have $\sigma(R) = \sum_{v \in V \setminus \{v_0\}} p_R(v)$. The challenge comes from the *multipath effect*, meaning that there may exist an exponential number of paths to propagate the influence from v_0 to an arbitrary node. In Fig. 4.5(a), v_1 can be possibly influenced via the v_0-v_1 path, or the v_0-v_2-v_1 path. Note that the number of paths in a graph increases exponentially with respect to the number of nodes, leading to the NP-hardness for the influence spread computation. To restrict the number of paths, the approach in [63] reduces the graph to a tree, as shown in Fig. 4.5(b). We get $p_R(v_1) = 0.5$ and $p_R(v_2) = 0.8$ by this approach. It is information-lossy since $e_{v_1 v_2}$ and $e_{v_2 v_1}$ are discarded. The improved approach in [64] reduces the graph to a DAG, as shown in Fig. 4.5(c). We get $p_R(v_1) = 0.5$ and $p_R(v_2) = 0.84$ for this approach, where only $e_{v_2 v_1}$ is discarded. Considering the multipath effect, the optimal result in Fig. 4.5(a) is that $p_R(v_1) = 0.5 + 0.8 \cdot 0.4 - 0.5 \cdot 0.8 \cdot 0.4 = 0.66$ and $p_R(v_2) = 0.8 + 0.5 \cdot 0.4 - 0.8 \cdot 0.5 \cdot 0.4 = 0.84$. Classic approaches are inaccurate, since some edges are removed to restrict the number of paths for influence propagations.

4.5.2 Multipath Effect in Influence Propagations

Our key idea for the influence spread computation is to mitigate the multipath effect by considering several top influence propagation paths within a non-exponential time complexity. There exists a tradeoff between the number of paths and the computation accuracy of the influence spread. If we consider more paths in the computation of the influence spread, then the result is more accurate, at the cost of a higher time complexity. Through a coarse estimation, this subsection shows how many paths are sufficient to obtain an accurate $p_R(v)$ for node v.

Our coarse estimation is based on the existing literature for scale-free networks [76]. OSNs are typically scale-free [77]. Let $\langle \cdot \rangle$ denote the mean value of a variable. For example, $\langle w \rangle$ is the average edge weight and $\langle d \rangle$ is the average out-degree. Let $|V|$ denote the number of nodes in G. Let $N_L(v)$ denote the expected number of paths from v_0 to v that have L intermediate nodes. We assume that each node (excluding v_0 and v) has a probability of λ_v to be an intermediate node in a path from v_0 to v. Note that λ_v ($0 \leq \lambda_v \leq 1$) represents the graph skewness among different users. We have:

$$N_L(v) = \left[\frac{(|V|-2)!}{(|V|-2-L)!} \cdot \lambda_v^L \cdot (1-\lambda_v)^{|V|-L} \right] \cdot \frac{\langle d \rangle^L}{|V|^L} \tag{4.5-3}$$

Here, $\frac{(|V|-2)!}{(|V|-2-L)!}$ is the number of permutations for selecting L nodes from $|V|-2$ nodes (excluding v_0 and v) as *ordered* intermediate nodes on the path from v_0 to v. The probability of the corresponding permutation is $\lambda_v^L \cdot (1-\lambda_v)^{|V|-L}$. Then, $\frac{\langle d \rangle}{|V|}$ is the average probability that a predecessor node on the path connects to the successor node. In scale-free networks, we typically consider that L is small with respect to $|V|$

[76], meaning that $\frac{(|V|-2)!}{(|V|-2-L)!} \cdot \frac{1}{|V|^L} \approx \frac{1}{|V|}$. We rewrite Eq. 4.5-3:

$$N_L(v) \approx \frac{(1-\lambda_v)^{|V|}}{|V|} \cdot \Big[\frac{\lambda_v}{1-\lambda_v}\langle d\rangle\Big]^L \tag{4.5-4}$$

When $\frac{\lambda_v}{1-\lambda_v}\langle d\rangle > 1$, $N_L(v)$ is expected to grow exponentially with respect to L, and thus, traversing all the paths from v_0 to v is time-consuming (the general case). Let L^* denote the length of the unweighted shortest path from v_0 to v, we have:

$$N_{L^*}(v) \approx \frac{(1-\lambda_v)^{|V|}}{|V|} \cdot \Big[\frac{\lambda_v}{1-\lambda_v}\langle d\rangle\Big]^{L^*} \tag{4.5-5}$$

L^* and $N_{L^*}(v)$ can be obtained by Dijkstra's algorithm. $|V|$ and $\langle d\rangle$ can be obtained through network statistics. Therefore, λ_v can be adaptively computed through Eq. 4.5-5.

The expected probability that v is influenced by v_0 through a path of length L is $\langle w\rangle^L$. All the paths with length L, in total, should bring an expected influence propagation probability of $1-(1-\langle w\rangle^L)^{N_L(v)}$. If these paths are independent of each other and $\langle w\rangle^L \ll 1$, then we have the following estimation:

$$1-(1-\langle w\rangle^L)^{N_L(v)} \approx \langle w\rangle^L N_L(v) \approx \frac{(1-\lambda_v)^{|V|}}{|V|} \cdot \Big[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\Big]^L \tag{4.5-6}$$

Eq. 4.5-6 implies two insights. When $\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle \geq 1$, v is likely to be eventually-influenced by v_0, since Eq. 4.5-6 becomes close to one. In this case, several top paths are sufficient to propagate v_0's influence to v. This is because these paths already bring an influence probability that is close to one. Consequently, we do not need to consider longer paths for the computation of $p_R(v)$. The hard scenario is $\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle < 1$. In this case, Eq. 4.5-6 decreases exponentially with respect to L. This indicates that v_0's influence is much more likely to propagate along the shorter paths than the longer paths. Top paths can dominate the influence propagation. When $\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle < 1$,

the following equation can show such dominations:

$$\frac{1-\prod_{L=L^*}^{L^*+l}(1-\frac{(1-\lambda_v)^{|V|}}{|V|}\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^L)}{1-\prod_{L=L^*}^{\infty}(1-\frac{(1-\lambda_v)^{|V|}}{|V|}\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^L)} \geq \frac{\sum_{L=L^*}^{L^*+l}\frac{(1-\lambda_v)^{|V|}}{|V|}\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^L}{\sum_{L=L^*}^{\infty}\frac{(1-\lambda_v)^{|V|}}{|V|}\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^L} = 1-\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^{l+1} \tag{4.5-7}$$

The fraction in the top line of Eq. 4.5-7 is the ratio of (i) the expected influence propagation probability brought by paths with lengths from L^* to $L^* + l$ to (ii) the expected influence propagation probability brought by all paths. Eq. 4.5-7 shows that, if we consider a little bit more paths (in addition to the shortest path), then the computational error of $p_R(v)$ can be exponentially reduced. Although the estimation in Eq. 4.5-7 is coarse-grained, it still provides an important intuition for the influence spread computation: if we rank all the paths from v_0 to v by their influence probabilities, then the several top paths are sufficient to approximate $p_R(v)$. The next subsection computes the influence spread based on this intuition.

4.5.3 Multipath-sensitive Influence Spread Computation

Motivated by Eq. 4.5-7, we propose a Polynomial-Time Approximation Scheme (PTAS) for the influence spread computation, assuming $\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle < 1$. The key idea to mitigate the multipath effect by considering several top influence propagation paths within a non-exponential time complexity. We argue that this PTAS should also work well when $\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle \geq 1$, since several top paths already bring an influence probability that is close to one (as analyzed in Eq. 4.5-6). Let ε denote a control parameter ($0 \leq \varepsilon \leq 1$). The PTAS is expected to obtain a ratio, $1-\varepsilon$, to the optimal algorithm (under some assumptions in the derivation of Eq. 4.5-7). Based on

Eq. 4.5-7, we have:

$$1-\varepsilon = 1-\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right]^{l+1} \tag{4.5-8}$$

Since parameters ε, $\langle w\rangle$, $\langle d\rangle$, and λ_v are known, we have:

$$l = -1+\ln\varepsilon \,/\, \ln\left[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle\right] \tag{4.5-9}$$

The total number of paths from v_0 to v, denoted by θ_v, is:

$$\begin{aligned}\theta_v &= \sum_{L=L^*}^{L^*+l} N_L(v) = N_{L^*}(v)\cdot\sum_{i=0}^{l}\left[\frac{\lambda_v}{1-\lambda_v}\langle d\rangle\right]^i \\ &= N_{L^*}(v)\cdot\frac{\left[\frac{\lambda_v}{1-\lambda_v}\langle d\rangle\right]^{\frac{\ln\varepsilon}{\ln[\frac{\lambda_v}{1-\lambda_v}\langle w\rangle\langle d\rangle]}}-1}{\left[\frac{\lambda_v}{1-\lambda_v}\langle d\rangle\right]-1}\end{aligned} \tag{4.5-10}$$

θ_v is rounded to an integer for the algorithm implementation.

Algorithm 4 is proposed as the PTAS. It is a sub-algorithm for line 3 in Algorithm 3. It include three stages:

- The first stage (lines 1 to 4) processes the friend recommendations. For $v \in R$, a new edge is added. The edge weight is determined by the existing works [62, 65, 63, 64]. f_{v_0v} is computed based on Eq. 4.4-1. Edge weights are updated to show the impact of the friend acceptance probability.
- The second stage (lines 5 to 7) determines the set of paths to compute the influence spread. Through a coarse estimation, the number of paths, θ_v, is *adaptively* determined. Note that θ_v can be different for different v, since the graph skewness is considered by λ_v. We convert the edge weights to the distance weights through a negative $\log(\cdot)$ operation. Yen's algorithm [78] is called to compute the set of loopless weighted shortest paths from v_0 to v. For each v,

Algorithm 4 Influence Spread Computation (ISC)

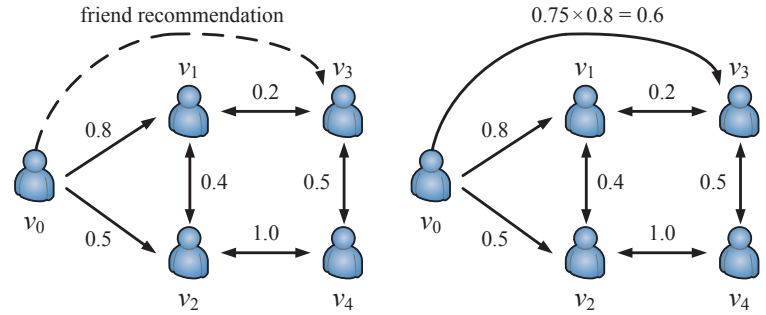
Input: The graph, G, and the given user, v_0;
The set of recommended people, R;
The control parameter, ε;
Output: The influence spread, $\sigma(R)$;

1: **for** each node v in R **do**
2: Add the edge, e_{v_0v}, with weight, w_{v_0v}, to the graph, G;
3: Compute f_{v_0v} based on Eq. 4.4-1;
4: Update $w_{v_0v} = f_{v_0v} \times w_{v_0v}$;
5: Call unweighted Dijkstra's algorithm from v_0 to determine the unweighted shortest paths from v_0 to the other nodes; then, L^* and $N_{L^*}(v)$ can be obtained for each v;
6: Parameters L^*, $N_{L^*}(v)$, $\langle d \rangle$, V, and ε are known; based on Eq. 4.5-5, compute λ_v for each v; based on Eq. 4.5-10, compute the total number of paths, θ_v, for v;
7: Convert the edge weights to distance weights through a negative $\log(\cdot)$ operation; call weighted Yen's algorithm to compute the loopless weighted shortest paths from v_0 to the other nodes; for each v, θ_v paths from v_0 are obtained;
8: **for** each node v in G except v_0 **do**
9: Based on θ_v loopless paths, construct a DAG from v_0 to v; based on the topological order, compute $p_R(v)$;
10: **return** $\sigma(R) = \sum_{v \in V \setminus \{v_0\}} p_R(v)$;

we obtain θ_v paths from v_0. If $\theta_v < 1$, we use only one path. If θ_v is larger than the number of available loopless paths, then all loopless paths are used.

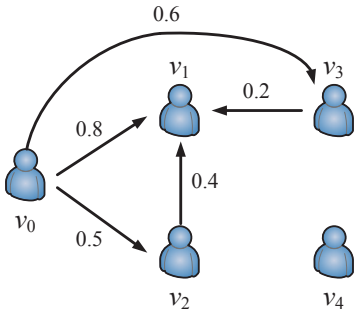
- In the third stage (lines 8 to 10), for each v, we compute the probability that v is influenced by v_0, i.e., $p_R(v)$. Since θ_v loopless paths are obtained by the second stage, a DAG can be constructed. Then, $p_R(v)$ can be computed following a topological order in the DAG.

An example of Algorithm 4 is shown in Fig. 4.6. Fig. 4.6(a) shows G with $R = \{v_3\}$. In the first stage of Fig. 4.6(b), $e_{v_0v_3}$ is added with $w_{v_0v_3} = 0.75$. Existing works [62, 65, 63, 64] are used to determine all the edge weights, including $w_{v_0v_3}$. Suppose $\alpha_{v_3} = 0.9$ and $\beta_{v_3} = 0.5$, then we compute $f_{v_0v_3} = 0.9 \times \frac{1}{3} + 0.5 = 0.8$ based on Eq. 4.4-1. To incorporate the friend acceptance probability, $w_{v_0v_3}$ is updated to be 0.6. In the second stage of Fig. 4.6(c), we determine the set of paths for the influence spread computation. The number of paths, θ_v, is determined

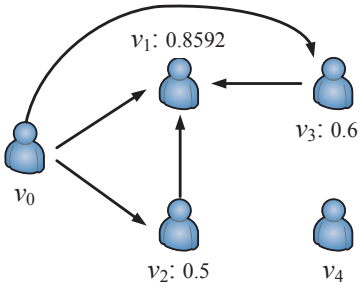


(a) The original graph.

(b) The first stage.



(c) The second stage (for v_1).



(d) The third stage (for v_1).

Figure 4.6: An example for the influence spread computation.

by Eqs. 4.5-5 and 4.5-10. Our example only includes the process for v_1. For simplicity, let $\theta_{v_1} = 3$. Then, Yen's algorithm determines three loopless weighted shortest paths from v_0 to v_1, i.e., v_0-v_1, v_0-v_2-v_1, and v_0-v_3-v_1. Since these paths are loopless, the third stage of Fig. 4.6(d) computes $p_R(v_1)$ with a topological order, $p_R(v_1) = 1 - (1 - 0.8) \times (1 - 0.5 \cdot 0.4) \times (1 - 0.6 \cdot 0.2) = 0.8592$.

Due to Yen's algorithm [78], the time complexity of Algorithm 4 is $O(\theta|V| \cdot (|E| + |V| \log |V|))$. Here, θ is the maximum number of paths for a node, i.e., $\theta = \max_v \theta_v$. The first stage takes $O(k)$ to go through each recommendation. The third stage takes $O(|V| \cdot (|V| + |E|))$ due to the topological sort. The key insight of Algorithm 4 is to mitigate the multipath effect by considering several top influence propagation paths within a non-exponential time complexity. To guarantee accuracies, the number of need paths is estimated by Eqs. 4.5-5 and 4.5-10.

Table 4.1: Dataset Statistics

	Facebook	Epinions	Wiki
Number of nodes	63,731	18,098	7,115
Number of edges	817,035	355,754	103,689
Average degree	25.6	19.6	14.6
Network Diameter	15	11	7
Global clustering coefficient	0.148	0.138	0.141
Average edge weight	0.0271	0.0285	0.0076

4.6 Experiments

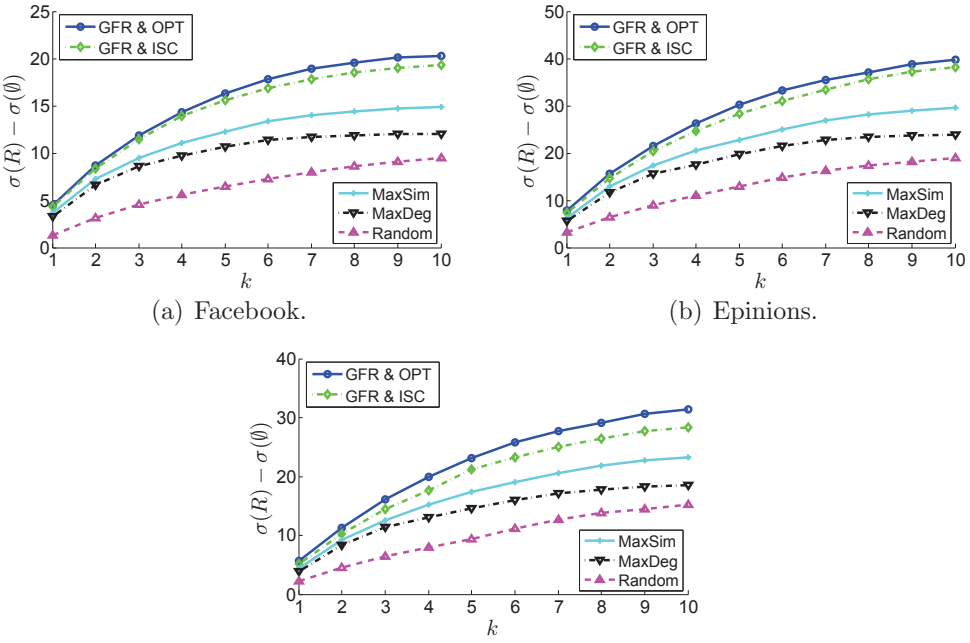
4.6.1 Dataset Information

Our experiments use three datasets: Facebook [79], Epinions [80], and Wiki [81]. Facebook is an online social networking service launched in 2004. Users on Facebook can create a user profile, add other users as friends, post status updates and photos, share videos, and receive notifications when others update their profiles. The Facebook "People You May Know" feature recommends new friends based on a friend-of-a-friend strategy [56]. Facebook also provides business page services [60] to users for social influence maximizations, meaning that our approach can be potentially applied on Facebook. Epinions is a general consumer review site launched in 1999. Epinions users could read new and old reviews about a variety of products to help them decide on a purchase. Our approach can be applied on users who want to disseminate their product reviews [82]. Wiki is a free encyclopedia written collaboratively by volunteers (i.e., users). A small portion of users are administrators. In order for a user to become an administrator, a request must be issued and voted. A directed edge is represented by that a user votes for another user. Our approach can be applied on users who want to get more votes. Note that these three datasets do not include information on edge weights. Following existing works [62, 65, 63, 64], we use directed common neighbor similarities [62] to determine edge weights. Finally, the statistics of these three datasets are shown in Table 4.1.

4.6.2 Comparison Algorithms

Our experiments focus on the increased influence spread brought by the recommendations, under different settings (e.g., numbers of recommended people, values of α_v and β_v, and outgoing degrees of v_0). The increased influence spread can be computed by $\sigma(R) - \sigma(\emptyset)$. Note that $\sigma(\emptyset)$ is not necessarily 0, since v_0 can still influence some people through existing friends. For comparison, six algorithms are included. (i) GFR & OPT stands for Algorithm 3 with the optimal influence spread computation. The influence spread is computed through time-consuming Monte-Carlo simulations. GFR & OPT guarantees an approximation ratio of $1 - \frac{1}{e}$ to the optimal algorithm. (ii) GFR & ISC is Algorithm 3 with a non-optimal influence spread computation. The influence spread is computed through Algorithm 4 (we set $\varepsilon = 0.1$ by default). We want to check the gap between GFR & OPT and GFR & ISC. (iii) MaxSim is for a greedy algorithm that iteratively selects a user with a maximum common neighbor similarity (in terms of v_0). It is currently used on Facebook [56]. Recommended people are friends-of-friends of v_0. (iv) MaxDeg stands for a greedy algorithm that iteratively selects a user with a maximum outgoing degree. Recommended people have the highest outgoing degrees. (v) Random is a baseline algorithm that recommends friends uniform-randomly. All these algorithms are implemented in a centralized manner.

We also set up experiments to compare different algorithms for the influence spread computation. For comparison, five algorithms are included, as described in the following. (i) Tree [63]. This algorithm computes the influence spread through a maximum influence arborescence model, where influences are restricted to propagate along maximum influence paths. (ii) DAG [64]. This algorithm reduces the graph to a DAG to compute the influence spread. The influence probabilities are sequentially determined following a topological order in the DAG. (iii) IMRank



(a) Facebook.

(b) Epinions.

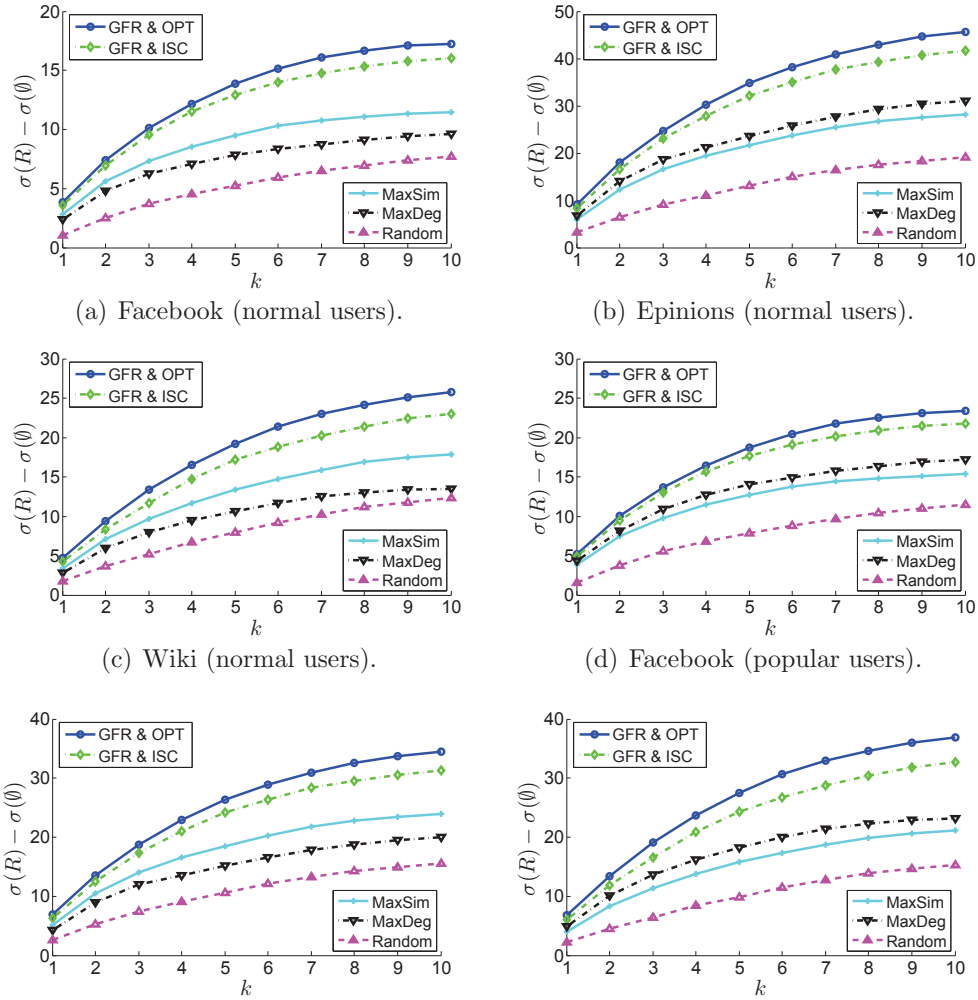
(c) Wiki.

Figure 4.7: The evaluation results on the impact of k (the number of recommended friends).

[75]. This algorithm computes the influence spread through iterative neighborhood exchanges. The probability that a given node is influenced depends on that of its neighbors. The drawback is that IMRank ignores path dependency issues. (iv) ISC denotes Algorithm 4, which computes the influence spread through considering additional paths. (v) OPT. This algorithm optimally computes the influence spread by time-consuming Monte-Carlo simulations [83].

4.6.3 Evaluation Results on Friend Recommendations

In this subsection, we report the evaluation results on friend recommendations. First, we investigate the impact of k (i.e., the number of recommended friends). We set $\alpha_v = 0.9$ and $\beta_v = 0.1$ for all people. v_0 is uniform-randomly picked from all the nodes. The results are averaged over 10,000 times for smoothness and are shown in Fig. 4.7. Three subfigures represent results in three different datasets, respectively. It



(a) Facebook (normal users).

(b) Epinions (normal users).

(c) Wiki (normal users).

(d) Facebook (popular users).

(e) Epinions (popular users).

(f) Wiki (popular users).

Figure 4.8: The evaluation results on the impact of v_0 (normal users or popular users).

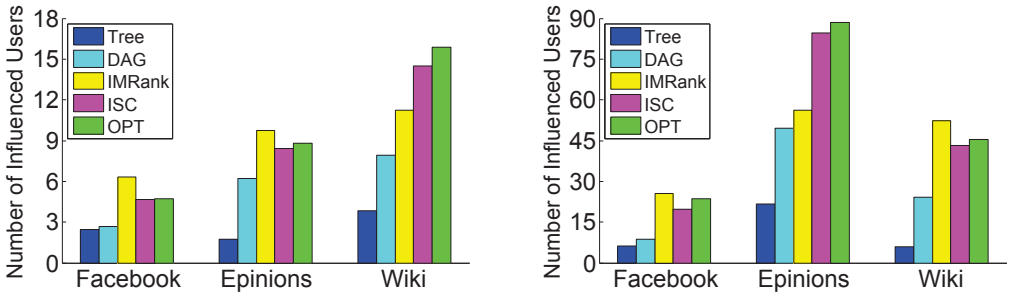
can be seen that GFR & OPT performs better than GFR & ISC, since the former one computes the influence spread optimally at the cost of time consumption. However, the gap between their performances is limited, meaning that GFR & ISC estimates the influence spread accurately. Meanwhile, MaxSim outperforms MaxDeg, meaning that we are more likely to recommended friends-of-friends than strangers (users who are not friends or friends-of-friends). For all the algorithms, the increased influence spread, $\sigma(R) - \sigma(\emptyset)$, satisfies the diminishing return effect with respect to k. This is because people influenced by later recommendations have potential overlays with those influenced by earlier recommendations.

We also study the impact of v_0. For convenience, we classify users into two types by their outgoing degrees. If a user has an outgoing degree that is no larger than 100, it is called a normal user. Otherwise, it is called a popular user. We would like to see the difference between recommendations for normal users and those for popular users, as shown in Fig. 4.8. An interesting phenomenon is that, our recommendations are more effective for popular users than normal users on Facebook and Wiki, but are less effective for popular users than normal users in Epinions. This is because popular users are already influential in Epinions. We also observe that, for popular users, MaxDeg outperforms MaxSim in Facebook and Wiki, while MaxSim outperforms MaxDeg in Epinions. A possible explanation for the above phenomenon is that, when we intend to recommend influential strangers over friends-of-friends (MaxDeg outperforms MaxSim), users could greatly improve their social influence, since their influences are no longer limited to their current social circles. Recommendations for popular users are not necessarily more effective than those for normal users, depending on the application scenario.

4.6.4 Evaluation Results on Influence Spread Computations

This subsection evaluates the influence spread computation. v_0 is uniform-randomly picked from all the nodes. We use $k = 0$, i.e., the result is the number of people that v_0 can influence through existing friends. Users are classified into normal users and popular users, according to the same rule in the previous subsection. The impact of the control parameter, ε, is studied. For ISC, we set $\varepsilon = 0.1$ and $\varepsilon = 0.3$, respectively. They are denoted as ISC (0.1) and ISC (0.3).

The evaluation results are shown in Fig. 4.9. On average, popular users have much larger influence spreads than normal users. The algorithm of Tree significantly underestimates the influence spread. Although DAG improves Tree by considering



(a) Normal users. (b) Popular users.

Figure 4.9: Evaluations of influence spread computation methods.

more paths, it still underestimates the influence spread. The latest approach of IMRank is outperformed by ISC (0.1), since IMRank does not consider the number of paths in the influence spread computation. In contrast, ISC (0.1) has an accurate estimation of the true influence spread, through considering the dominating influence propagation paths. Errors of ISC (0.1) are within 10% for both normal users and popular users in the three datasets. The impact of the control parameter is also significant. Note that ISC is expected to obtain a ratio, $1-\varepsilon$, to the optimal algorithm (under some assumptions in Eqs. 4.5-5 and 4.5-10). A smaller ε brings a more accurate computation of the influence spread. While $\varepsilon = 0.3$ brings an inaccurate influence spread computation, $\varepsilon = 0.1$ can bring a result that is close to the optimal algorithm.

4.7 Summary

This chapter studies the friend recommendation strategy with the perspective of social influence maximization. For the system provider, the objective is to recommend k new friends to a given user, such that the given user can maximize his/her social influence through new friends. Our problem is NP-hard. A greedy approximation algorithm with a ratio of $1 - \frac{1}{e}$ is proposed based on the submodular property. By considering the multipath effect, a novel method is proposed to compute the influence spread. Experiments verify the efficiency and effectiveness of our approach.

CHAPTER 5

SOCIAL INFLUENCE MAXIMIZATION IN HYPERGRAPHS: NON-SUBMODULARITY AND APPROXIMABILITY

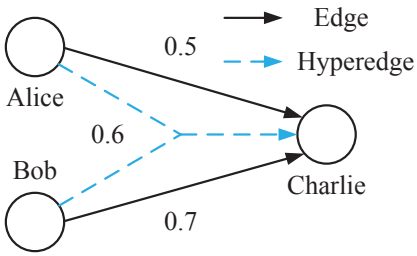
Chapters 4 and 5 focus on the influence propagation applications for social networks. More specifically, this chapter extends the Social Influence Maximization Problem (SIMP). SIMP aims to select k initially-influenced seed users to maximize the number of eventually-influenced users. Under the independent cascade model, the SIMP has been proved to be NP-hard, monotone, and submodular. Therefore, a naive greedy algorithm that maximizes the marginal gain obtains an approximation ratio of $1 - e^{-1}$. This chapter extends the SIMP by considering the crowd influence. This is because the crowd influence surpasses the combination of the independent influence from each person in the crowd. Our problem is proved to be NP-hard and monotone, but not submodular. It is proved to be inapproximable within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$. However, since user connections in OSNs are not random, approximations can be obtained by leveraging the structural properties of OSNs. The modularity, denoted by Δ, is proposed to measure to what degree our problem violates the submodularity. Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1 - e^{-1/(\Delta+1)}$, respectively. Experiments demonstrate the efficiency and effectiveness of our algorithms.

5.1 Introduction

Online Social Networks (OSNs) mainly focus on building social relationships among users who share interests, activities, backgrounds, stories, and real-life connections. OSNs are very valuable tools used by numerous people to extend their daily contacts. Existing OSNs such as Facebook and Twitter are two of the top ten most-visited webs in the world. As of January 2014, 74% of online adults use OSNs.

Motivated by many OSN applications such as viral marketing [84] and personalized recommendation [67], social influence propagations have received tremendous attention in the last decade, especially for the Social Influence Maximization Problem (SIMP). Given an influence propagation model such as the independent cascade model, the SIMP selects k initially-influenced seed users to maximize the number of eventually-influenced users [62]. In the literature, the influence propagation model is generally submodular [85]. Therefore, a simple greedy algorithm can obtain an approximation ratio of $1 - \frac{1}{e}$ to the optimal algorithm. However, few results [3] are provided when the influence propagation model even slightly violates the submodularity. In contrast, this chapter studies a non-submodular SIMP under the independent cascade model in hypergraphs.

Our key justification for the non-submodularity comes from the crowd psychology: most people do follow the crowd [86], and there are various reasons to explain why people do so. One reason is that people are afraid of doing anything new. Attempting new things always requires courage, since we do not know what will happen in the future. Following the crowd gives people a cushion of comfort to make mistakes. Another reason is because of criticism. Once we do something atypical, we may be criticised heavily by friends, family and parents for not doing what everyone else is doing. After we fail, we would start doing what the crowd does. As a result, the crowd psychology reveals that the crowd influence is different



Influenced?		Probability to propagate the influence to Charlie
Alice	Bob	
Yes	No	0.5
No	Yes	0.7
Yes	Yes	1-(1-0.5)(1-0.6)(1-0.7)

Figure 5.1: Social influences through edges and hyperedges.

from the combined independent influences of people in the crowd. This phenomenon yields non-submodularity in social influence propagations, which are modeled through hypergraphs.

Fig. 5.1, which shows three people (Alice, Bob, and Charlie), provides a more specific example. Directed edges represent the influence from Alice or from Bob to Charlie. The influences Charlie receives from Alice and Bob are independent of each other. According to the crowd psychology, if both Alice and Bob are influenced, there should exist a crowd influence in addition to Alice's and Bob's influences. A hyperedge is used to depict such a crowd influence. Note that influences through hyperedges is not submodular, since seed user selections in the SIMP are no longer diminishing return. Consequently, unique challenges are posed to solve the SIMP in hypergraphs. The first challenge is to deal with the non-submodularity. The problem hardness and approximability need to be explored. New algorithms are needed, since a simple greedy algorithm can no longer guarantee an approximation ratio. Another challenge is the scalability. Since hyperedges change the scalability of the SIMP, it is difficult to reduce their complexities.

This chapter studies the SIMP under the independent cascade model in hypergraphs. The problem is proven to be NP-hard, and cannot be approximated within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$. Here, $|V|$ is the number of nodes in the hypergraph, meaning that no algorithm can do better than a random seed user selection in terms of the asymptotic approximation ratio. However, since

user connections in OSNs are not random, approximation algorithms are proposed by leveraging certain structural properties of OSNs. A concept called modularity (denoted by Δ) is proposed. Δ can measure to what degree our problem violates the submodularity, and Δ is expected to be bounded in OSNs. Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1-e^{-\frac{1}{\Delta+1}}$, respectively.

Our main contributions are summarized as follows:

- Motivated by the crowd psychology, we study the SIMP in hypergraphs. Our problem is proven to be NP-hard, monotone, non-submodular, and inapproximable.
- A new concept called modularity is proposed to measure to what degree hyperedges violate the submodularity. The modularity is expected to be bounded in OSNs.
- Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1-e^{-\frac{1}{\Delta+1}}$, respectively. Here, Δ is the modularity. Algorithms are scalable for large OSNs.
- Real data-driven experiments are conducted to evaluate the proposed solutions. The results are shown from different perspectives to provide insightful conclusions.

The remainder of this chapter is organized as follows. Section 5.2 surveys the related work. Section 5.3 describes the model, and then, formulates the problem. Section 5.4 demonstrates the NP-hardness, non-submodularity, and approximability of our problem. Section 5.5 proposes two approximation algorithms for the SIMP with the modularity. Section 5.6 includes experiments. Finally, Section 5.7 concludes the chapter.

5.2 Related Works

Motivated by applications such as viral marketing [84] and personalized recommendations [67], researches on the social influence propagation have received tremendous attention in the last decade, especially for the SIMP. The original SIMP was proposed by Kempe et al. [62] with two influence propagation models of independent cascade and linear threshold. The SIMP aims to select k initially-influenced seed users to maximize the number of eventually-influenced users. Under the independent cascade and linear threshold models, the SIMP has been proven to be NP-hard, monotone, and submodular. Consequently, a simple greedy algorithm, which iteratively maximizes the marginal gain, obtains an approximation ratio of $1-\frac{1}{e}$ to the optimal algorithm. A fruitful literature for the SIMP [87, 88, 89, 90] has been developed. However, almost all variations of the SIMP are submodular or unbounded. For example, Chen et al. [91] considered a variation of the SIMP with both positive and negative influence propagations. Their model maintains submodularity for maximizing the spread of positive influences. Unbounded variations of the SIMP are studied, usually through a data mining approach. For example, Goyal et al. [92] used available traces to learn how influence propagates in OSNs. Based on the learned model, the expected influence spread can be estimated to solve the SIMP.

Unfortunately, few results [3] are provided when the influence propagation model even slightly violates the submodularity. Hung et al. [3] studied a variation of the SIMP with multiple items. Their problem is NP-hard and non-submodular, and thus, only heuristic algorithms are provided. This is because the problem of non-submodular function maximization [93] has not been perfectly solved in the literature [94]. Although the problem of supermodular function maximization can be optimally solved by the minimum-norm-point algorithm [95], non-submodular functions are not the same. The latest approach is based on the curvature [96], which assumes that the

marginal gain of the non-submodular function varies within a given curvature. This chapter can be viewed as a curvature-based approach that is specially designed for the SIMP in OSNs.

Additionally, this chapter relates to the structural properties of OSNs. Recent advances in network science show that user connections in OSNs are not truly random [97]. The degree distribution in OSNs is acknowledged to follow the power-law distribution [97]: a majority of users are inactive with a small number of connections, while a minority of users are active with a large number of connections. Based on [98], OSNs usually have small diameters (about 6), high clustering coefficients (larger than 0.1), and community structures. These structural properties can be incorporated into algorithmic designs. For example, Wang et al. [99] proposed a community-based greedy algorithm to find the most influential nodes in OSNs. This chapter leverages the structural properties of OSNs to solve the non-submodular SIMP in hypergraphs.

5.3 Model and Formulation

5.3.1 Model and Notations

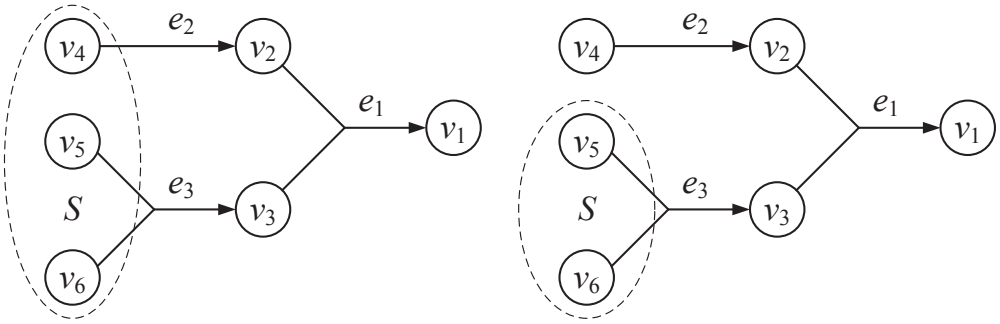
Our scenario is based on a directed hypergraph $G = (V, E)$, where $V = \{v\}$ is a set of nodes (i.e., users in an OSN), and $E = \{e\}$ is a set of directed hyperedges. Hyperedges represent influence propagation directions, including personal and crowd influences. $|\cdot|$ denotes the set cardinality: $|V|$ and $|E|$ are the numbers of nodes and hyperedges, respectively. For a hyperedge e, let H_e and T_e denote its head and tail sets of nodes (i.e., e connects nodes in H_e to nodes in T_e). Hyperedges are a generalization of normal edges. As a special case, when $|H_e| = |T_e| = 1$, e becomes a normal edge. Let w_e denote the weight of e, representing the influence propagation probability ($0 \leq w_e \leq 1$). Given an OSN, the hypergraph G can be generated using Hung's approach [3]: while nodes are just users in the OSN, hyperedges and their

weights can be learned based on a statistical inference-based framework. Therefore, this chapter assumes that G is known a priori.

5.3.2 Independent Cascade in Hypergraphs

The independent cascade is a classic model [62] that simulates influence propagations in OSNs. Since the independent cascade is designed for normal graphs, a simple extension is made for hypergraphs. Let us start with a set, S, of nodes. Nodes in S are initially active and are also called seed users [62]. In contrast, all other nodes are initially inactive. Independent cascade unfolds in discrete steps according to the following randomized process. Given a hyperedge e, when all nodes in H_e first becomes all active in step t, attempts are made to activate each inactive node in T_e. Each activation attempt is independent of all others, and it succeeds with a probability w_e. Here, w_e represents the influence propagation probability. If an inactive node has received multiple activation attempts, these activation attempts can be sequenced in an arbitrary order. If an inactive node is successfully activated, it will become active in step $t+1$. In addition, whether or not an activation attempt succeeds, it will have no further impacts in subsequent steps. The above process iterates step by step and terminates when no more activations are possible.

Active nodes in the independent cascade model represent influenced nodes. All seed nodes in S are initially-influenced, while all active nodes at the end of the process are eventually-influenced. Note that a hyperedge e could only propagate the influence when all nodes in H_e first become all active. If H_e includes an inactive node, e cannot propagate the influence. This is because hyperedges represent crowd influences. We use $\sigma(S)$ to denote the expected number of eventually-influenced nodes. $\sigma(S)$ is also called the influence spread of S.



(a) v_1 is reachable from S. (b) v_1 is not reachable from S.

Figure 5.2: An example of the reachability.

5.3.3 Problem Formulation

Our objective is to select k initially-influenced seed users to maximize the number of eventually-influenced users:

$$\text{maximize} \quad \sigma(S) \tag{5.3-1}$$

$$\text{s.t.} \quad |S| \leq k \tag{5.3-2}$$

k is a pre-defined constant to bound the size of the seed set S. Given S, $\sigma(S)$ is determined by the independent cascade model. Our problem is almost the same as the classic SIMP in [62], except that our problem uses a hypergraph rather than a normal graph. However, this difference, despite how small it seems, leads to unique challenges. This is because the SIMP has become non-submodular instead of submodular.

For presentation simplicity, four concepts (i.e., hyperdegree, cycle, path, and reachability) are defined:

Definition 5.1. Let d_v denote the hyperdegree of the node v. d_v is the number of hyperedges that include v in their heads or tails, i.e., $d_v = |\{e \,|\, v \in H_e \text{ or } v \in T_e\}|$.

Definition 5.2. For a directed hypergraph $G = (V, E)$, a cycle in G is defined as a sequence $(v_0, e_0, v_1, e_1, ..., e_{n-1}, v_0)$ of alternating nodes and hyperedges where (i) e_i is distinct for $\forall i \in \{0, ..., n\text{–}1\}$, (ii) $v_i \in H_{e_i}$ for $\forall i \in \{0, ..., n\text{–}1\}$, and (iii) $v_{(i+1)\%n} \in T_{e_i}$

for $\forall i \in \{0, ..., n\text{–}1\}$.

Definition 5.3. For a directed hypergraph $G = (V, E)$, a path from v_0 to v_n in G is a sequence $(v_0, e_0, v_1, e_1, ..., e_{n-1}, v_n)$ of alternating nodes and hyperedges where (i) e_i is distinct for $\forall i \in \{0, ..., n\text{–}1\}$, (ii) $v_i \in H_{e_i}$ for $\forall i \in \{0, ..., n\text{–}1\}$, and (iii) $v_{i+1} \in T_{e_i}$ for $\forall i \in \{0, ..., n\text{–}1\}$.

Definition 5.4. For a directed hypergraph $G = (V, E)$, a node v is said to be reachable from a node set S if (i) there exists a path from a node in S to v and (ii) each head node of each hyperedge of the above path is reachable from S.

Note that the reachability is recursively defined. An example is shown in Fig. 5.2. In Fig. 5.2(a), v_1 is reachable from S since v_2 and v_3 are also reachable from S. In contrast, in Fig. 5.2(b), v_1 is not reachable from S since v_2 is not reachable from S.

5.4 Analysis

This section mainly analyzes the SIMP in a hypergraph. Our problem is proven to be NP-hard, monotone, non-submodular, and inapproximable within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$.

5.4.1 NP-hard and Monotone

We start with the problem hardness:

Theorem 5.5. *The SIMP in a hypergraph is NP-hard.*

This is because the classic SIMP in a normal graph [62] is NP-hard by reduction from the set cover problem, which is NP-complete. Meanwhile, every instance in the classic SIMP is a special case of our problem (a graph is a special case of a hypergraph). Therefore, the SIMP in a hypergraph is NP-hard.

Theorem 5.6. *Given a hypergraph G, $\sigma(S)$ is monotone with respect to S, meaning*

that $\sigma(S') \leq \sigma(S)$ *for* $\forall S' \subseteq S$.

Proof: We prove through formulating an equivalent view of the independent cascade. Let us convert the hypergraph G to another hypergraph G'. G and G' have the same nodes. Each hyperedge, e, in G is mapped to a set, $\{e'\}$, of hyperedges in G': $\{e' \,|\, w_e = w_{e'}, H_{e'} = H_e, |T_{e'}| = 1, T_{e'} \subseteq T_e\}$. Such a conversion decomposes hyperedges in G by separating their tail nodes. By definition, the independent cascades in G and G' should be exactly the same.

In G', let us consider a hyperedge e' whose head nodes first become all active. Note that e' has exactly one tail node by definition. Now e' tries to activate its tail node, succeeding with probability $w_{e'}$. We can view the outcome of this random event as being determined by flipping a coin of bias $w_{e'}$. From the view of the independent cascade, it does not matter whether the coin was flipped at the moment that e' tries to activate its tail node, or whether it was flipped at the very beginning of the whole process and is only being revealed now. Continuing this reasoning, we can assume that for each hyperedge $e' \in G'$, a coin of bias $w_{e'}$ is flipped at the very beginning of the process (independently of the coins for all other hyperedges), and the result is stored so that it can be checked later when e' tries to activate its tail node.

The remainder is similar to [62]. With all the coins flipped in advance, the independent cascade in G' can be equivalently viewed as follows. In G', the hyperedges, for which the coin flip indicated an activation will be successful, are declared to be live; the remaining hyperedges are declared to be blocked. If we fix the outcomes of the coin flips and then initially activate a seed set S, it is clear how to determine the full set of active nodes at the end of the independent cascade: a node v ends up active if and only if it is reachable (see Definition 5.4) from S via only live hyperedges. In each possible set of outcomes (in terms of all coin flip outcomes on the hyperedges), if a node v is reachable from S', it must be reachable from S, since

$S' \subseteq S$. Therefore, $\sigma(S') \leq \sigma(S)$ holds for $\forall S' \subseteq S$, and the proof completes. ■

The monotonicity for the influence propagation in a normal graph is preserved in a hypergraph. The intuition is that more people always bring more influences.

5.4.2 Non-Submodular and Inapproximability

Theorem 5.7. *Given a hypergraph G, $\sigma(S)$ is not submodular with respect to S, meaning that $\sigma(S \cup \{v\}) - \sigma(S) > \sigma(S' \cup \{v\}) - \sigma(S')$ for $\exists\, v \in V, S' \subset S, S \subseteq V$.*

Proof: We prove by a counterexample in Fig. 5.2(a). We focus on three nodes of v_1, v_2, v_3, and one hyperedge of e_1 with $w_1 = 1$. Let us set $S' = \emptyset$, $S = \{v_2\}$, and $v = v_3$. We have $\sigma(S \cup \{v\}) = 3$, since v_1 will be influenced by v_2 and v_3. Meanwhile, we have $\sigma(S) = 1$ and $\sigma(S' \cup \{v\}) = 1$, since not all head nodes of e_1 are active, and thus, e_1 cannot activate v_1. In addition, we have $\sigma(S') = 0$ since $S' = \emptyset$. As a result, $\sigma(S \cup \{v\}) - \sigma(S) = 2 > \sigma(S' \cup \{v\}) - \sigma(S') = 1$. ■

The submodularity for the influence propagation in a normal graph is not preserved in a hypergraph. The intuition is that the SIMP is no longer diminishing return due to the crowd influence, which is in addition to the influence of each person in the crowd. This phenomenon yields non-submodularity. A simple variation of Hung's first theorem in [3] can lead to the following approximability result:

Theorem 5.8. *For the SIMP in a general hypergraph, unless $\mathbb{P} = \mathbb{NP}$, no algorithm can guarantee an approximation ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$.*

Theorem 5.8 validates that the SIMP in a general hypergraph is not approximable, i.e., any given algorithm must perform poorly under certain hypergraphs. Therefore, Theorem 5.8 poses a unique challenge to solve the SIMP in OSNs.

Algorithm 5 Naive Greedy (NG)

Input: a hypergraph, G, and a constant, k.
Output: a set of seed nodes, S.

1: Initialize $i = 0$ and $S_0 = \emptyset$.
2: **while** $|S_i| < k$ **do**
3: Find out $v = \arg\max_{v \in V} \sigma(S_i \cup \{v\}) - \sigma(S_i)$.
4: Update $S_{i+1} = S_i \cup \{v\}$.
5: Update i to be $i + 1$.
6: **return** $S = S_i$ as the set of seed nodes.

5.4.3 Naive Greedy

Since Theorem 5.8 shows the inapproximability of the SIMP in a general hypergraph, this subsection focuses on a naive greedy algorithm, as shown in Algorithm 5. Starting with an empty seed set (line 1), it iteratively add a node that maximizes the marginal gain of $\sigma(S_i)$, until k nodes are selected (lines 2 to 5). Since $\sigma(\cdot)$ is no longer submodular, such a greedy algorithm cannot guarantee an approximation ratio of $1 - 1/e$. Note that Algorithm 5 involves the sub-problem of computing $\sigma(S)$ for a given S. This sub-problem is NP-hard [63] and has been extensively studied in the literature. This chapter does not explore this sub-problem, and uses the Monte Carlo simulation [63] to compute $\sigma(S)$ for a given S in G.

5.5 Algorithms

The previous section has proven that the SIMP in a general hypergraph is NP-hard, monotone, non-submodular, and not approximable within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$. However, OSNs are not general hypergraphs. Recent studies in network science validate that user connections in OSNs are not truly random [97]. Consequently, approximation algorithms become possible by leveraging certain structural properties of OSNs.

5.5.1 Modularity

To enable approximation algorithms by revealing structural properties of OSNs, two concepts are proposed:

Definition 5.9. Given a monotone objective function $\sigma(\cdot)$, the modularity set of a node v is $M_v = \{v' \,|\, \sigma(S \cup \{v, v'\})\text{–}\sigma(S \cup \{v'\}) > \sigma(S \cup \{v\})\text{–}\sigma(S)$ for $\exists\, v \in V, v' \in V, S \subseteq V\}$, which includes all nodes that might increase the marginal gain of v.

Definition 5.10. The modularity, Δ, is defined as the maximum cardinality among all modularity sets, i.e., $\Delta = \max_v |M_v|$.

For a node v, only nodes in M_v might increase the marginal gain of v for the objective function $\sigma(\cdot)$. In contrast, nodes that are not in M_v can never increase the marginal gain of v. If v is locally submodular for $\sigma(\cdot)$, then $M_v = \emptyset$. Consequently, the modularity Δ measures the degree to which $\sigma(\cdot)$ violates the submodularity. $\sigma(\cdot)$ gets closer to the submodularity for a smaller Δ, and is submodular when $\Delta = 0$.

For a general hypergraph, Δ is not bounded, and can be as large as $O(|V|)$. This is the reason that SIMP in a general hypergraph is not approximable. However, recent studies in network science show that OSNs are scale-free networks [97], meaning that Δ can be bounded in OSNs. Therefore, the next subsection will discuss the bound of Δ in scale-free OSNs.

5.5.2 OSNs as Scale-Free Hypergraphs

Recent studies in network science show that OSNs are scale-free networks [97], meaning that the degree distribution in an OSN follows the power-law distribution [100]. Let p_d denote the fraction of nodes with a hyperdegree d. The power-law means that $p_d = (\gamma - 1)d^{-\gamma}$, in which γ ranges from 2 to 4 in OSNs [97]. Let $\overline{w} = \frac{1}{|E|}\sum_e w_e$ denote the average weight of the hyperedges. We have the following theorem:

Theorem 5.11. *In scale-free OSNs with γ and $\overline{w}$, it is expected to have $\Delta \in o(|V|)$*

when $4 + 6\overline{w}\frac{\gamma-1}{\gamma-2} \leq 3(\frac{\gamma-1}{\overline{w}^{\gamma+1}})^2$.

Proof: Similar to the proof of Theorem 5.6, we again form an equivalent view of the independent cascade to compute Δ. For each hyperedge e, coins are flipped based on its weight, w_e. Let C_v be the maximum Weakly Connected Component (WCC) containing v via live hyperedges in G after the coin flip. Here, two nodes are weakly connected if there exists a path connecting them (see Definition 5.3), when each hyperedge is regarded as bi-directional. We claim that $M_v \subseteq C_v$. This is because nodes outside C_v cannot increase the marginal gain of v. Consequently, we can conclude that Δ is upper-bounded by the size of the maximum WCC via live hyperedges in G.

The following part of the proof uses Molloy's Theorem 1 in [101] to derive the size of the maximum WCC through live hyperedges in G. All prerequisites of this theorem are satisfied. In addition, Molloy's Theorem 1 was developed under general graphs, but it can be applied to hypergraphs through separating each hyperedge into a set of normal edges. Let q_d denote the fraction of nodes with a live-hyperdegree d (live-hyperdegree is the hyperdegree that only counts live hyperedges). We have:

$$q_d = \frac{1}{\overline{w}} p_{d/\overline{w}} = \frac{\gamma-1}{\overline{w}} \Big(\frac{d}{\overline{w}}\Big)^{-\gamma} \tag{5.5-3}$$

Let $\overline{d} = \overline{w}\frac{\gamma-1}{\gamma-2}$ be the average live-hyperdegree. We define:

$$\begin{aligned}
\chi(\alpha) &= \overline{d} - 2\alpha - \sum_{d=1}^{\infty} dq_d \Big(1 - \frac{2\alpha}{\overline{d}}\Big)^{\frac{d}{2}} \approx \overline{d} - 2\alpha - \int_{d=1}^{\infty} dq_d \Big(1 - \frac{2\alpha}{\overline{d}}\Big)^{\frac{d}{2}} \mathrm{d}d - \psi \\
&= \overline{d} - 2\alpha - \frac{\gamma-1}{\overline{w}^{\gamma+1}} \Big[\frac{2d^{1-\gamma}(1-\frac{2\alpha}{\overline{d}})^{\frac{d}{2}+1}}{d+2}\Big]\Big|_{d=1}^{\infty} - \psi \\
&= \overline{d}(1 - \frac{2\alpha}{\overline{d}}) + \frac{\gamma-1}{\overline{w}^{\gamma+1}} \times \frac{2}{3}(1-\frac{2\alpha}{\overline{d}})^{\frac{3}{2}} - \psi
\end{aligned} \tag{5.5-4}$$

Eq. 5.5-4 is derived under $0 \leq 2\alpha \leq \overline{d}$. Eq. 5.5-4 does not consider the case of $d < 1$

(no impact on the graph connectivity). Here, ψ comes from Euler-Maclaurin formula for integral boundaries:

$$\psi = \frac{1}{2} \times [\frac{\gamma-1}{\overline{w}^{\gamma+1}}(1-\frac{2\alpha}{\overline{d}})^{\frac{1}{2}} + 0] = \frac{1}{2}\frac{\gamma-1}{\overline{w}^{\gamma+1}}(1-\frac{2\alpha}{\overline{d}})^{\frac{1}{2}} \tag{5.5-5}$$

Let α_D be the smallest positive root for $\chi(\alpha) = 0$. We have the following result for α_D and definition for ϵ_D:

$$(1-\frac{2\alpha_D}{\overline{d}})^{\frac{1}{2}} = \frac{\sqrt{12(\frac{\gamma-1}{\overline{w}^{\gamma+1}})^2 + 9\overline{d}^2} - 3\overline{d}}{4\frac{\gamma-1}{\overline{w}^{\gamma+1}}} \tag{5.5-6}$$

$$\begin{aligned}\epsilon_D &= 1 - \sum_d q_d (1-\frac{2\alpha_D}{\overline{d}})^{\frac{d}{2}} \leq 1 - q_1(1-\frac{2\alpha_D}{\overline{d}})^{\frac{1}{2}} \\ &= \frac{(4+3\overline{d}) - \sqrt{12(\frac{\gamma-1}{\overline{w}^{\gamma+1}})^2 + 9\overline{d}^2}}{4}\end{aligned} \tag{5.5-7}$$

We can have $\epsilon_D \leq 0$ when $4 + 6\overline{w}\frac{\gamma-1}{\gamma-2} \leq 3(\frac{\gamma-1}{\overline{w}^{\gamma+1}})^2$. Molloy's Theorem 1 in [101] proved that the size of the maximum WCC via live hyperedges in G (the size of a giant component in a random graph) is $\epsilon_D|V| + o(|V|)$, or just $o(|V|)$ when $\epsilon_D \leq 0$. Since Δ is upper-bounded by the size of the maximum WCC via live hyperedges in G, the proof completes. ■

The insight of Theorem 5.11 is that the influence propagation from a node decays quickly with respect to $\overline{w}$. The influence of a node becomes limited when $\overline{w}$ is small (we have $\Delta = 0$ when $\overline{w} = 0$). Note that $4 + 6\overline{w}\frac{\gamma-1}{\gamma-2} \leq 3(\frac{\gamma-1}{\overline{w}^{\gamma+1}})^2$ is satisfied for most OSNs that have $\overline{w} < 0.7$ and $\gamma > 2.1$ [63]. Therefore, the SIMP in OSNs is approximable. In addition, γ has a big impact on Δ. When γ is smaller, the hyperdegree distribution is closer to "uniform," and Δ is larger. On the other hand, when γ is larger, fewer nodes have large hyperdegrees and Δ is smaller. A smaller Δ represents a smaller gap from the submodularity. The following subsections will use the modularity techniques [102] to approximate the SIMP with Δ in OSNs.

Algorithm 6 Improved Greedy (IG)

Input: a hypergraph, G, and a constant, k.
Output: a set of seed nodes, S.

1: Initialize $i = 0$ and $S_0 = \emptyset$.
2: **while** $|S_i| < k$ **do**
3: Find out $\arg\max_{v \in V, M'_v \subseteq M_v} \sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i)$ constrained by $|S_i \cup \{v\} \cup M'_v| \leq k$.
4: Update $S_{i+1} = S_i \cup \{v\} \cup M'_v$.
5: Update i to be $i + 1$.
6: **return** $S = S_i$ as the set of seed nodes.

5.5.3 Improved Greedy

By leveraging the structural properties of OSNs, Δ is shown to be bounded in OSNs (Theorem 5.11). Consequently, approximation algorithms become possible. The key idea is that, when the node v is selected as a seed node, nodes in M_v should be further considered, since they can improve v's influence propagations. This observation can improve Algorithm 5.

Consequently, Algorithm 6 is proposed as another greedy algorithm. In line 1, it initializes $i = 0$ and $S_0 = \emptyset$. Lines 2 to 5 describe greedy iterations. While Algorithm 5 iteratively selects one seed node, Algorithm 6 iteratively selects a set of seed nodes, in order to mitigate the negative impact resulting from the non-submodularity. In line 3, once v is selected as a seed node, partial nodes in M_v (denoted as M'_v) are jointly selected as seed nodes. The greedy criterion is that v and M'_v can maximize the marginal gain of the current seed set, i.e., maximize $\sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i)$. The constraint is that $|S_i \cup \{v\} \cup M'_v| \leq k$, i.e., at most k seeds are selected. Lines 4 and 5 update the seed set S_i and the index i. The greedy iteration terminates once k seed nodes are selected.

Computing $\sigma(S)$ for a given S is considered to take $O(|E|)$ in the Monte Carlo simulation [63]. Algorithm 6 has at most k greedy iterations, and each iteration it exhausts v and M_v in line 3. Consequently, the time complexity of Algorithm 6 is

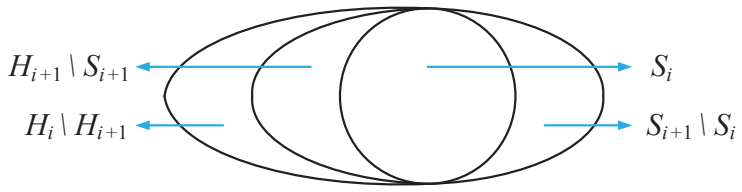


Figure 5.3: Relationship between S_i and H_i. We have $S_{i+1} = (S_{i+1} \setminus S_i) \cup S_i$, $H_{i+1} = (H_{i+1} \setminus S_{i+1}) \cup S_{i+1}$, and $H_i = (H_i \setminus H_{i+1}) \cup (H_{i+1} \setminus S_{i+1}) \cup S_i$.

$O(2^{\Delta}k|V||E|)$. We claim that Algorithm 6 is bounded:

Theorem 5.12. *Algorithm 6 has an approximation ratio of* $\frac{1}{\Delta+2}$ *to the optimal algorithm.*

Proof: Let S^* denote the optimal set of seed nodes, in terms of maximizing $\sigma(\cdot)$. An auxiliary parameter, H_i, is used. With $H_0 = S^*$, H_i is recursively defined as an arbitrary subset of $H_{i-1} \cup S_i$, under the constraint that $S_i \subseteq H_i$ and $|H_i| = k$. Intuitively, H_i consists of S_i and a part of S^*. The relationship between S_i and H_i is shown in Fig. 5.3. When i becomes larger, nodes from S_i are added to H_i, and nodes in S^* are removed from H_i, maintaining $|H_i| = k$. By definition, we have:

$$|H_{i+1}| = |H_i| - |H_i \setminus H_{i+1}| + |S_{i+1} \setminus S_i| \tag{5.5-8}$$

Since $|H_{i+1}| = |H_i| = k$, we have:

$$|H_i \setminus H_{i+1}| = |S_{i+1} \setminus S_i| \leq |\{v\} \cup M'_v| \leq 1 + \Delta \tag{5.5-9}$$

This is because $M'_v \subseteq M_v$ and $\Delta = \max_v |M_v|$. Eq. 5.5-9 means that, in each greedy iteration, at most $1 + \Delta$ nodes in S^* are ignored by Algorithm 6.

We claim that the marginal gain in each greedy iteration of Algorithm 6 has a lower bound with respect to $\sigma(H_i)$:

$$\sigma(H_i) - \sigma(H_{i+1}) \leq (\Delta+1) \times [\sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i)] \tag{5.5-10}$$

To prove Eq. 5.5-10, let us order nodes of $H_i \setminus H_{i+1}$ in an arbitrary order (say $v_1, v_2, ..., v_l$), and let $H_i^j = H_i \setminus \{v_1, v_2, ..., v_j\}$ for $1 \leq j \leq l$ ($H_i^0 = H_i$ and $H_i^l \subseteq H_{i+1}$). For each j, we have:

$$\begin{aligned}
&\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap H_i^j)) - \sigma(S_i) \\
\geq &\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap H_i^j)) - \sigma(S_i \cup (M_{v_j} \cap H_i^j)) \\
\geq &\sigma(S_i \cup \{v_j\} \cup H_i^j) - \sigma(S_i \cup H_i^j) \qquad (5.5\text{-}11)
\end{aligned}$$

The first inequality is from the monotonicity in Theorem 5.6, since $S_i \subseteq S_i \cup (M_{v_j} \cap H_i^j)$ and $\sigma(S_i) \leq \sigma(S_i \cup (M_{v_j} \cap H_i^j))$. The second inequality is from the definition of the modularity set, because only nodes in M_{v_j} can increase the marginal gain of v_j. Hence, nodes in $H_i^j \setminus M_{v_j}$ might decrease the marginal gain of v_j. By accumulating Eq. 5.5-11 among j, we obtain:

$$\begin{aligned}
&\sum_{j=1}^{l} [\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap H_i^j)) - \sigma(S_i)] \\
\geq &\sum_{j=1}^{l} [\sigma(S_i \cup \{v_j\} \cup H_i^j) - \sigma(S_i \cup H_i^j)] \\
= &\sigma(S_i \cup H_i^0) - \sigma(S_i \cup H_i^l) \geq \sigma(H_i) - \sigma(H_{i+1}) \qquad (5.5\text{-}12)
\end{aligned}$$

The first inequality is from Eq. 5.5-11. The equality results from the definition of H_i^j, since $\{v_j\} \cup H_i^j = H_i^{j-1}$. The last inequality is because $\sigma(S_i \cup H_i^0) = \sigma(H_i)$ and $\sigma(S_i \cup H_i^l) \leq \sigma(H_{i+1})$. We have $\sigma(S_i \cup H_i^0) = \sigma(H_i)$, since $H_i^0 = H_i$ and $S_i \subseteq H_i$. We have $\sigma(S_i \cup H_i^l) \leq \sigma(H_{i+1})$ by the monotonicity, since $S_i \subseteq S_{i+1} \subseteq H_{i+1}$ and $H_i^l \subseteq H_{i+1}$. We have:

$$\begin{aligned}
&(\Delta + 1) \times [\sigma(S_i \cup \{v\} \cup M_v') - \sigma(S_i)] \geq \sum_{j=1}^{l} [\sigma(S_i \cup \{v\} \cup M_v') - \sigma(S_i)] \\
\geq &\sum_{j=1}^{l} [\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap H_i^j)) - \sigma(S_i)] \geq \sigma(H_i) - \sigma(H_{i+1}) \qquad (5.5\text{-}13)
\end{aligned}$$

The first inequality results from Eq. 5.5-9, in which $1 \leq j \leq l = |H_i \setminus H_{i+1}| \leq \Delta + 1$. The second inequality comes from line 3 in Algorithm 6, which always selects the maximum marginal gain in each greedy iteration. The third inequality comes from Eq. 5.5-12. Therefore, Eq. 5.5-10 is valid.

Since the marginal gain in each greedy iteration of Algorithm 6 has a lower bound, we can accumulate Eq. 5.5-10 among all greedy iterations (note that $S_{i+1} = S_i \cup \{v\} \cup M'_v$):

$$\sigma(H_0) - \sigma(H_i) \leq (\Delta + 1) \times [\sigma(S_i) - \sigma(S_0)] \tag{5.5-14}$$

Since $H_0 = S^*$, $H_i = S_i = S$ when Algorithm 6 terminates, and $S_0 = \emptyset$, we have $\sigma(S^*) \leq (\Delta + 2) \times \sigma(S)$. ■

The key insight of Theorem 5.12 is that Algorithm 6 ignores at most $1 + \Delta$ nodes in the optimal set of seed nodes, resulting in a bounded marginal gain for each greedy iteration. Theorem 5.12 does not violate the inapproximability in Theorem 5.8, since Δ can be as large as $\Theta(|V|)$ in a general hypergraph. However, Algorithm 6 still has a critical drawback. Although Theorem 5.11 validates that $\Delta \in o(|V|)$ in OSNs, Δ may still be numerically large, in terms of the time complexity and the approximation ratio. As a result, Algorithm 6 may perform poorly in an OSN with a small γ, especially when γ gets closer to 2.

5.5.4 Capped Greedy

Since Δ has a critical impact on Algorithm 6, we need to further identify its role in the algorithm design. The key idea is that, although $|M_v|$ could be large, not all nodes in M_v have huge impacts on the marginal gain of v for $\sigma(\cdot)$. Intuitively, only v's neighbors, who share hyperedges with large weights, are important in M_v. Moreover, the optimal set of seed nodes are not able to include all nodes in M_v if $|M_v| > k$.

Algorithm 7 Capped Greedy (CG)

Input: a hypergraph, G, and a constant, k.
Output: a set of seed nodes, S.

1: Initialize $S = \emptyset$.
2: **for** each node $v' \in V$, each Δ' from 1 to Δ, and each set $S_0 \subseteq M_{v'}$ constrained by $|S_0| = k \mod (\Delta' + 1)$ **do**
3: Initialize $i = 0$.
4: **while** $|S_i| < k$ **do**
5: Find $\arg\max_{v \in V, M'_v \subseteq M_v} \sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i)$ constrained by $|S_i \cup \{v\} \cup M'_v| \leq k$ and $M'_v \leq \Delta'$.
6: Update $S_{i+1} = S_i \cup \{v\} \cup M'_v$.
7: Update i to be $i + 1$.
8: **if** $\sigma(S_i) > \sigma(S)$ **then**
9: Update S to be S_i.
10: **return** S as the set of seed nodes.

Therefore, capping the number of selected seed nodes in M_v might lead to a better performance, since low impact nodes in M_v can be replaced by high impact nodes outside M_v. To find out the best cap, we can simply exhaust all possible caps.

As a result, Algorithm 7 is proposed as an extension of Algorithm 6. In line 1, it initializes $S = \emptyset$. Line 2 includes a loop statement to exhaust all possible scenarios, in terms of the combination of each node $v' \in V$, each Δ' from 1 to Δ, and each set $S_0 \subseteq M_{v'}$ constrained by $|S_0| = k \mod (\Delta' + 1)$. Instead of Δ, Δ' is used as the cap. Lines 3 to 7 are basically the same as Algorithm 6, except for the cap. This part embeds Algorithm 6 to search the set of seed nodes in each possible scenario. The cap is added at the end of line 5 ($M'_v \leq \Delta'$), while Algorithm 6 uses $M'_v \leq \Delta$ by default ($\Delta = \max_v |M_v|$ by definition). Lines 8 and 9 record the best set of seed nodes searched among all possible scenarios (specified by line 2).

The total number of all possible scenarios is $O(|V| \cdot \Delta \cdot 2^{\Delta})$. The time complexity of Algorithm 6 is $O(2^{\Delta} k|V||E|)$. Hence, the time complexity of Algorithm 7 is $O(4^{\Delta}\Delta k|V|^2|E|)$. But Algorithm 5.13 has a better bound than Algorithm 6:

Theorem 5.13. *Algorithm 7 has an approximation ratio of* $1 - e^{-\frac{1}{\Delta+1}}$ *to the optimal*

algorithm.

Proof: Let S^* denote the optimal set of seed nodes, in terms of maximizing $\sigma(\cdot)$. Since Algorithm 7 exhausts all possible v', Δ', and S_0 in line 2, there must exist a scenario in which $v' = \arg\max_{v'} |M_{v'} \cap S^*|$, $\Delta' = |M_{v'} \cap S^*|$, $S_0 \subseteq M_{v'} \cap S^*$, and $|S_0| = k \mod (\Delta' + 1)$. All the following proof is based on the above scenario, although Algorithm 7 picks the best effort among all scenarios (lines 8 and 9).

In the above scenario, we claim that $\sigma(S_i)$ in each greedy iteration of Algorithm 6 has a lower bound to $\sigma(S^*)$:

$$\sigma(S_i) \geq (1 - \frac{1}{k'})^i \times \sigma(S_0) + [1 - (1 - \frac{1}{k'})^i] \times \sigma(S^*) \tag{5.5-15}$$

Here, k' is defined as $k - [k \mod (\Delta' + 1)]$. In other words, k' is the largest multiple of $\Delta' + 1$ constrained by $k' \leq k$. Eq. 5.5-15 is proved by induction. It is trivial that Eq. 5.5-15 holds when $i = 0$, since $(1 - \frac{1}{k'})^0 = 1$. Assume that Eq. 5.5-15 holds for i, and we prove that Eq. 5.5-15 holds for $i+1$. Since $S_0 \subseteq (M_{v'} \cap S^*) \subseteq S^*$ and $|S_0| = k \mod (\Delta' + 1)$, we have:

$$|S^* \setminus S_0| = |S^*| - |S^0| = k - [k \mod (\Delta' + 1)] = k' \tag{5.5-16}$$

Similarly, let us order nodes of $|S^* \setminus S_0|$ in an arbitrary order (say $v_1, v_2, ..., v_{k'}$), and let $S_j^* = \{v_1, v_2, ..., v_j\}$ for $1 \leq j \leq j'$ ($S_0^* = \emptyset$). Similar to Eq. 5.5-11, we have:

$$\begin{aligned}
&\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S^*)) - \sigma(S_i) \geq \sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S_{j-1}^*)) - \sigma(S_i) \\
\geq\ &\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S_{j-1}^*)) - \sigma(S_i \cup (M_{v_j} \cap S_{j-1}^*))) \\
\geq\ &\sigma(S_i \cup \{v_j\} \cup S_{j-1}^*) - (S_i \cup S_{j-1}^*)
\end{aligned} \tag{5.5-17}$$

The first and second inequalities are from the monotonicity in Theorem 5.6, since

$S^*_{j-1} \subseteq S^*$ and $S_i \subseteq S_i \cup (M_{v_j} \cap S^*_{j-1}))$. So $\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S^*)) \geq \sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S^*_{j-1}))$ and $\sigma(S_i) \leq \sigma(S_i \cup (M_{v_j} \cap S^*_{j-1})))$. The third inequality is from the definition of the modularity set, since only nodes in M_{v_j} can increase the marginal gain of v_j (other nodes might decrease the marginal gain of v_j). By accumulating Eq. 5.5-17 among j, we have the following inequality:

$$\begin{aligned}
&\sum\nolimits_{j=1}^{k'} [\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S^*)) - \sigma(S_i)] \\
\geq &\sum\nolimits_{j=1}^{k'} [\sigma(S_i \cup \{v_j\} \cup S^*_{j-1}) - (S_i \cup S^*_{j-1})] \\
= &\sigma(S_i \cup S^*) - \sigma(S_i) \geq \sigma(S^*) - \sigma(S_i) \qquad (5.5\text{-}18)
\end{aligned}$$

The first inequality is from Eq. 5.5-17. The equality results from the definition of S^*_j, since $\{v_j\} \cup S^*_{j-1} = S^*_j$. Note that, since $S^*_{k'} \subseteq S^* \setminus S_0$ and $S_0 \subseteq S_i$, we have $S_i \cup S^*_{k'} = S_i \cup S^*$. We have $S_i \cup S^*_0 = S_i$ since $S^*_0 = \emptyset$. The last inequality is from the monotonicity, since $S^* \subseteq S_i \cup S^*$. We have:

$$\begin{aligned}
&\sigma(S_{i+1}) - \sigma(S_i) = \sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i) = \frac{1}{k'} \sum\nolimits_{j=1}^{k'} [\sigma(S_i \cup \{v\} \cup M'_v) - \sigma(S_i)] \\
&\geq \frac{1}{k'} \sum\nolimits_{j=1}^{k'} [\sigma(S_i \cup \{v_j\} \cup (M_{v_j} \cap S^*)) - \sigma(S_i)] \geq \frac{1}{k'} [\sigma(S^*) - \sigma(S_i)] \qquad (5.5\text{-}19)
\end{aligned}$$

The first inequality is because line 5 in Algorithm 7 always selects the maximum marginal gain in each greedy iteration. $M_{v_j} \cap S^*$ is also constrained by $|(M_{v_j} \cap S^*)| \leq \Delta'$, since the scenario sets $v' = \arg\max_{v'} |M_{v'} \cap S^*|$ and $\Delta' = |M_{v'} \cap S^*|$. The second inequality is from Eq. 5.5-18. We rewrite Eq. 5.5-19 as:

$$\begin{aligned}
&\sigma(S_{i+1}) \geq \frac{1}{k'} [\sigma(S^*) - \sigma(S_i)] + \sigma(S_i) = \frac{1}{k'} \times \sigma(S^*) + (1 - \frac{1}{k'}) \times \sigma(S_i) \\
&\geq (1 - \frac{1}{k'})^{i+1} \times \sigma(S_0) + [1 - (1 - \frac{1}{k'})^{i+1}] \times \sigma(S^*) \qquad (5.5\text{-}20)
\end{aligned}$$

The first equality is from Eq. 5.5-19 and the last equality is from the induction

Table 5.1: Dataset Statistics.

	Forum	Board	Citation
Number of nodes	899	355	16,726
Number of hyperedges	67,332	2,684	92,462
Maximum hyperdegree	8,577	107	351
Power-law exponent γ	2.36	3.50	3.36

hypothesis (substituting $\sigma(S_i)$ in Eq. 5.5-15). Eq. 5.5-15 is proved by induction.

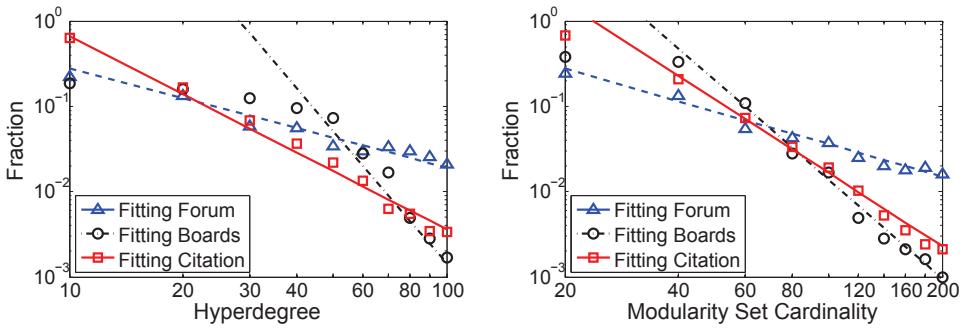
Since each greedy iteration of Algorithm 7 selects at most $\Delta' + 1$ seed nodes, Algorithm 7 has at least $\lfloor k/(\Delta'+1) \rfloor = k'/(\Delta'+1)$ greedy iterations. If we use $i = k'/(\Delta'+1)$ for Eq. 5.5-15, the proof completes:

$$\begin{aligned}
&\sigma(S) \geq \sigma(S_{k'/(\Delta'+1)}) \\
\geq &(1 - \frac{1}{k'})^{\frac{k'}{\Delta'+1}} \times \sigma(S_0) + [1 - (1 - \frac{1}{k'})^{\frac{k'}{\Delta'+1}}] \times \sigma(S^*) \\
\geq &[1 - (1 - \frac{1}{k'})^{\frac{k'}{\Delta'+1}}] \times \sigma(S^*) \\
\geq &(1 - e^{-\frac{1}{\Delta'+1}}) \times \sigma(S^*) \geq (1 - e^{-\frac{1}{\Delta+1}}) \times \sigma(S^*) \qquad (5.5\text{-}21)
\end{aligned}$$

The approximation ratio is $1 - e^{-\frac{1}{\Delta'+1}} \geq 1 - e^{-\frac{1}{\Delta+1}}$. ■

5.5.5 Time Complexity Reduction

Although Algorithm 7 has a better bound than Algorithm 6, its time complexity is much larger. However, we claim that the time complexity of Algorithm 7 can be reduced for OSNs. This is mainly because we do not need to exhaust all possible scenarios for practical usage. Rather than exhausting Δ' from 1 to Δ, we can simply stop at a small constant. For example, we only exhaust Δ' from 1 to 3. This is because only v's neighbors who share hyperedges with large weights are important in M_v (people are not likely to have many close friends). Similarly, we do not need to exhaust each node v for the initialization of S_0. Instead, we can focus on the largest-hyperdegree nodes (e.g., only the top 100 nodes). This is because the optimal



(a) Distribution of d_v among v. (b) Distribution of $|M_v|$ among v.

Figure 5.4: Distribution of d_v and $|M_v|$ in three datasets.

set of seed nodes is not likely to exclude the largest-hyperdegree nodes. Using this approach, the time complexity of Algorithm 7 can be reduced to $O(k|V||E|)$, which is asymptotically the same as Algorithm 5. Our experiments demonstrate that this approach only slightly hurts the performance of Algorithm 7. In addition, to compute M_v for Algorithms 6 and 7, we can simply exhuast v's neighborhoods (e.g., all two-hop neighbors) as an estimation (since other nodes are not impactful).

5.6 Experiments

5.6.1 Dataset Information and Statistics

Our experiments are based on three datasets (Forum, Board, and Citation) from Tore Opsahl [103]. Forum records user activities in a forum with different topics. Board records directors belonging to the boards of some companies. Citation records collaborations among chapter authors. The dataset statistics are shown in Table 5.1. The distributions of node hyperdegree (i.e., d_v) and modularity set cardinality (i.e., $|M_v|$) are shown in Fig. 5.4. For the above three datasets, flags (triangles, circles, and squares) represent the real distributions by statistics, and lines (dotted, dashed, and solid) are the fitting curves. Fig. 5.4 is plotted in a log-log manner, and the y-axis shows the fraction of nodes corresponding to the x-axis. Fig. 5.4(a) validates the

power-law distribution. It can be seen that the fraction of nodes with hyperdegree d is proportional to $d^{-\gamma}$ in each of these three dataset. The distribution of modularity set cardinality also follows power-law, as shown in Fig. 5.4(b). However, the power-law exponents for d_v and $|M_v|$ may not be the same in a given dataset. Fig. 5.4(b) further shows that only a small fraction of nodes have modularity sets with cardinalities larger than 100. Finally, according to Table 5.1 and Fig. 5.4, we can find that Forum is relatively denser than Board and Citation, since the average and maximum hyperdegrees in Forum are largest.

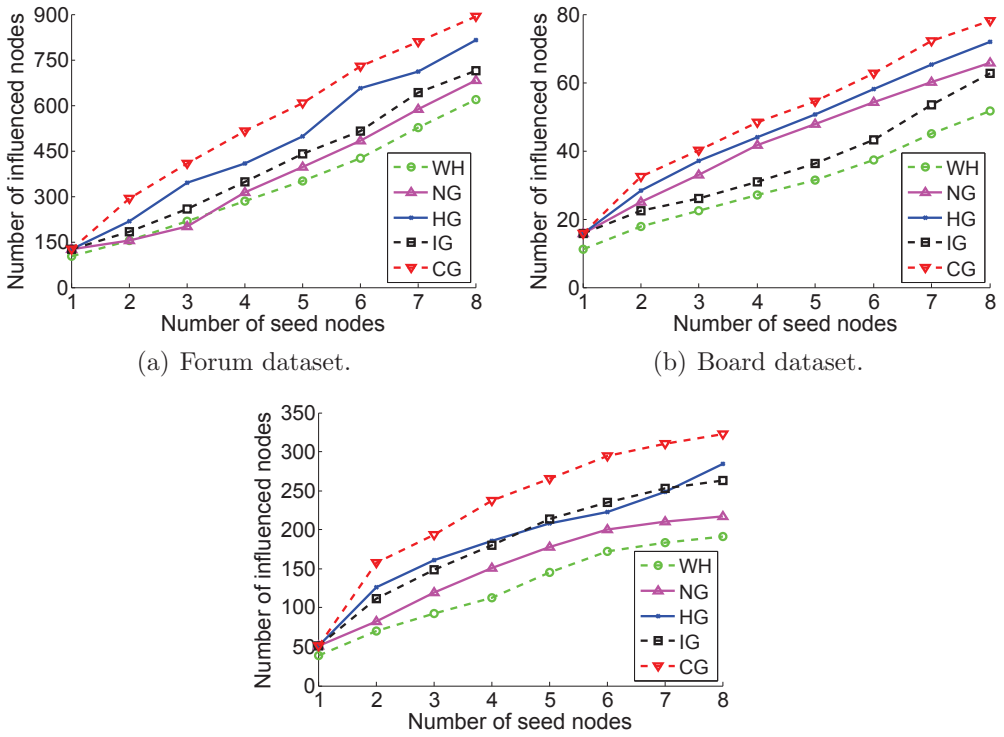
5.6.2 Comparison Algorithm and Performance

This subsection focuses on evaluating the performances of proposed algorithms, in terms of maximizing the number of eventually-influenced users. Algorithms 5, 6, and 7 are denoted as NG, IG, and CG, respectively. Comparison algorithms are:

- Weighted Hyperdegree (WH). It ranks all nodes by their hyperdegrees and selects the top k nodes as seed nodes.
- Hyperedge-aware Greedy (HG) from Hung et al. [3]. It iteratively selects the set of nodes that maximizes the ratio of marginal gain to set cardinality. The greedy iteration terminates when k nodes are selected. HG is not bounded.

As a drawback, WH ignores the crowd influence during the seed node selection. The drawback of HG is its complexity, since an exponential time complexity is needed to consider all sets of nodes in each greedy iteration. To control the running time, the set cardinality is capped at for HG, IG, and CG.

All evaluation results are shown in Fig. 5.5. Figs. 5.5(a), 5.5(b), and 5.5(c) correspond to the Forum, Board, and Citation datasets, respectively. A larger result represents a better performance, since seed nodes could eventually influence more nodes on expectation. Interestingly, Fig. 5.5 shows that not all algorithms follow the



(a) Forum dataset.

(b) Board dataset.

(c) Citation dataset.

Figure 5.5: Algorithm performance for SIMP in hypergraphs.

principal of diminishing return. The marginal gain of one seed node might not scale down with respect to the number of existing seed nodes in S. This is because our SIMP in hypergraphs is not submodular, i.e., $\sigma(S)/|S|$ might be larger than $\sigma(S')/|S'|$ for $S' \subset S$. Among all the algorithms, CG achieves the best performances in all these three datasets, while WH has the worst performances. This is simply because CG considers the impact of crowd influences while WH does not. Compared to other algorithms, CG has at least 10%, 5%, and 15% more eventually-influenced users in Forum, Board, and Citation, respectively (for $k = 8$).

HG has the second best performance, although it does not outperform IG in Citation. This is because NG and IG are essentially special cases of HG. HG reduces to NG by only selecting one node in each greedy iteration, and it reduces to IG by ignoring the set cardinality in each greedy iteration. CG, HG, IG, and NG become identical when only one seed node is selected (i.e., $k = 1$). HG loses to CG, since

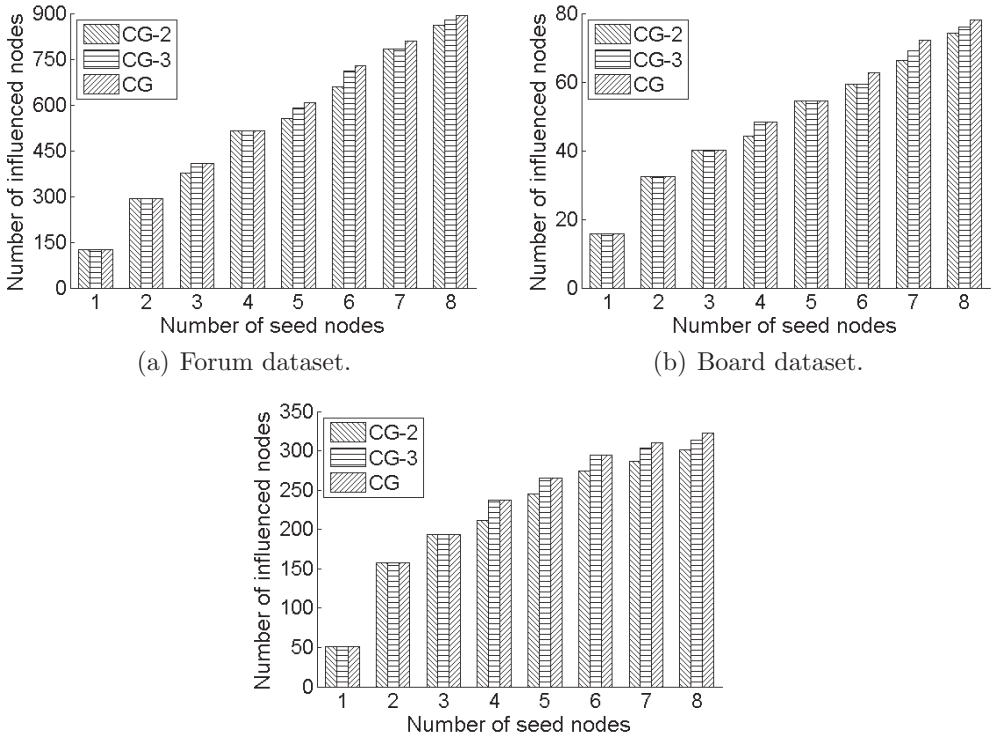
CG has a better granularity control through its cap to capture the crowd influence. Moreover, HG is unbounded and has a larger time complexity than CG. Finally, we find that the network density may not significantly change the algorithm performance. Both Board and Citation are sparse, but their algorithm performance gaps are not similar. Forum and Citation have different densities, but their algorithm performance gaps are similar.

5.6.3 Running Time and Complexity Reduction

This subsection evaluates the running time of the proposed algorithms. Codes are implemented in Matlab and are executed on Dell Inspiron i15RN-3647BK laptop with a 2.5GHz Intel Core i5 2450M processor. We further introduce two variations of CG by using different maximum cap sizes. The first one is CG-2, using 2 as its maximum cap size (Δ' ranges from 1 to 2). In each greedy iteration, CG-2 selects at most 2 nodes into seed nodes. The second one is CG-3, using 3 as its maximum cap size. We evaluate the impact of the cap size, in terms of both performance and running time. k is set to be 8.

	WH	NG	HG	IG	CG-2	CG-3	CG
Forum	2s	31m	2d	45m	75m	155m	22h
Board	1s	7m	21h	15m	31m	58m	5h
Citation	18s	85m	28d	114m	169m	423m	5d

We start with the running time, as shown in the above table (units are seconds, minutes, hours, and days). WH is fastest, since it is linear and does not evaluate $\sigma(S)$ for a given S. As a trade-off, it has the worst performance. Both NG and IG take minutes. IG runs slower than NG, since IG exhausts a set of nodes during each greedy iteration. However, IG is not exponentially slower than NG, since IG has fewer greedy iterations. The performance gap between NG and IG is limited in these three datasets. HG and CG have the longest running times to obtain the best



(a) Forum dataset.

(b) Board dataset.

(c) Citation dataset.

Figure 5.6: The impact of cap size for CG.

performances. The running times of both HG and CG grow quickly with respect to the dataset size. However, if we cap Δ' at 2 or 3, the running time of CG can be significantly reduced. CG-2 and CG-3 have asymptotically the same time complexity as NG. Fig. 5.6 shows the impact of the cap size for CG in these three datasets. CG-2, CG-3, and CG have close performances (CG-2 $\leq$ CG-3 $\leq$ CG). This is because CG is not likely to select a large set of nodes in each iteration (people are not likely to have many close friends). CG-3 has almost the same performance as CG, especially when seed nodes are few. If we jointly consider the running time aspect, capping Δ' to 3 in CG is a practical strategy for large-scale OSNs.

5.7 Summary

Motivated by the impact of the crowd influence, this chapter focuses on the Social Influence Maximization Problem (SIMP) in hypergraphs, which is NP-hard, monotone, non-submodular, and inapproximable within a ratio of $|V|^{\epsilon-1}$ for any $\epsilon > 0$ in a general hypergraph. However, since user connections in OSNs are not random, approximations could be obtained by leveraging the structural properties of OSNs. The modularity, Δ, can measure to what degree our problem violates the submodularity. Two approximation algorithms are proposed with ratios of $\frac{1}{\Delta+2}$ and $1 - e^{-1/(\Delta+1)}$, respectively. Experiments demonstrate the efficiency and effectiveness of our algorithms.

CHAPTER 6

EFFECTIVE SOCIAL NETWORK QUARANTINE WITH MINIMAL ISOLATION COSTS

Chapters 6, 7, and 8 focus on the information propagation applications for social networks. More specifically, this chapter studies the epidemic information propagation and proposes a quarantine strategy. Note that the notion of diseases has been extended from real human diseases to general epidemic information propagations, such as the rumors in distributed systems. Controlling the spread of a disease is usually done through quarantine, where people that have, or are suspected to have, a disease are isolated from having interactions with others. As a tradeoff, normal human interactions are inevitably degraded by the quarantine. This motivates us to explore a robust quarantine strategy that can eliminate epidemic outbreaks with minimal isolation costs. Our problem is shown to be NP-hard. A bounded algorithm with an approximation ratio of 2 is proposed, through utilizing the feasibility and minimality properties. Real data-driven experiments demonstrate the efficiency and effectiveness of the proposed algorithms in real-world applications.

6.1 Introduction

Nowadays, the notion of diseases has been extended from real human diseases to general epidemic information propagations, such as the rumors in distributed systems. Controlling the spread of a disease in a population (human communities and

distributed systems) is usually done through quarantine where people that have, or are suspected to have, a disease are restricted from having interactions with others. However, the human interactions are inevitably degraded by the quarantine. This motivates us to explore an effective quarantine strategy that can maximally preserve the human interactions.

This chapter uses the classic Susceptible-Infected-Susceptible (SIS) model to simulate the epidemic spreading, where each person has a state of being either susceptible or infected [104]. People transfer their states through a cycle in which their susceptibility (S) causes them to become infected (I), and they return to being susceptible (S) by recovery. The epidemic breaks out when the average infection rate becomes larger than the average recovery rate. In such a case, a large portion of people in the social network will eventually become infected. Consequently, it is necessary to isolate a set of people as a quarantine strategy to depress the infection rate. Epidemic outbreaks could be controlled once the infection rate is cut down by isolations, which are usually costly.

Our objective is to explore an effective quarantine strategy that can eliminate epidemic outbreaks with minimal isolation costs. In other words, we want to isolate a set of people with minimal costs to eliminate epidemic outbreaks. Our problem is extremely challenging, since eliminating epidemic outbreaks and preserving social connections cannot be simultaneously achieved. Intuitively, the isolation of an arbitrary person can help in the elimination of epidemic outbreaks. Moreover, social networks are structurally heterogenous, meaning that the impacts of isolations are very hard to quantify. Should we simply isolate people who have lots of normal social connections? Or should we isolate people who only have a few, but important connections? The network structural heterogeneity should be considered within the quarantine strategy design.

Currently, social network epidemic outbreaks [104] have been well-studied with very rich literatures. The novelty of this chapter lies in the quarantine strategy with minimal isolations. Our work casts some new light on real-world quarantines. For example, in an infectious global epidemic (SARS or Ebola), we can avoid epidemic outbreaks while isolating a minimal number of people. Moreover, our work has broader impacts, since it can be applied to situations aside from real social networks. Typical applications of our work can include:

- In computer networks, epidemics are worms. Some computers are turned off to eliminate worm spreading. Our work points out a strategy that can resist worms with maximally-preserved computers.
- In online social networks (e.g., Facebook and Twitter), epidemics are rumors (malicious Facebook posts) that can be shared among friends. The service provider can refer to our work to control rumors with minimal user blocks.

Our main contributions are summarized as follows:

- We address a novel problem on the effective quarantine that can restrict epidemic outbreaks with minimal isolations. It has broader impacts on real-world applications.
- An approximation algorithm, which guarantees a constant ratio to the optimal quarantine strategy, is proposed. The properties of feasibility and minimality are explored for eliminating epidemic outbreaks.
- Real data-driven experiments are conducted to evaluate the proposed algorithms. Evaluation results are shown from different perspectives to provide insightful conclusions for real-world applications.

The remainder of this chapter is organized as follows. Section 6.2 surveys related works. Section 6.3 formulates the problem and describes the SIS epidemic model with discussions on the properties of isolations. Section 6.4 studies the effective quarantine

strategy with minimal isolations. Section 6.5 includes real data-driven experiments. Finally, Section 6.6 concludes the chapter and suggests future research directions.

6.2 Related Work

Epidemic spreading models have been extensively explored over the past two decades. As one of the most popular epidemic models, the SIS model divides a given population into two compartments: susceptible and infected. People transfer their states through a cycle of being infected after being susceptible, and going back to susceptible by recovery [104]. The SIS model in social networks that have power-law degree distributions was summarized by Lee et al. [105]. There are many other epidemic models. For example, the Susceptible-Infected-Recovered (SIR) model [104] adds one more compartment to represent vaccinated individuals who are no longer susceptible to the infection. The SIS and SIR models were also used to stimulate worm spreadings in distributed systems [106]. A state-of-the-art review on epidemic models is available in [107].

Social networks have also been extensively studied over the past decade. Structures of social networks were confirmed to be scale-free [97], where the node degree follows power-law distribution. The triadic closure phenomenon (a friend-of-a-friend is likely to become a friend) is identified [108]. Social networks are considered to have small-world structures [109]. Compared to random networks, social networks have smaller network diameters and a larger clustering coefficient [110].

Although epidemic models and social network properties have been well-studied, to the best of our knowledge, this chapter is the first interdisciplinary study on a quarantine strategy that minimizes isolations without epidemic outbreaks. Our approximation scheme is based on the classic solutions to the knapsack problem [111, 112, 113] and the set cover problem [114, 115, 116, 117].

6.3 Problem Formulation and Epidemic Model

6.3.1 Problem Formulation

Our social network model is based on a directed graph $G = (V, E)$, where V is a set of nodes (persons), and $E \subseteq V^2$ is a set of directed edges (social relationships). Let $|\cdot|$ denote the cardinality of the corresponding variable. For example, $|V|$ and $|E|$ are the total number of nodes and edges, respectively. To control epidemic outbreaks, some nodes are isolated by the quarantine strategy. A node v is isolated, if all the incoming and outgoing edges of v are removed. An isolated node can no longer interact with its neighbors, but it remains in the network. The isolation cost of v is C_v. The set of nodes isolated by the quarantine strategy is denoted by Q. The objective is to explore a quarantine strategy that eliminates epidemic outbreaks with minimal $\sum_{v \in Q} C_v$.

6.3.2 Epidemic Outbreak Model

Our epidemic spreading model is based on the classic SIS model [105]. Nodes have states of either being susceptible or infected. Nodes in the susceptible state are people who do not have the disease, but can potentially catch it. Nodes in the infected state are people who have the disease and can spread the disease to their neighbors in G. Infected nodes can go back into the susceptible state upon recovery, and then can be reinfected. We consider that the infection rate of a given node depends on its infected incoming neighbors. For the node v, each of its infected incoming neighbors independently brings a constant infection rate (the infection probability per time unit) of λ to v. Meanwhile, the recovery rate is set to be a constant of r, as used in many existing models [104, 118, 119].

Let $f(t)$ denote the average fractions of nodes that are infected at time t. To capture the structural heterogeneity of the social network, let $p(d)$ denote the fraction

of nodes with in-degree d, and let $f_d(t)$ denote the fraction of infected nodes with in-degree d at time t. By definition, we have $f(t) = \sum_d p(d) \cdot f_d(t)$. Then, $\Theta(f(t))$ is the probability that a uniform-randomly selected edge comes from an infected node at the time t. It can be calculated as:

$$\Theta(f(t)) = \frac{\sum_d d \cdot p(d) \cdot f_d(t)}{\sum_d d \cdot p(d)} \tag{6.3-1}$$

The fraction of susceptible nodes with in-degree d at time t is $[1 - f_d(t)]$. Each of these nodes has an incoming degree of d, meaning that it is expected to have $d \times \Theta(f(t))$ infected incoming neighbors. Since each infected incoming neighbor brings an infection rate of λ, the total infection rate is:

$$1 - (1 - \lambda)^{d \cdot \Theta(f(t))} \approx \lambda \cdot d \cdot \Theta(f(t)) \tag{6.3-2}$$

λ is assumed to be small. Otherwise, the epidemic is not controllable due to an overly-large infection rate. We have:

$$\frac{\partial f_d(t)}{\partial t} = \lambda d \Theta(f(t))[1 - f_d(t)] - r f_d(t) \tag{6.3-3}$$

The first term of $\lambda d[1 - f_d(t)]\Theta(f(t))$ indicates the fraction of newly infected nodes that have in-degrees of d. The last term of $r f_d(t)$ shows the recovery. If we consider a stable epidemic state of $\frac{\mathrm{d}f_d(t)}{\mathrm{d}t} = 0$, then Eq. 6.3-3 can be solved as:

$$f_d(t) = \frac{\lambda d \Theta(f(t))}{r + \lambda d \Theta(f(t))} \tag{6.3-4}$$

According to Eq. 6.3-4, Eq. 6.3-1 can be rewritten as:

$$\Theta(f(t)) = \frac{1}{\sum_d d p(d)} \sum_d d p(d) \frac{\lambda d \Theta(f(t))}{r + \lambda d \Theta(f(t))} \tag{6.3-5}$$

The epidemic outbreak elimination indicates that $\Theta(f(t)) = 0$ and $\Theta(f(t))$ will not increase with respect to the time:

$$\frac{\partial}{\partial \Theta(f(t))}\left(\Theta(f(t)) - \frac{\sum_d dp(d)\frac{\lambda d\Theta(f(t))}{r+\lambda d\Theta(f(t))}}{\sum_d dp(d)}\right) \geq 0 \tag{6.3-6}$$

When $\Theta(f(t)) = 0$, Eq. 6.3-6 should be satisfied to control the growth trend of infected nodes. Based on Eq. 6.3-6, we can derive the following prerequisite to control epidemic outbreaks:

$$\frac{\lambda \sum_d d^2 p(d)}{r \sum_d dp(d)} \leq 1 \quad \text{or} \quad \frac{\langle d^2 \rangle}{\langle d \rangle} \leq \frac{r}{\lambda} \tag{6.3-7}$$

Let $\langle \cdot \rangle$ denote the mean value of the corresponding variable. Then, we have $\sum_d d^2 p(d) = \langle d^2 \rangle$ and $\sum_d dp(d) = \langle d \rangle$ by the definition. Eq. 6.3-7 represents the prerequisite of controlled outbreaks in social networks. The key insight of Eq. 6.3-7 is that both a larger average degree and a larger degree variance bring a more vulnerable network with respect to epidemic outbreaks:

$$\frac{\langle d^2 \rangle}{\langle d \rangle} = \frac{\langle d^2 \rangle - \langle d \rangle^2}{\langle d \rangle} + \langle d \rangle \tag{6.3-8}$$

Note that $\langle d \rangle$ is the average in-degree and $\langle d^2 \rangle - \langle d \rangle^2$ is the in-degree variance. The ratio of $\langle d^2 \rangle$ to $\langle d \rangle$ represents the network vulnerability to epidemics (the larger, the more vulnerable). If we want to control epidemic outbreaks, then we need to control the in-degree distribution though the isolations. Once a node is isolated, its associated incoming and outgoing edges are removed, leading to a degradation on $\frac{\langle d^2 \rangle}{\langle d \rangle}$ to control epidemic outbreaks. For simplicity, let $\Delta(Q)$ denote the degradation of $\frac{\langle d^2 \rangle}{\langle d \rangle}$, when nodes in Q are isolated by the quarantine strategy. We introduce a constant coefficient of $\delta = \frac{\langle d^2 \rangle}{\langle d \rangle} - \frac{r}{\lambda}$ as the degradation threshold to control epidemic outbreaks. At this time, our objective can be reformulated as minimizing $\sum_{v \in Q} C_v$ with the constraint

that $\Delta(Q) \geq \delta$. Further analysis is conducted in the next subsection.

6.3.3 Feasibility and Minimality

This subsection explores the inherent properties of $\Delta(Q)$ to obtain more insights on the effective quarantine strategy design. We start with the following definition:

Definition 6.1. A quarantine strategy, Q, is said to be feasible, if the constraint of $\Delta(Q) \geq \delta$ is satisfied.

Basically, a quarantine strategy that can eliminate epidemic outbreaks is defined as a feasible quarantine strategy. In reality, a feasible quarantine strategy usually isolates lots of nodes to control epidemic outbreaks. But for the sake of the theory, we still consider the event that $\{\exists v \in V | \Delta(\{v\}) \geq \delta\}$. It means that isolating only one node of v may be sufficient to control epidemic outbreaks. To facilitate the quarantine strategy design, we make a cutoff on $\Delta(Q)$. If $\Delta(\{v\}) \geq \delta$, we force $\Delta(\{v\})$ to be δ as a cutoff. Note that such a cutoff will not change the feasibility of an arbitrary quarantine strategy. Hence, the optimal quarantine strategy is not changed by this cutoff. We have the following definition:

Definition 6.2. A feasible quarantine strategy, Q, is said to be minimal, if $Q \setminus \{v\}$ is not feasible for an arbitrary $v \in Q$.

A minimal quarantine strategy means that each node in this quarantine strategy is necessarily isolated. If an arbitrary node in this quarantine strategy is no longer isolated, this quarantine strategy becomes infeasible and can no longer control epidemic outbreaks. Our key observation is that a minimal feasible quarantine strategy has the following property:

Theorem 6.3. *A minimal feasible quarantine strategy, Q, satisfies the property that $\delta \leq \Delta(Q) \leq 2\delta$.*

Proof: By the definition of feasibility, we have $\Delta(Q) \geq \delta$. Therefore, we focus on

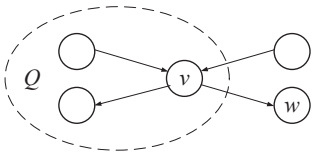


Figure 6.1: Proof of Theorem 6.3.

proving $\Delta(Q) \leq 2\delta$. Let us start with a special case, where all the nodes in Q do not have outgoing neighbors. In such a case, the isolation of a node in Q will not diminish the in-degrees of the remaining nodes. Let d denote the node in-degree when no node is isolated. d_v is the in-degree of the node v. Then, $\Delta(Q)$ can be calculated as:

$$\Delta(Q) = \frac{\langle d^2 \rangle}{\langle d \rangle} - \frac{\langle d^2 \rangle - \frac{1}{|V|}\sum_{v \in Q} d_v^2}{\langle d \rangle - \frac{1}{|V|}\sum_{v \in Q} d_v} \tag{6.3-9}$$

In Eq. 6.3-9, $\frac{1}{|V|}$ results from the fact that each node in Q is a fraction, $\frac{1}{|V|}$, of all the nodes. As another form of Eq. 6.3-7, $\langle d^2 \rangle$ and $\langle d \rangle$ can also be computed by $\langle d^2 \rangle = \frac{1}{|V|}\sum_{v \in V} d_v^2$ and $\langle d \rangle = \frac{1}{|V|}\sum_{v \in V} d_v$, respectively. We assume that $\langle d \rangle \gg \frac{1}{|V|}\sum_{v \in Q} d_v$, leading to the following approximation:

$$\frac{1}{\langle d \rangle - \frac{1}{|V|}\sum_{v \in Q} d_v} \approx \frac{1 + \frac{1}{\langle d \rangle}\frac{1}{|V|}\sum_{v \in Q} d_v}{\langle d \rangle} \tag{6.3-10}$$

Back to Eq. 6.3-9 with the substitution in Eq. 6.3-10, we can obtain:

$$\Delta(Q) \approx \frac{1}{|V|\langle d \rangle}\Big[\sum_{v \in Q} d_v^2 - \frac{\langle d^2 \rangle}{\langle d \rangle}\sum_{v \in Q} d_v\Big] + o\Big(\frac{1}{|V|\langle d \rangle}\Big) \tag{6.3-11}$$

In Eq. 6.3-11, $o(\frac{1}{|V|\langle d \rangle})$ represents the second order term that is relatively ignorable. Eq. 6.3-11 implies the following result:

$$\Delta(Q) \leq \Delta(Q \setminus \{u\}) + \Delta(\{u\}) \tag{6.3-12}$$

Eq. 6.3-12 is obtained through comparing the first and second order terms in Eq. 6.3-

11 for the left and right parts of Eq. 6.3-12. According to the definition of the minimality, we can obtain the result that $\Delta(Q \setminus \{u\}) < \delta$, since $Q \setminus \{u\}$ is not a feasible quarantine strategy. Meanwhile, we have $\Delta(\{u\}) \leq \delta$, since we have made a cutoff on $\Delta(Q)$. Therefore, we have:

$$\Delta(Q) \leq \delta + \delta = 2\delta \tag{6.3-13}$$

Eq. 6.3-13 concludes that $\delta \leq \Delta(Q) \leq 2\delta$ is true in the special case, where all the nodes in Q do not have outgoing neighbors. The insight of this case is that the isolation of each node is independent to each other. Hence, $\Delta(Q)$ can be decomposed, as shown in Eq. 6.3-12, to obtain its upper bound.

Let us go back to the general case, where nodes in Q may have outgoing neighbors. Note that the isolation of a node in Q may diminish the in-degree of a node that is not in Q. An example is shown in Fig. 6.1, where the isolation of v diminishes the in-degree of w. Let Q' denote the set of nodes that are not in Q, but have diminished in-degrees due to the isolations of nodes in Q. For the node $v \in Q'$, let ρ_v denote its in-degree. Then, for the general case, Eq. 6.3-9 is rewritten as:

$$\begin{aligned}\Delta(Q) &= \frac{\langle d^2 \rangle}{\langle d \rangle} - \frac{\langle d^2 \rangle - \frac{1}{|V|}\sum_{v \in Q} d_v^2 - \frac{1}{|V|}\sum_{v \in Q'}(d_v^2 - \rho_v^2)}{\langle d \rangle - \frac{1}{|V|}\sum_{v \in Q} d_v - \frac{1}{|V|}\sum_{v \in Q'}(d_v - \rho_v)} \\ &= \frac{\langle d^2 \rangle}{\langle d \rangle} - \frac{\langle d^2 \rangle - \frac{1}{|V|}\sum_{v \in Q \cup Q'} d_v^2 + \frac{1}{|V|}\sum_{v \in Q'} \rho_v^2}{\langle d \rangle - \frac{1}{|V|}\sum_{v \in Q \cup Q'} d_v + \frac{1}{|V|}\sum_{v \in Q'} \rho_v}\end{aligned} \tag{6.3-14}$$

Eq. 6.3-14 has a similar format with Eq. 6.3-9. Through a similar derivation, we find that the analysis in Eq. 6.3-12 still holds for the general case. Therefore, we conclude that $\delta \leq \Delta(Q) \leq 2\delta$ is true, which completes the proof. ■

The key insight behind Theorem 6.3 is that a minimal feasible quarantine strategy would not lead to excessive isolations. Unnecessary isolations are saved once the epidemic outbreak is controlled. In a minimal feasible quarantine strategy, each node

is necessarily isolated. By contradiction, it can be seen that the optimal solution for our problem must be a minimal feasible quarantine strategy. We will describe an effective quarantine strategy through utilizing the minimality property.

6.4 Effective Social Network Quarantine

6.4.1 NP-hardness and Marginal Greedy Strategy

The objective of this chapter is to design an effective quarantine strategy that eliminates epidemic outbreaks with minimal isolation costs. This problem is NP-hard:

Theorem 6.4. *Searching an optimal quarantine strategy of Q, which minimizes $|Q|$ with $\Delta(Q) \geq \delta$, is NP-hard.*

Proof: We prove the NP-hardness by a reduction to the partial set cover problem [120] in a special case, where node in-degrees are identical. Note that node out-degrees may not be the same. In such a case, $\frac{\langle d^2 \rangle}{\langle d \rangle} = \langle d \rangle$, meaning that the prerequisite of controlling epidemic outbreaks is reduced to controlling the average node in-degree. In other words, epidemic outbreaks could be eliminated through breaking up $\delta|V|$ edges. Our problem becomes minimizing $\sum_{v \in Q} C_v$ with the constraint that $\delta|V|$ edges are broken. If we correspond an edge to an element, and correspond a node to a set, then our problem reduces to a partial set cover problem that uses the sets with minimal costs to cover $\delta|V|$ elements. Since the partial set cover problem is NP-hard by a reduction to the set cover problem [120], our problem is also NP-hard. ■

We first present an intuitive greedy solution, as shown in Algorithm 8. It iteratively isolates the node v that can minimize $\frac{C_v}{\Delta(\{v\} \cup Q) - \Delta(Q)}$ (i.e., minimal "cost-to-benefit" ratio). Algorithm 8 will terminate, when the quarantine strategy of Q becomes feasible, i.e., $\Delta(Q) \geq \delta$. The time complexity of Algorithm 8 is $O(V^2)$. This is

Algorithm 8 Marginal Greedy

Input: The social network, G, and the threshold, δ.
Output: The quarantine strategy, Q.

1: Initialize $Q = \emptyset$.
2: **while** $\Delta(Q) < \delta$ **do**
3: $\quad v = \arg\min_{v \in V \setminus Q} \frac{C_v}{\Delta(\{v\} \cup Q) - \Delta(Q)}$.
4: $\quad Q = Q \cup \{v\}$.
5: **return** Q as the quarantine strategy.

because it has $O(V)$ iterations, and each iteration takes $O(V)$ to go through all the remaining nodes for the isolation decision. However, Algorithm 8 cannot guarantee an approximation ratio to the optimal solution. The difficulty comes from the fact that the number of isolated nodes in the optimal solution is unknown. Algorithm 8 may isolate many more, or many fewer nodes than the optimal solution. On the other hand, even if the number of isolated nodes in the optimal solution is known a prior, we cannot guarantee the feasibility of the solution with the same number of isolated nodes. Further explorations are conducted.

6.4.2 Homogeneous Greedy Strategy

The key observation of our approach is that a minimal feasible quarantine strategy would not lead to excessive isolations (Theorem 6.3). Following this intuition, a homogeneous greedy solution is proposed, as shown in Algorithm 9. It is a recursive algorithm that splits the node cost through a homogeneous function. At each recursion level, it isolates the node v that can minimize $\frac{C_v}{\Delta(\{v\} \cup Q) - \Delta(Q)}$ (i.e., minimal "cost-to-benefit" ratio), as implemented in lines 3 to 5. Then, in lines 6 to 9, it splits the node cost through a homogeneous function for a recursive call. Finally, in lines 10 to 12, some nodes in Q are removed to satisfy the minimality property. We claim that Algorithm 9 can guarantee an approximation ratio:

Theorem 6.5. *Algorithm 9 guarantees an approximation ratio of two to the optimal solution for the isolation costs.*

Algorithm 9 Homogeneous Greedy (recursive)

Input: The social network, G, the threshold, δ,
and the incomplete quarantine strategy, Q'.
Output: The quarantine strategy, Q.

1: **if** $\delta < 0$ **then**
2: **return** $\emptyset$;
3: $v = \arg\min_{u \in V \setminus Q'} \frac{C_u}{\Delta(Q' \cup \{u\}) - \Delta(Q')}$.
4: Set coefficient $\epsilon = \frac{C_v}{\Delta(Q' \cup \{v\}) - \Delta(Q')}$.
5: $Q' = Q' \cup \{v\}$.
6: **for** each $u \in V \setminus Q'$ **do**
7: $C'_u = \epsilon \times \Delta(\{u\})$. /* split node cost */
8: $C_u = C_u - C'_u$. /* residual node cost */
9: $Q = Q' \cup \text{RECURSIVE}(G, \delta - \Delta(Q'), Q')$.
10: **for** each $u \in Q$ **do**
11: **if** $Q \setminus \{u\}$ is a feasible quarantine strategy **then**
12: $Q = Q \setminus \{u\}$.
13: **return** Q as the quarantine strategy.

Proof: We prove by induction. For the base case, we have $\delta < 0$ with $Q = \emptyset$ and $\sum_{v \in Q} C_v = 0$. Therefore, the base case holds. For the general case, Algorithm 9 isolates the node v that can minimize $\frac{C_v}{\Delta(\{v\} \cup Q) - \Delta(Q)}$, splitting the corresponding node cost in line 7. The residual node cost is shown in line 8 for a recursive call. We have:

$$\sum_{u \in Q} C_u = \sum_{u \in Q} C'_u + \sum_{u \in Q} (C_u - C'_u) \tag{6.4-15}$$

Let Q_1^*, Q_2^*, and Q^* denote the optimal solutions for G with the isolation costs to be C'_u, $C_u - C'_u$, and C_u, respectively. According to Theorem 6.3, we have $\sum_{u \in Q} C'_u \leq 2 \sum_{u \in Q_1^*} C'_u$ by the minimality property. This is because C'_u scales linearly with respect to $\Delta(\{u\})$. Meanwhile, by induction, we have $\sum_{u \in Q} (C_u - C'_u) \leq 2 \sum_{u \in Q_2^*} (C_u - C'_u)$.

Table 6.1: Dataset Statistics

	Epinions	Wikipedia
Number of nodes	18,098	7,115
Number of edges	355,754	103,689
Average degree	19.6	14.6
In-degree Variance	3615.8	1006.9
Network Diameter	11	7
Global clustering coefficient	0.138	0.141
Average edge weight	0.0285	0.0076

Consequently, the following inequality holds:

$$\sum_{u\in Q} C_u = \sum_{u\in Q} C'_u + \sum_{u\in Q}(C_u - C'_u) \leq 2\sum_{u\in Q_1^*} C'_u + 2\sum_{u\in Q_2^*}(C_u - C'_u)$$
$$\leq 2\sum_{u\in Q^*} C'_u + 2\sum_{u\in Q^*}(C_u - C'_u) = 2\sum_{v\in Q*} C_u \qquad (6.4\text{-}16)$$

The last inequality holds, since Q^* is not the optimal solution for G, with the isolation costs being C'_u or $C_u - C'_u$. The result in Eq. 6.4-16 completes the proof that Algorithm 9 guarantees an approximation ratio of two to the optimal solution. ■

The key idea of Theorem 6.5 is to utilize the minimality property. A minimal feasible quarantine strategy will not have excessive isolations, leading to bounded isolation costs. Algorithm 9 has a complexity of $O(V^2)$. It has a recursion depth of $O(V)$, while each recursion call takes $O(V)$ to select the node and keep the minimality.

6.5 Experiments

6.5.1 Dataset Information and Settings

Our experiments are based on two datasets of Epinions [80] and Wikipedia [81]. Epinions is a general consumer review site, which was launched in 1999. Epinions users can read new and old reviews about a variety of products to help them decide on a purchase [121]. Our approach can be applied to the isolations of users in eliminating

online rumor spreading. Wikipedia is a free encyclopedia written collaboratively by volunteers (i.e., users) around the world. A small portion of users are administrators. In order for a user to become an administrator, a request must be issued and then voted upon. If a user (say v) votes for another user (say u), then there exists a directed edge from v to u. Our approach can be applied on isolating users to control disordered votes in Wikipedia. The dataset statistics are summarized in Table 6.1.

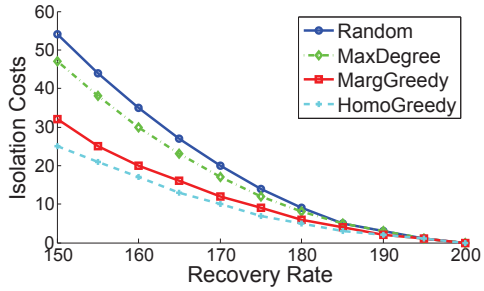
Note that these two datasets do not include information on node isolation costs. Hence, three cost functions are designed:

- The first cost function uses a constant cost for each node, meaning that all the nodes have identical isolation costs.
- The second cost function determines the node isolation cost based on the node in-degree with a logarithmic mapping. For the node v, we have $C_v = \log(d_v+1)$.
- The third cost function determines the node isolation cost based on the node in-degree with a square root mapping. For the node v, we have $C_v = \sqrt{d_v}$.

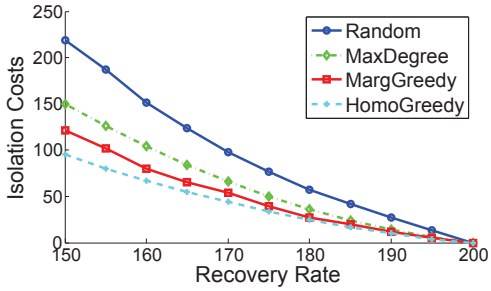
The node costs are normalized for fair comparison. As for the parameters on epidemic spreading, we set a fixed infection rate of $\lambda = 1$. The recovery rate, r, is tuned within the experiments. Note that $\langle d^2 \rangle / \langle d \rangle \leq r/\lambda$ shows the prerequisite for controlling epidemic outbreaks. Therefore, a smaller recovery rate means that more isolations (and thus higher isolation costs) are needed to control epidemic outbreaks.

The following four algorithms are involved for evaluations:

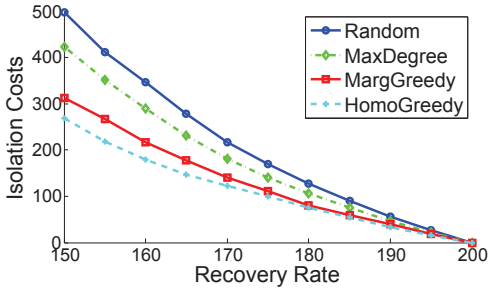
- Random. It iteratively and uniform-randomly isolates a node in G, until the epidemic outbreak is eliminated by the quarantine strategy.
- MaxDegree. This algorithm ranks nodes by their degrees. Top ranked nodes are iteratively isolated until the epidemic outbreak is eliminated by the quarantine strategy.



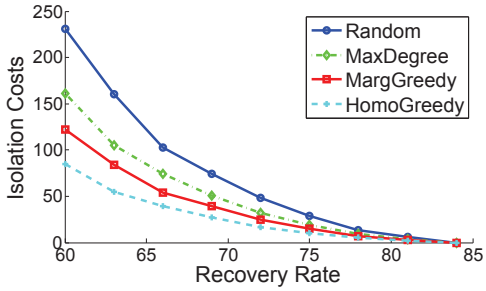
(a) Epinions with constant cost.



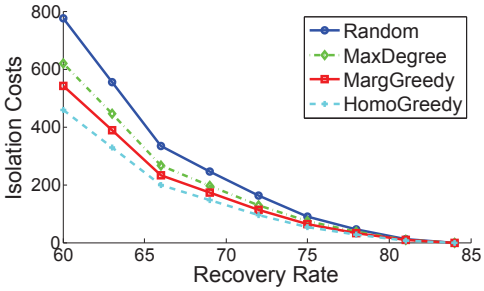
(b) Epinions with logarithmic cost.



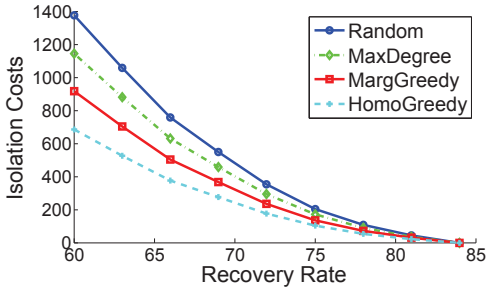
(c) Epinions with square root cost.



(d) Wikipedia with constant cost.



(e) Wikipedia with logarithmic cost.



(f) Wikipedia with square root cost.

Figure 6.2: Evaluation results with respect to the isolation costs.

- MargGreedy. This is Algorithm 8, which iteratively isolates the node v that can minimize $\frac{C_v}{\Delta(\{v\}\cup Q)-\Delta(Q)}$ (i.e., minimal "cost-to-benefit" ratio).
- HomoGreedy. This is Algorithm 9, which obtains a bounded result through the minimality property.

6.5.2 Evaluation Results

The evaluation results are shown in Fig. 6.2, in terms of the relationship between the recovery rate and the isolation cost. Fig. 6.2 has three columns, each of which

corresponds to a different node isolation cost function. Figs. 6.2(a), 6.2(b), and 6.2(c) are the results for the Epinions dataset, while Figs. 6.2(d), 6.2(e), and 6.2(f) are the results for the Wikipedia dataset.

It can be seen that the isolation cost decreases monotonously with respect to the recovery rate. This is because a high recovery rate can resist epidemic spreadings. If the recovery rate is high enough, then epidemic outbreaks can be eliminated without isolations. HomoGreedy always has the lowest isolation costs among the comparison algorithms. Another interesting observation is that, when the cost of a node isolation is a constant, the total isolation cost is the smallest. This is because the quarantine strategy tends to isolate large degree nodes, while their costs are relatively cheap after normalization. On the other hand, when the cost of a node isolation scales with its degree (in a logarithmic manner or a square root manner), the overall isolation costs become very large. This is because the isolations of large degree nodes take relatively large costs after normalization. The last observation is that, the total isolation costs in Epinions are smaller than those in Wikipedia. One reason is that Epinions has a much larger degree variance than Wikipedia, as shown in Table 6.1 (355,754 to 103,689). Another reason is that Epinions has more users than Wikipedia (18,098 to 7,115), bringing larger isolation costs.

6.6 Summary

This chapter explores a robust quarantine strategy that can eliminate epidemic outbreaks with minimal isolation costs. This problem is proved to be NP-hard. The classic SIS epidemic model is introduced to model epidemic spreading, where people transfer their states through a cycle of being infected from susceptible, and going back to susceptible by recovery. We show that a minimal feasible quarantine strategy will not have excessive isolations. A bounded algorithm with an approximation ratio

of two is proposed, through utilizing the feasibility and minimality properties. This algorithm has a time complexity of $O(V^2)$. Real data-driven experiments demonstrate the efficiency and effectiveness of the proposed algorithms in real-world applications.

CHAPTER 7

DYNFLUID: PREDICTING TIME-EVOLVING RATING IN RECOMMENDATION SYSTEMS VIA FLUID DYNAMICS

Chapters 6, 7, and 8 focus on the information propagation applications for social networks. More specifically, this chapter studies the trust information propagation in recommendation systems. If a user is predicted to have a high rating of a product, then this product is recommended to that user for shopping potential. Therefore, rating predictions are critical for qualified recommendations. In this chapter, based on the fluid dynamics theory, we propose a novel rating prediction scheme called DynFluid. The key observation is that the rating of a user depends on his/her user experience, as well as the ratings of other users. For example, users may refer to friends' ratings upon rating a product, themselves. DynFluid analogizes the rating reference among the users to the fluid flow among containers: each user is represented by a container; the rating of a user is mapped to be the fluid temperature in the corresponding container. Two user characteristics, persistency and persuasiveness, are also incorporated into DynFluid. Finally, real data-driven experiments in Epinions and Ciao validate the efficiency and effectiveness of the proposed DynFluid.

7.1 Introduction

Nowadays, increasing amounts of people are involved in Online Social Networks (OSNs) for completing daily activities, including forming an opinion on a particular

product. One central issue in OSNs is the notion of *trust*. More specifically, trust in OSNs denotes the subjective probability by which a user expects another given user to perform a given action. One important application of trust is in online recommendation systems, such as Epinions and Ciao [122]. Such systems have two essential components: the rating (or opinion) of a user on a product and the trust relationships among users. The product rating of a user depends on his/her user experience, as well as the existing ratings of the other users (such as trusted friends).

This chapter focuses on the rating prediction problem in trust-based recommendation systems [123]. If a user is predicted to have a high rating of a product, then the service provider can recommend this product to that user for shopping potential. As shown in Fig. 7.1, a typical online recommendation system provides the following product information to a user: a brief product description; a public broadcast channel (or simply *public channel*) that shows the average product rating of all users; the ratings and comments from the *trusted friends* of that user; the ratings and comments from strangers. The last one is not shown in Fig. 7.1, and is not considered in this chapter. This is because very few users would read the ratings and comments from strangers. On the other hand, trusted friends' ratings and the public channel are likely to be referred to when users give out their own ratings.

Our rating predictions are based on the references of the ratings among the users. To further understand the rating behavior of a user, here we introduce two concepts called *persistency* and *persuasiveness*. Persistency denotes how much a user insists on his/her own user experience. A user with a low persistency would like to refer to the ratings of other users. On the other hand, persuasiveness denotes the convincingness of a user's rating. A user with a higher persuasiveness indicates that his/her ratings are more likely to be referred to by the other users. These two user characteristics are critical for improving the accuracy of the rating predictions.

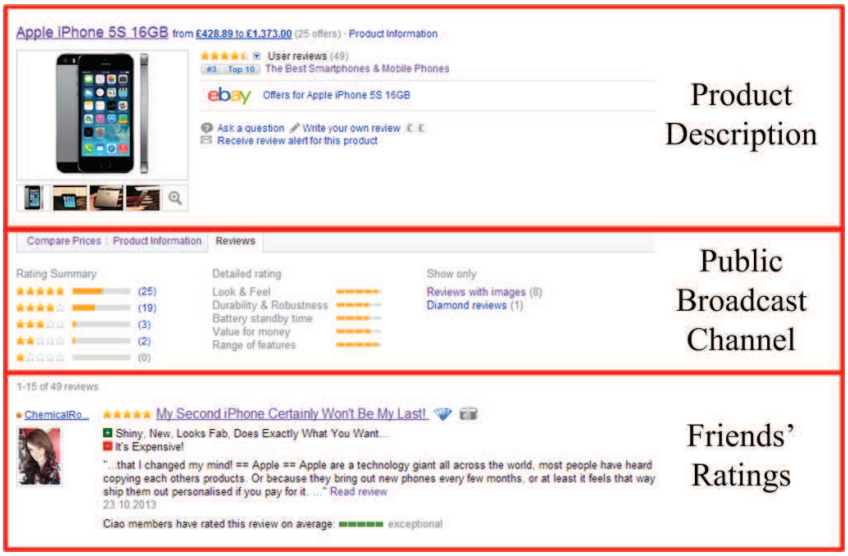


Figure 7.1: An illustration for the rating of a product in Ciao.

We first consider the scenario with one product. A subset of users (raters, denoted as R) have prior ratings on the product. The remaining non-raters (denoted as N) have not rated the product, but they can refer to the existing ratings before giving out their own ratings. In other words, the non-raters are influenced by the raters, in terms of the ratings. An example for our system is shown in Fig. 7.2, where the numbers on the top of the raters are their prior ratings of the product. Directional links are trust relationships. Then, a directional link from a_1 to a_3 means that a_1 is trusted by a_3, and thus a_3 may refer to a_1's rating before giving out its own rating.

As shown in Fig. 7.2, our main idea is to analogize the rating reference (representing the opinion propagation) among the users to the fluid flow among the containers. The users are mapped to be containers, while the trust relationships among users are mapped to be directional pipes (pipes with one-way valves). The fluid temperature in a container represents the rating of the corresponding user. For example, in Fig. 7.2, the rater a_1 has a rating of 3, and thus the corresponding fluid temperature is 3°C. Then, the persistency and persuasiveness of a user are represented by the fluid height and the cross-sectional area of the corresponding

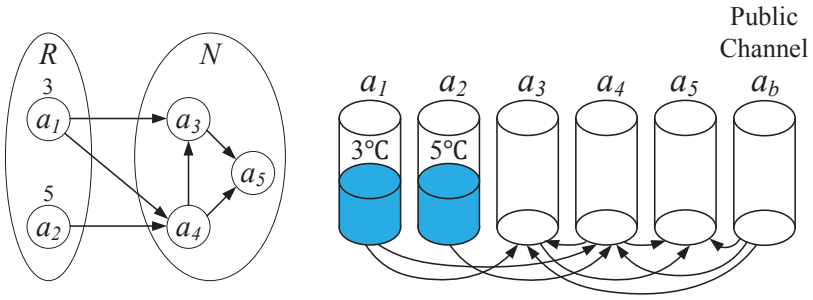


Figure 7.2: An illustration of DynFluid. Each user corresponds to a container, while the directional pipes represent the trust relationships among users. The container a_b is added to represent the public channel.

container, respectively. A higher fluid height indicates a larger persistency, and a larger cross-sectional area indicates a higher persuasiveness. This is because a container with a higher fluid height and a larger cross-sectional area can send out more fluid to change the ratings of its neighbors. Once there exists a fluid height difference between two connecting containers, fluid will gradually flow from one container to the other. These kinds of fluid dynamics represents how the users value their trusted friends' ratings in a time-evolving manner. To further incorporate the public channel, we add an extra container (the container a_b in Fig. 7.2) into the system. This extra container connects to all the non-raters, since the public channel can be seen by all the users. We are very interested in the impact of the public channel, compared to the impact of the trusted friends. For one user, is the public channel more trustworthy than one of the trusted friends? This question will be explored in our chapter.

Our results and contributions are summarized as follows:

- We propose a clean-slate rating prediction method, DynFluid, to capture the time-evolving ratings. DynFluid considers both the influence from the public channel and the ratings from trusted friends.
- We introduce the two concepts of persistency and persuasiveness, which reveal the users' rating behaviors. The fluid dynamics theory is used to model the rating process.

- Real data-driven experiments in Epinions and Ciao are conducted to evaluate the proposed DynFluid. The results are shown from different perspectives to validate the efficiency and effectiveness of our approach.

The remainder of this chapter is organized as follows: Section 7.2 surveys related work. Section 7.3 states the model and then formulates the problem. Section 7.4 describes DynFluid, including its analogy insights and algorithmic properties. Section 7.5 includes extensive real data-driven experiments. Finally, Section 7.6 concludes this chapter.

7.2 Related Work

This section reviews the literature of trust models, trust propagations, and ratings in the recommendation systems.

Trust models. Currently, a wide range of disciplines have examined various issues related to trust [124]. However, there is no consensus on how trust should be defined. In [125, 126, 127], trust in a person is defined as a commitment to an action, based on a belief that the future actions of that person will lead to a good outcome. Here, trust is subjective and personalized. We consider trust to be asymmetric. A user may trust another user more than he is trusted back. Another closely-related concept is reputation, which is usually an objective measure. One may trust a stranger if he/she has a strong reputation [128].

Trust Propagations. Generally speaking, the trust follows the principle of transitivity [124]. Trusts among users form a trusted graph. Sun et al. [129] proposed an information-theoretic framework on trust propagation by stating two axioms as possible guiding principals: (1) concatenation propagation of trust does not increase trust, and (2) multipath propagation of trust does not reduce trust. The existing path-based propagation methods include the Dempster-Shafer combination rule [130],

serial-parallel merge [131] using subjective logic, and path concatenation [132] from path algebra. Several models have been proposed for graph-based propagation. Both MoleTrust [126] and TidalTrust [125] are based on breadth-first search. TidelTrust selects the strongest shortest path, while MoleTrust uses the hop count (also called horizon) to control the length of the selected path. However, these approaches are information-lossy, while some more interesting approaches are graph analogy-based. In [133], a generalized reliability theory is applied to a trusted network with failure-prone elements. RelTrust [134] emulates a trusted graph with a resistive network, using a logarithmic function to map the trust values to the resistance values.

Ratings. In rating-based systems, the rating (or opinion) of a user is usually a numeric value on an online website [122]. Anderson et al. [135] introduced a finite integer set with $\{+, -, 0\}$ representing positive, negative, and neutral ratings. The predictions of positive ratings are more useful than those of negative ratings. This is because only products with predictions of positive ratings are recommended to the users. In our model, the rating is measured by the fluid temperature, which can be easily updated based on the fluid dynamics theory. Zhu et al. [136] found that a person's opinion is significantly swayed by others' opinions. Our DynTrust takes [137] as a foundation: when new opinions come, each person refines his/her opinion through exchanging opinions with friends.

7.3 Model and Problem Formulation

This chapter focuses on the rating prediction problem in trust-based recommendation systems. Accurate predictions can help the service provider recommend appropriate products to the user for shopping potential. Generally speaking, the rating of a user depends on his/her user experience, as well as the ratings of other users. For example, users may refer to the ratings of trusted friends

upon their own ratings (i.e., the opinion propagations among users). However, user experience depends on many external unknown factors, and thus is hard to predict. Therefore, we predict the rating of a user, based on the ratings of his/her trusted friends and the public channel.

To model the rating behavior of a user, the concepts of persistency and persuasiveness are introduced into our scheme. Persistency denotes how much a user relies on his/her own user experience. A user with a low persistency would like to refer to the ratings given out by the other users (including the trusted friends and the public channel). On the other hand, persuasiveness denotes the convincingness of a user's rating. A user with a higher persuasiveness means that his/her ratings are more likely to be referred to by the other users. Our scenario is based on a directed graph $G = (V, E)$, where V is a set of nodes (i.e., users), and $E \subseteq V^2$ is a set of directed edges (i.e., trust relationships). The edge $e_{aa'}$ has the direction from the node a to a', indicating that a is trusted by a', and thus a' may refer to a's rating before giving out his/her own rating. The node set V can be divided into two subsets: a subset of raters, R, and a subset of non-raters, N. The raters have given out their ratings on the product, and they cannot change their ratings anymore. The non-raters have not rated the product, but they may refer to the existing ratings to give out their own ratings, based on the persistency and persuasiveness of the users. The objective of this chapter is to predict the ratings of non-raters in a time-evolving manner.

7.4 DynFluid: Algorithm Details

7.4.1 Analogy Insights

Our main idea is that the rating reference process (or opinion propagation process) is analogous to the fluid flow. The users are modeled as containers, while the trust relationships among users are modeled as directional pipes (i.e., pipes with one-way

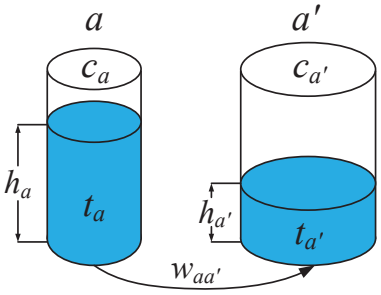


Figure 7.3: A motivational example to illustrate the analogy insights.

valves). All the containers are placed on the same horizontal plane. The containers are large enough to hold the fluid (the fluid will not overflow). Pipes are installed at the bottom of the containers, and may have different pipe sizes (the cross-sectional areas of pipes). For a pipe from a to a', its pipe size is denoted as $w_{aa'}$, which represents the strength of the corresponding trust relationship. A larger pipe size indicates a higher level of trust. As shown in Fig. 7.2, the fluid temperature in a container represents the rating of the corresponding user. Then, the persistency and persuasiveness of a user are represented by the fluid height and the cross-sectional area of the corresponding container, respectively. For the user a, the corresponding fluid temperature, fluid height, and the cross-sectional area is denoted by t_a, h_a, and c_a, respectively. A higher fluid height indicates a larger persistency, while a larger cross-sectional area indicates a higher persuasiveness.

To capture the analogy insights, a motivational example is provided in Fig. 7.3, where we have two containers (users) of a and a'. Then, we have four analogy insights. (1) The height of the fluid in a' is lower than that in a. Therefore, the fluid in a flows into a' through the one-way pipe, meaning that a' refers to the rating given out by its trusted friend a. In other words, a' does not insist on its own user experience (i.e., a low persistency), and thus it wants to refer to the ratings of its trusted friend for its own rating. (2) Since a has a very small cross-sectional area (i.e., a low persuasiveness), only a small volume of fluid would eventually flow from a to a' at the

end. This means that the rating of a is not very convincing, and thus, it only slightly changes the rating of a'. (3) The fluid from a mixes up with the existing fluid in a', representing that a' refines its own opinion in a time-evolving manner. The fluid temperature (the rating) of a' is changed by the fluid from a. (4) The pipe size has some impacts on the duration of the fluid flowing time. If $w_{aa'}$ is large, then the fluid in a flows into a' within a short time. This means that a' updates its rating more quickly, if a' trusts a more. These four analogy insights show that the fluid flow can accurately capture the rating reference process (or opinion propagation process).

7.4.2 Fluid Update Principles

Discrete Approach. In the real world, the fluid flow is a continuous-time system, which is hard to compute. Hence, the discrete approach is used. The continuous physical time is discretized into a series of time slots. The duration of each time slot is denoted as Δt. Fluid flows are described as a set of partial differential equations. We consider that the fluid update is performed synchronously at the end of each time slot. The update process is shown in Fig. 7.4, which corresponds to the example in Fig. 7.2. Let R denote the total number of fluid updates. At the beginning of the i^{th} time slot, we prepare the fluid update and check whether the fluid will flow in each pipe, by comparing the fluid heights of two connected containers. If there is a directional pipe from a to a', and the fluid height in a is higher than that of a', the fluid will flow from a to a'; if either of the two conditions do not meet, no fluid will flow. The first condition means that a is trusted by a', and thus, the rating of a may be referred to by a'. The second condition means that a' has a low persistency, and thus a' wants to refer to the rating of a. We record the volume and temperature of the flowing fluid. At the end of each time slot, we mix up the flowing fluid and the remaining fluid as the fluid updates.

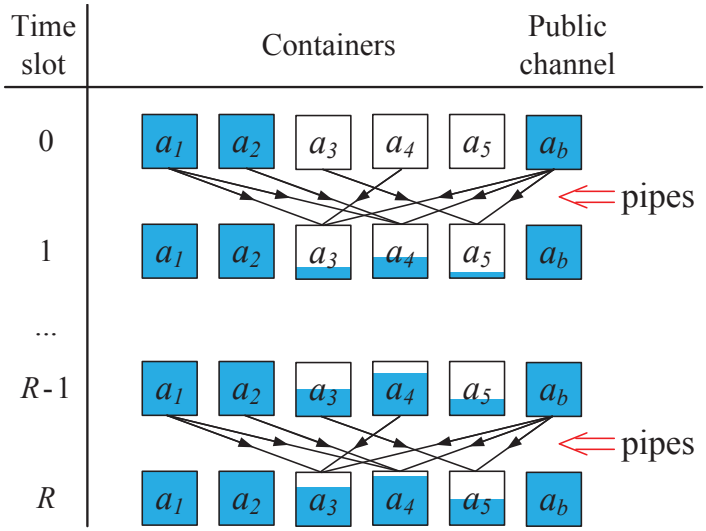


Figure 7.4: A discrete approach to compute the fluid flow.

Initialization. The fluid heights of all the non-raters are initialized to be zero, since they have not rated the product yet. The fluid height of each rater is initialized according to the persistency of that rater. The fluid temperatures of all the raters are initialized according to their ratings, as shown in Fig. 7.2. The cross-sectional area of each user is initialized based on the persuasiveness of that user. As for the public channel, it is modeled as an extra container that connects to all the non-raters. The initialization of the public channel is similar to that of the raters, the only exception being that its fluid temperature is set as the average fluid temperature among all the raters.

Update Details. First, let us consider a single pipe, say the pipe connecting a and a', with cross-sectional area $w_{aa'}$. This scenario has been shown in Fig. 7.3. At the beginning of the i^{th} time slot, if a has more fluid than a', fluid will flow from a to a' during this time slot of duration Δt. Based on the Bernoulli's principle [138], the speed of fluid flowing at the bottom of a will be $2g\sqrt{h_a(i) - h_{a'}(i)}$. Here, g is the gravitational acceleration, which is a constant. Hence, the volume of fluid that flows

from a to a' in the i^{th} time slot is:

$$s_{aa'}(i) = 2g\sqrt{h_a(i) - h_{a'}(i)} \times w_{aa'} \times \Delta t \tag{7.4-1}$$

The insight behind Eq. 7.4-1 is that, if a' has a lower persistency and a is trusted more by a', then the rating of a is more valuable to a'. In addition, the fluid temperature that corresponds to $s_{aa'}(i)$ is denoted as $t_{aa'}(i)$, which is equal to the temperature of the fluid in the container a. Let $s_a(i)$ denote the volume of the fluid in a at the i^{th} time slot. Then, for the $(i+1)^{th}$ time slot, the volume of the fluid in a can be calculated as:

$$s_a(i+1) = s_a(i) - \sum_{a' \in N_a^+} s_{aa'}(i) + \sum_{a' \in N_a^-} s_{a'a}(i) \tag{7.4-2}$$

where N_a^+ and N_a^- are the outgoing and incoming neighbors of a, respectively. The fluid height in a can be calculated by $h_a(i+1) = s_a(i+1)/c_a$, where c_a is the cross-sectional area of the container a. The cross-sectional area of a represents its persuasiveness, which has an impact on the final ratings of its outgoing neighbors. This is because a larger cross-sectional area will lead to a smaller fluid height change, after the current update. If a has a larger cross-sectional area, more fluid will eventually flow into its outgoing neighbors.

Let $t_a(i)$ denote the temperature of the fluid in a at the i^{th} time slot. According to the fluid mixing formula [138], the fluid temperature after being mixed up is:

$$t_a(i+1) = \frac{[s_a(i) - \sum_{a' \in N_a^+} s_{aa'}(i)] \cdot t_a(i) + \sum_{a' \in N_a^-} [s_{a'a}(i) \cdot t_{a'a}(i)]}{s_a(i+1)} \tag{7.4-3}$$

Eq. 7.4-3 is essentially $\sum(\text{volume} \cdot \text{temperature}) / \sum \text{volume}$. The first part of the numerator represents the remaining fluid in container a, while the second part

corresponds to the incoming fluid flowing from the other containers. The denominator is the volume of the mixed fluids, as shown in Eq. 7.4-2.

Considering that the raters have given out their ratings (i.e., they no longer update their ratings), the fluid heights of the raters are fixed. This can be viewed as giving additional fluid injections to the raters. The raters will no longer refer to the ratings of the other users, and their ratings are only referred to by the non-raters. As for the public channel, its fluid height is also fixed. If a user has given out his/her rating, then this user can no longer change his/her rating on the public channel.

7.4.3 Algorithm Overview and Time Complexity Analysis

DynFluid is shown in Algorithm 10. Lines 1 and 2 associate nodes and edges in G with containers and pipes, respectively. In line 3, an extra container is added to represent the public channel. Line 4 shows the initialization process. Lines 5 to 12 show the discrete approach for calculating the fluid flow. The continuous physical time is discretized into R time slots, i.e., R rounds of fluid updates. In each round, we first calculate the volume of the flowing fluid in each pipe, as shown in lines 6 to 8. In lines 9 and 10, we inject additional fluid to the containers that correspond to the raters and the public channel. At the end of each round, we update the fluid in the non-raters' containers (lines 11 and 12). Finally, in line 13, the fluid temperature in the non-rater's container is returned as the predicted rating.

The initialization of Algorithm 10 (lines 1 to 4) takes a time complexity of $O(V + E)$. Each round of fluid update in lines 5 to 12 also takes a time complexity of $O(V + E)$. Since fluid updates have R rounds, the total time complexity of Algorithm 10 is $O(R \cdot (V + E))$. Considering that social networks are generally sparse and R should be a small constant, the proposed DynFluid could be very efficient for real-world rating predictions that involve millions of users.

Algorithm 10 DynFluid

Input: The directed graph G and the existing ratings;
The parameters for the initialization;
Δt, R, and g for the fluid flow update;

Output: The predicted ratings of the non-raters;

1: Associate each node in G with a container;
2: Associate each edge in G with a directional pipe;
3: Add an extra container (i.e., public channel) that connects to all the non-raters' containers.
4: Initialize the fluid height and temperature in each container;
5: **for** $i = 0$ to $R - 1$ **do**
6: **for** each pipe from a to a' **do**
7: **if** $h_a(i) > h_{a'}(i)$ **then**
8: Calculate the volume and temperature of the outgoing flowing fluid, based on Eq. 7.4-1;
9: **for** each rater's container and the extra container **do**
10: Inject additional fluid to maintain the fluid height and temperature;
11: **for** each non-rater's container **do**
12: Mix up the incoming flowing fluid, as to update the fluid height and temperature, based on Eqs. 7.4-2 and 7.4-3;
13: **return** the fluid temperature in the non-rater's container.

7.4.4 Convergence Analysis

This subsection focuses on the convergence of the proposed DynFluid. First, we have the following theorem:

Theorem 7.1. *If we use a constant value (denoted by h) to initialize the fluid heights of all the raters and the public channel, then the fluid heights of all the non-raters will always be no larger than h, during the fluid updating process.*

Proof: We proof this theorem by contradiction. Suppose there is a non-rater a, whose fluid height h_a is larger than h. There are two possible cases for explaining a's fluid height. The first case is that the fluid in a comes directly from a rater (or the public channel) a', where $h_{a'} > h_a > h$. However, the fluid heights of all the raters (or the public channel) are h, since they only give out their fluids to non-raters. Therefore, the first case contradicts the assumption, which should be invalid. The second case

is that the fluid in a comes directly from a non-rater. However, working iteratively with this case will eventually result in a non-rater whose fluid comes directly from a rater. Therefore, the second case will be reduced to the first case, which is not valid by contradiction. ■

Theorem 7.2. *If we use a constant value (denoted by h) to initialize the fluid heights of all the raters and the public channel, then, after a time period that is sufficiently long, the fluid heights of all the non-raters will be h.*

Proof: Suppose there is a non-rater a. According to Theorem 7.1, we have $h_a \leq h$. If $h_a = h$ for any non-rater a, then the proof completes. If $h_a < h$, we also have two cases. The first case is that the fluid in a comes directly from a rater (or the public channel) a'. This means that $h_{a'} = h > h_a$. Since there is a pipe from a' to a, the flowing fluid in this pipe will eventually fill up the height gap between a' and a, meaning that we have $h_a = h$ after a sufficiently long time period. The second case is that the fluid in a comes directly from a non-rater. Iteratively doing this case will eventually reduce this case to the first case, since the fluids of all the non-raters originate from the raters and the public channel. ■

Theorems 7.1 and 7.2 show that the DynFluid will converge after certain rounds of fluid updates. It can be explained by the rating reference process in the real world. Initially, a user has no idea about the given product. Upon referring to the ratings of the other users, this user formulates and refines his/her own rating in a time-evolving manner. During this process, a person's opinion becomes more and more mature, indicating increased persistency. Therefore, this phenomenon is consistent with our real-world experiences.

7.4.5 Algorithm Properties

This subsection studies the properties of the proposed DynFluid. First, we have the following property:

Property 7.3. *In DynFluid, the opinion influence from a user, a, to another user, a', decays monotonously with respect to the hop-count distance from a to a'.*

This property indicates that a user is influenced more by his/her trusted friends and the public channel than strangers. In DynFluid, the 1-hop neighbor of a non-rater can pass their fluids directly to this non-rater, during the 1^{st} round of fluid updates. Meanwhile, the k-hop neighbor can only pass his/her fluids to that non-rater during the k^{th} round of fluid updates. The fluids from nearby neighbors arrive at the non-rater earlier (in terms of the discrete time slot) with a larger volume, since the fluid height of the non-rater is lower at an earlier time. The fluid from strangers arrives at that non-rater later with a smaller volume, since the fluid height of the non-rater becomes higher at a later time. The ratings of trusted friends and the public channel are more valuable than the strangers' ratings. Then, another property of DynFluid is:

Property 7.4. *In DynFluid, the certainty of the rating prediction for a non-rater can be measured by the fluid height (or persistency) of that non-rater.*

This property states that the certainty of the rating prediction is highly related to the persistency of the corresponding non-rater. DynFluid shows how a user refines its rating in a time-evolving manner. At the beginning, the user has a low persistency, and thus he/she receives multiple opinions from his/her trusted friends and the public channel. As time goes by, the opinion of a user becomes more and more mature, indicating that the persistency of that user gets higher and higher. Therefore, the rating of a user with a higher persistency is more stable than the one with a lower persistency, representing that the certainty can be measured by the persistency.

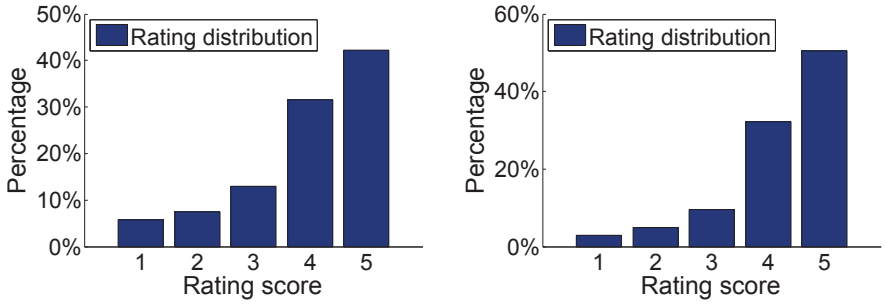
7.5 Evaluations

7.5.1 Basic Settings

Dataset information. In our experiments, the datasets of Epinions and Ciao [139] are used. These two datasets are collected online from www.epinions.com and www.ciao.com. These two datasets include the directional trust relationships among users, as well as the users' ratings (recorded as rating scores from 1 to 5) on some products. The distributions of rating scores in Epinions and Ciao are shown in Fig. 7.5. It can be seen that users are more likely to give out high ratings. More than 40% of rating scores are 5 (the highest rating score). Then, the Epinions dataset consists of 49,290 users who rated a total of 139,738 different products. The total number of issued trust relationships is 487,181. The Ciao dataset consists of 2,248 users who rated a total of 16,861 different products. The total number of issued trust relationships is 57,544.

Since the ratings on a specified product are generally sparse with respect to the graph size, a user may not have a trusted friend who has rated the same product as he/she did. Therefore, we do not run experiments directly on the whole dataset. Alternatively, given a product, we extract subgraphs to test rating predictions: if a user does not rate that product, or this user does not have a trusted friend who has rated that product, then this user is not considered in our experiments; otherwise, we generate a subgraph centered on that user to predict his/her rating. This subgraph is composed of all neighbors of that user within a certain hop count. We use the 1-hop, 2-hop, and 3-hop subgraphs. A larger subgraph provides more information, and thus a better performance should be obtained by the DynFluid.

Performance metric. To measure the prediction errors of the proposed DynFluid, the leave-one-out method [140] is used. For the generated subgraph of



(a) Rating scores in Epinions. (b) Rating scores in Ciao.

Figure 7.5: The distribution of the rating scores.

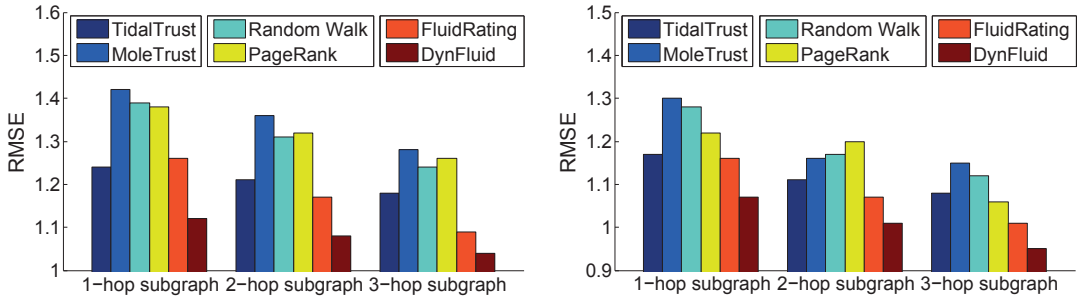
a specific user, the groundtruth rating of that user is masked and then predicted based on the generated subgraph. We compare the predicted rating with the masked groundtruth rating, the difference between which is the prediction error. To measure the prediction errors among different subgraphs and different products, we adopt the root mean squared error (RMSE) [127], which is the root of the mean squared prediction error among all the generated subgraphs and all the products. RMSE represents the standard deviation of the differences between predicted ratings and groundtruth ratings. A smaller RMSE indicates a better prediction.

The second performance metric is the classic F-score, which is the harmonic mean of precision and recall:

$$\text{F-score} = \frac{2TP}{2TP + FP + FN} \tag{7.5-4}$$

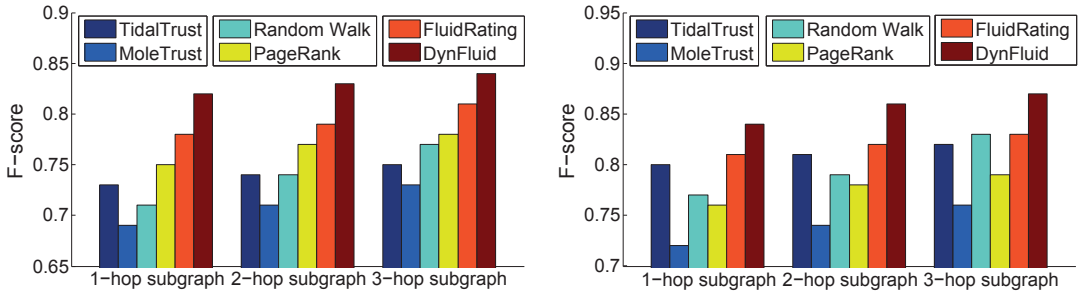
When the groundtruth is that the corresponding user has a rating score of no less than three (called positive rating), TP and FP (true and false positive cases) are the numbers of correct and incorrect predictions, respectively. Then, the false negative case, FN, is the number of incorrect predictions, when the groundtruth is that the corresponding user has a rating score of less than three (called negative rating). Note that a larger F-score indicates a better prediction.

Default parameters. In our experiments, the subgraph for rating predictions is



(a) The RMSE in the Epinions dataset. (b) The RMSE in the Ciao dataset.

Figure 7.6: Compare DynFluid with the other methods, in terms of RMSE.



(a) The F-score in the Epinions dataset. (b) The F-score in the Ciao dataset.

Figure 7.7: Compare DynFluid with the other methods, in terms of F-score.

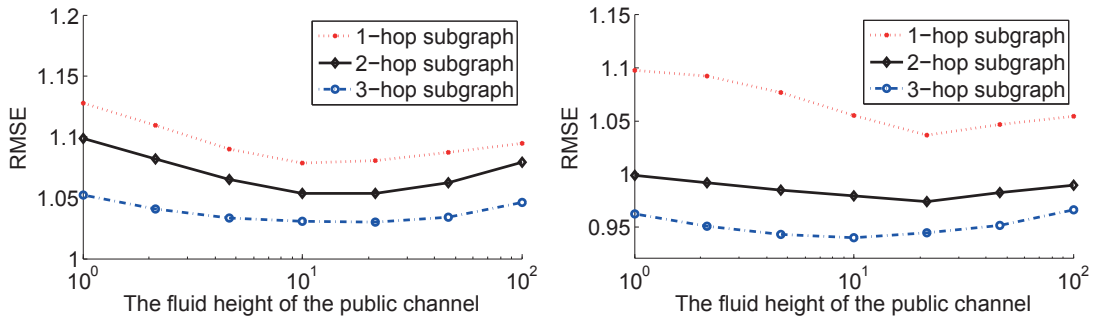
generated, in default, by all neighbors within 3 hops of a given user (also called 3-hop subgraph). Unless specified, we use $\Delta t = 0.1$ as the duration of each time slot and $R = 10$ as the rounds of fluid updates. The fluid heights of all the raters and the public channel are 10. The cross-sectional areas of all the containers are 1. The fluid temperature of each rater is initialized to be its rating score.

7.5.2 Comparisons with the Other Methods

This subsection compares DynFluid with the other state-of-the-art algorithms in terms of RMSE and F-score. Comparison algorithms include TidalTrust [125], MoleTrust [126], Random Walk [127], PageRank [135], and FluidRating [137]. (1) TidalTrust finds all trusted raters through the shortest path to the given user, and then aggregates their ratings as the predicted rating of that given user. (2) MoleTrust has two steps [141]. In the first step, it takes two given nodes (source and destination)

as the input and then builds a directed trust graph from the source to the destination. In the second step, it walks through the directed trust graph and calculates the trust values of the visited nodes. The rating of the destination node serves as the predicted rating of the source node. (3) Random Walk aggregates the ratings of users arriving by random walks from a given user as the predicted rating of that given user [127]. (4) PageRank computes the user's reputation for trust propagations. We take its result when it converges. (5) FluidRating is the foundation of this work. However, FluidRating does not consider the public channel, which is really impactful for the rating predictions.

The comparison results are shown in Figs. 7.6 and 7.7. The former one shows the results with the RMSE metric and the later one shows the results with the F-score metric. A smaller RMSE means a better result, while a larger F-score means a better result. Figs. 7.6(a) and 7.6(b) show the results for Epinions and Ciao, respectively. The left part of Fig. 7.6(a) is the result for the subgraphs that are generated by all neighbors within 1 hop of the given user, while the middle and right parts are those within 2 and 3 hops, respectively. DynFluid outperforms all the other algorithms, since it considers the ratings of the trusted friends and the public channel. For a 1-hop subgraph, DynFluid has a more than 10% improvement with respect to all the other methods. For a 3-hop subgraph, DynFluid has a more than 5% improvement over FluidRaing, and an over 10% improvement compared to the other methods. DynFluid performs better for subgraphs of neighbors within 3 hops than those within 1 hop. When the subgraph is larger, users can refer to more rating scores from their trusted friends and make a better rating. The performance gap between DynFluid and FluidRating gets smaller, when the generated subgraph is larger. This is because a larger subgraph brings more information on the public channel. The results in Ciao are similar to those in Epinions. However, the overall RMSE in Ciao is lower than that in Epinions (about 10% lower).

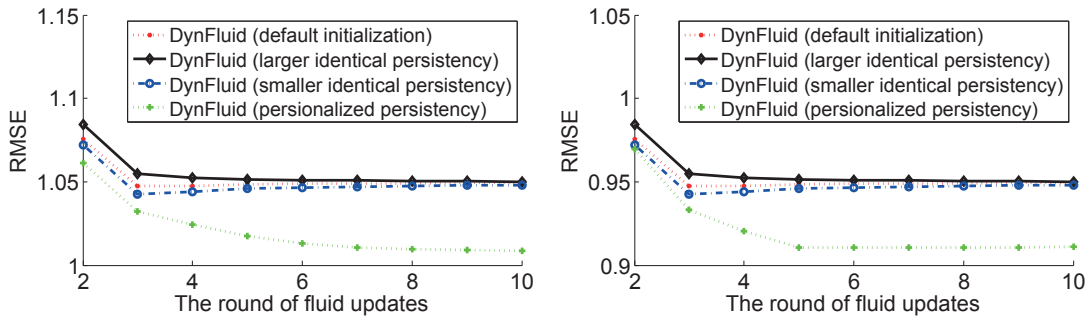


(a) The result for the Epinions dataset.

(b) The result for the Ciao dataset.

Figure 7.8: The impact of the public channel.

Fig. 7.7 shows the comparasion results with the F-score metric. Fig. 7.7(a) and 7.7(b) show the results for Epinions and Ciao, respectively. In Fig. 7.7(a), DynFluid has a significant performance improvement, compared to Random Walk (about a 15% higher F-score). For the other methods, a performance improvement that is larger than 5% is also obtained by DynFluid. According to the definition of F-score, DynFluid can accurately predict the true positive case, where the groundtruth is that the corresponding user has a rating score of no less than three (positive rating). Note that the true positive case is the most important case for DynFluid, since its motivation is the trust-based recommendation. If a user is predicted to have a high rating on a specified product, then the service provider can recommend this product to that user for the shopping potential. Although F-score does not consider the true negative case, this case is not important for recommendations. This is because the service provider should not recommend a product to a user, if this user is predicted to have a negative rating on that product. DynFluid has an F-score of about 85% when we use the 3-hop subgraph. Therefore, DynFluid is qualified for trust-based recommendations. The F-score of DynFluid in Ciao is a little bit higher than that in Epinions, indicating that the rating scores in Ciao are more predictable in the F-score metric. This is consistent with that in the RMSE metric.



(a) The result for the Epinions dataset.

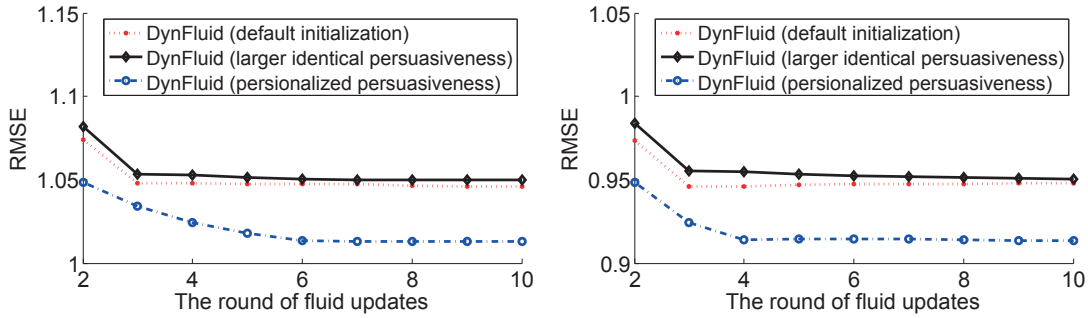
(b) The result for the Ciao dataset.

Figure 7.9: The impact of the user persistency.

7.5.3 The Impact of The Public Channel

In the previous subsection, we compare DynFluid with other state-of-the-art methods. Here, we conduct additional experiments to further understand the impact of the public channel. Instead of the default setting, where the fluid height of the public channel is initialized to be 10, we tune the fluid height of the public channel, as to observe the RMSE variance of the DynFluid. Note that the fluid heights of all the raters are set to be 10. A higher fluid height of the public channel means that it has a larger impact on the users.

The results are shown in Fig. 7.8, where we conduct the above experiments for DynFluid in subgraphs generated by all neighbors within 1 hop, 2 hops, and 3 hops, respectively. Fig. 7.8(a) and 7.8(b) show the results for Epinions and Ciao, respectively. In both datasets, the RMSE of DynFluid first decreases and then increases, with respect to the initial fluid height of the public channel. When the initial fluid height of the public channel is 1, the DynFluid mainly uses the ratings of the trusted friends for rating predictions, leading to a high RMSE. In other words, if the impact of the public channel is ignored, then the rating prediction becomes inaccurate. On the other hand, when the initial fluid height of the public channel is 100, the DynFluid mainly uses the public channel for rating predictions, which also leads to a high RMSE. This means that the ratings of the trusted friends are also



(a) The result for the Epinions dataset.

(b) The result for the Ciao dataset.

Figure 7.10: The impact of the user persuasiveness.

not negligible. Note that the RMSE of the DynFluid goes to the minimum, when the initial height of the public channel is set to be about 10 to 20. Meanwhile, the initial heights of all the raters are 10. This observation implies that the public channel helps to improve the accuracy of the prediction. We also find that the impact of the public channel is more significant in 1-hop subgraphs than that in 3-hop subgraphs.

7.5.4 The Impact of The Persistency and Persuasiveness

In the previous experiments, we use a constant of 10 to initialize the fluid heights of all the raters. The cross-sectional areas of all the containers are set to be 1. Here, we initialize the fluid heights and the cross-sectional areas in a personalized way, as to see the impact of the persistency and persuasiveness. We estimate those two characteristics of a user through the total number of ratings (on different products) given out by that user. If a user has rated more products, we consider that user to have a higher persistency and persuasiveness, through a linear mapping process. This is because the user is more likely to insist on his/her own opinions, and is more likely to be an authority on the product for the other users.

First, we conduct experiments to observe the impact of the non-identical persistency, while the persuasiveness remains identical. The experimental results are shown in Fig. 7.9. It can be seen that the DynFluid with personalized persistency

has a better performance than the DynFluid with the default setting (about 5% lower RMSE in both Epinions and Ciao). If we initialize the persistency to be identical, then a larger initial value or a smaller initial value has a very limited impact on the converged RMSE of the DynFluid. The variance caused by different initial values is less than 3%. This is because a higher fluid height also leads to a larger volume of flowing fluid as a self-regulation in the fluid flow system. DynFluid with personalized persistency has a slower convergence speed than that with identical persistency. This is because the user with a low persistency needs more time to refine his/her opinions. We also conduct experiments to observe the impact of the non-identical persuasiveness, while the persistency is identical. The experimental results are shown in Fig. 7.10. It can be seen that the DynFluid with personalized persuasiveness has a better performance (more than 5% lower RMSE) than the DynFluid with the default setting. If we initialize the persuasiveness to be identical, then its initial value has some impacts on the converged RMSE of the DynFluid. A too-large initial value leads to a performance degradation, since larger cross-sectional areas can weaken the impact of flowing fluid.

7.6 Summary

This chapter studies a rating prediction problem in trust-based recommendation systems. We find that the rating of a user depends on his/her user experience, as well as the ratings of other users. This chapter proposes a novel rating prediction scheme called DynFluid, based on the fluid dynamics theory. DynFluid analogizes the rating reference among the users to the fluid flow among containers: each user is represented by a container; the rating of a user is mapped to be the fluid temperature. Two user characteristics, persistency and persuasiveness, are incorporated into DynFluid. Real data-driven experiments validate the performance of our DynFluid.

CHAPTER 8

FAST INFORMATION CASCADE PREDICTION THROUGH SPATIOTEMPORAL DECOMPOSITIONS

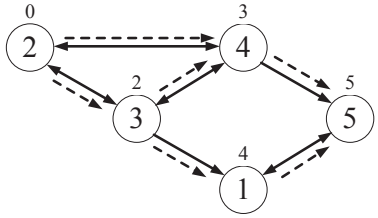
Chapters 6, 7, and 8 focus on the information propagation applications for social networks. More specifically, this chapter studies the information cascade in online social networks. Information cascades occur when people observe the actions of others (followees) and then make the same choices that the others have made (followers). Cascade predictions are important, since they can detect and help resist bad cascades. We focus on photo cascade predictions in Flickr: given the current cascade and social topology, we want to predict the number of propagated users at a future time slot. Information cascades include a large amount of data that crosses both space and time. To reduce prediction time complexities, our idea is to decompose the spatiotemporal cascade information (a larger size of data) to user characteristics (a smaller size of data) for subsequent predictions. Space and time matrices are introduced to record the cascade information. We introduce a set of new notions, persuasiveness and receptiveness (represented as two vectors for complexity reduction), to capture characteristics of followees and followers. Persuasiveness includes followees' abilities to propagate information, while receptiveness includes followers' willingness to accept information. Then, we propose a three-stage parallel prediction scheme as follows. (1) We map the spatiotemporal cascade information to a weighted matrix, in which the weights of space and time information are tuned. (2) Singular value decomposition is

used to extract nodes' persuasiveness and receptiveness from the weighted matrix. (3) Predictions are conducted based on nodes' persuasiveness and receptiveness. Evaluations are conducted to verify the performance of the proposed scheme.

8.1 Introduction

Nowadays, online social networks (OSNs), which belong to typical large distributed systems, are a fundamental medium for spreading information, such as sharing startling news, creative ideas, and interesting stories. An information cascade may occur if a user follows another user: if Alice (a followee) shares a photo, Bob (a follower) may scan this photo and then share it to his/her followers later. This type of iterative information propagation is called an *information cascade*. Meanwhile, cascade predictions are important in various aspects of human lives, such as in the control of computer viruses, prevention of infectious diseases, inhibition of terrible rumors, estimation of economic products, and the forecast of marketing strategies. However, the cascade prediction is very difficult, due to its intrinsic complexities: when will a user further propagate the information? This chapter captures propagation boundaries spatiotemporally, i.e., through both social topological information and time information. More specifically, given a cascade before a time τ_1 and the social topology, we want to predict the number of propagated users (called the cascade size) at a future time slot τ_2 (assuming that the information source appears at $\tau_0 = 0$). To reduce prediction time complexities, our main idea is to decompose the spatiotemporal cascade information (a larger size of data) to user characteristics (a smaller size of data) with bounded information loss; then, predictions are conducted based on the decomposed information, as to have a low time complexity.

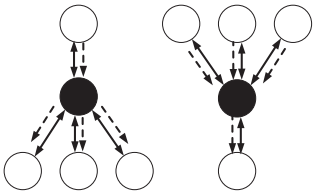
In a macro view, information cascades include a large amount of data that crosses



(a) A spatiotemporal cascade.

$$T = \begin{bmatrix} \infty & \infty & \infty & \infty & 5 \\ \infty & \infty & 2 & 3 & \infty \\ 4 & \infty & \infty & 3 & \infty \\ \infty & \infty & \infty & \infty & 5 \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

(b) The time matrix for (a).



(c) Characteristics of nodes.

$$u_1 = \begin{bmatrix} 0.00 \\ 0.83 \\ 0.56 \\ 0.00 \\ 0.00 \end{bmatrix} \quad v_1 = \begin{bmatrix} 0.26 \\ 0.00 \\ 0.57 \\ 0.78 \\ 0.00 \end{bmatrix}$$

(d) The decomposed vectors.

Figure 8.1: In (a) and (c), solid directional edges among nodes (numbers inside nodes are user IDs) represent follower-followee relationships (the pointed node is the follower). Dashed directional edges indicate the cascade. The label on the top of a node indicates the time when this user starts to propagate information after having been influenced. Node 2 is the information source. In (c), the left dark node has high persuasiveness and receptiveness (the right one is the opposite). The decomposition result for the cascade of the first four time slots is shown in (d).

both time and space. Therefore, we use matrices S and T to respectively capture the space and time dimensions of cascades. Here, S is the network adjacency matrix, which shows the social topology. Then, the time matrix T indicates the propagated nodes (i.e., users) in terms of time sequences. The time matrix T, which corresponds to the cascade of all five time slots in Fig. 8.1(a), is shown in Fig. 8.1(b). The element t_{ij} of T is the time when user j starts to propagate information after having been influenced by user i. We assume that a propagated node influences its followers immediately without a delay. Note that T includes complete time information and partial space information: nodes that are closer within the social topology are more likely to be propagated at closer times. Although S and T can be used for predictions directly, the prediction time complexity is unacceptable due to matrix operations. For example, in the Flickr dataset [103], S and T involve 2,302,925 users with 11,267,320

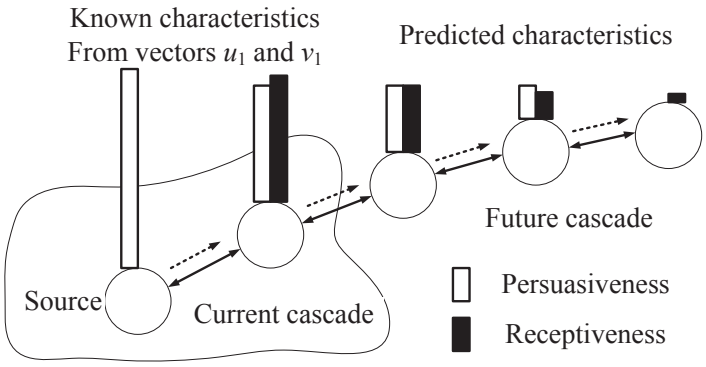


Figure 8.2: The decay pattern of nodes' persuasiveness and receptiveness.

photos, which are unacceptable for matrix operations.

The micro view of cascades is that followees iteratively propagate information to their followers. Therefore, we introduce *persuasiveness* and *receptiveness* to capture characteristics of followees and followers: the persuasiveness is defined as followees' abilities to propagate information; the receptiveness represents followers' willingness to accept information. As shown in Fig. 8.1(c), the left dark node has high persuasiveness and receptiveness (the right dark node is the opposite). Vectors u_1 and v_1 are used to record nodes' persuasiveness and receptiveness, where the i^{th} elements in the vectors u_1 and v_1 show node i's persuasiveness and receptiveness, respectively. We further consider these two characteristics to be *spatiotemporally-sensitive*: if a node with a high out-degree is spatially far away from the information source, it may not be propagated, and thus it cannot positively propagate the information further (i.e., low persuasiveness). In the case of a temporal remote node, it also has low persuasiveness, since its followers may have been propagated by other nodes. The same rule works for the receptiveness. Therefore, in terms of the distribution, nodes' persuasiveness and receptiveness should decay with respect to their spatiotemporal distances to the information source.

Our prediction scheme is based on both the macro and micro properties of cascades. This scheme has three stages as follows. (1) In the first stage (Section 8.4), we

map the time matrix, T, to a weighted matrix M. The mapping objective is to tune the weights of space and time information in T. We also highlight earlier propagations in the mapping process, since they are more important than later ones. (2) In the second stage (Section 8.5), we introduce the singular value decomposition (SVD [142]) to extract nodes' persuasiveness and receptiveness (two vectors) from the weighted matrix M, with bounded information loss. This is because the element m_{ij} of M represents a joint result of followee i's persuasiveness and follower j's receptiveness. Fig. 8.1(d) shows the result for the cascade of the first four time slots ($\tau_1 = 4$, and node 5 is waiting for the prediction) in Fig. 8.1(a). u_1 shows that nodes 2 and 3 are followees, while v_1 shows that nodes 1, 3 and 4 are followers. Now, the spatiotemporal cascade information (matrices) is compressed into nodes' persuasiveness and receptiveness (vectors), resulting in a reduced prediction time complexity. (3) In the third stage (Section 8.6), we conduct predictions based on the decomposed information. The decay pattern of nodes' persuasiveness and receptiveness along shortest paths are focused, as shown in Fig. 8.2. Then, the persuasiveness and receptiveness of currently unpropagated nodes are predicted. For example, in Fig. 8.1, node 5's persuasiveness and receptiveness are predicted according to vectors u_1 and v_1. Based on the prediction result, $\hat{u}_1$ and $\hat{v}_1$, we can reconstruct the predicted weighted matrix, $\hat{M}$. The predicted number of propagated users can be obtained by mapping $\hat{M}$ back to the predicted time matrix.

Our contributions are manifold: (1) we consider cascades spatiotemporally, and propose a parallel prediction scheme to deal with the large amount of cascade information. (2) We introduce persuasiveness and receptiveness to capture characteristics of followees and followers, which are completely novel. Persuasiveness and receptiveness can be decomposed from the spatiotemporal cascade information, i.e., the complete cascade information is compressed efficiently with bounds. (3) User personalities (e.g., gender and age) can be incorporated into our model. (4)

Prediction methods, based on nodes' persuasiveness and receptiveness, are proposed, the performance of which are verified by real-data driven evaluations.

The remainder of this chapter is organized as follows: In Section 8.2, we survey the related work; in Section 8.3, basic concepts are shown with the dataset description; in Section 8.4, we show the mapping process; in Section 8.5, the spatiotemporal decomposition is introduced to extract nodes' persuasiveness and receptiveness; in Section 8.6, we show the whole prediction process; in Section 8.7, real data-driven evaluations are shown; and finally, in Section 8.8, we conclude this chapter.

8.2 Related Work

An information cascade occurs when people observe the actions of others and then make the same choices that the others have made. The most popular cascade models include the linear threshold model [143, 144], and the independent cascade model [143, 144, 145, 146]. In the linear threshold model, each person has a weight and a threshold. A person starts to spread information further, only if the weight summation of propagated persons that he/she follows is larger than his/her own threshold. Instead of the deterministic model, the independent cascade model introduces probabilities: once propagated, each node has a certain likelihood of further spreading the information to its followers. More models are derived from these two models. For example, Ghasemiesfeh et al. [147] considers a k-complex model, where a node is further propagated if no less than k neighbors of this node are propagated. Sadikov et al. [148] considers a k-tree model. However, these models mainly focus on the spatial cascade information.

The study on spatiotemporal cascade has been proposed in [149], where the time dimension also matters. Differing from former studies, we compress the spatiotemporal cascade information into nodes' persuasiveness and receptiveness,

Table 8.1: Flickr Dataset Summary

Time period (two periods)	11/02/2006 to 12/03/2006 02/03/2007 to 05/18/2007
# Links	17,034,807 to 33,140,018
# Users	1,487,058 to 2,302,925
# Photos	11,267,320
# Favorite marks	34,734,221
# Popular photos	14,002
Most popular photo	Marked by 2,998 times
Largest in / out-degree	21,001 / 26,367

which are completely novel. This compression also sheds light on the big data processings [150], since it reduces the dimensions for describing cascades. Rather than using statistical approaches, our method reserves insights on cascades. Our model can also be extended by considering user personalities.

Another branch of cascade studies focuses on the data mining of real datasets, such as Facebook [151], Flickr [103] and Twitter [144]. These studies observe real cascades and then match real cascade properties to theoretical models [152, 153]. Our study is based on the Flickr dataset [103].

8.3 Basic Concepts and Dataset Description

8.3.1 Basic Concepts

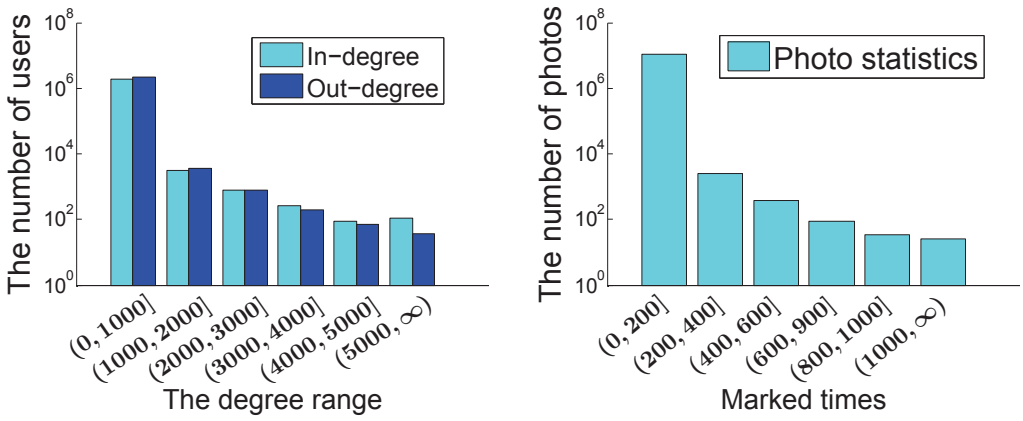
Flickr is an online social network for sharing photos (i.e., the information to propagate) among users, the relationships of which are directional: a directional edge from Bob to Alice means that Bob follows Alice. Users share photos among each other by labeling a "favorite-mark" to a photo. We refer to users who label photos with a "favorite-mark" as *propagated* users in the cascade of that photo. Meanwhile, users are called *influenced* if they have seen this photo. An influenced user may not be a propagated user, since he/she may not mark the photo as a favorite for further

Table 8.2: Notations

Notation	Description
τ_1 / τ_2	The current / future cascade time ($\tau_2 > \tau_1$).
τ_0	The appearance time of the information source.
S / T	The space / time matrix with elements s_{ij} / t_{ij}.
N_i / N_p / N	The set of influenced / propagated / total users.
E_i / E_p / E	The edge set corresponding to V_s / V_t / V.
M	A matrix mapped from the time matrix T.
U / Σ / V	The SVD result of M ($M = U\Sigma V^*$).
σ_i	The i^{th} largest singular value of the matrix M.
u_i / v_i	The vector in U / V corresponding to σ_i, and u_1 / v_1 shows persuasiveness / receptiveness.
$\hat{u}_1$ / $\hat{v}_1$	The predicted u_1 / v_1 in the future cascade.
$\hat{M}$ / $\hat{T}$	The predicted M / T in the future cascade.
$f(t) = e^{-ct}$	The mapping function, which maps t_{ij} to m_{ij}.

sharing. Then, a photo cascade process can be formally defined as a spatiotemporal photo spreading process on all influenced users, rather than on all propagated users. Information cascades include a large amount of data that crosses both time and space, i.e. spatiotemporal information. Then, the space matrix, S, is defined as the adjacency matrix of the social topology among all the users (including the users that need to be predicted). Theoretically, S should include the complete social topology (i.e., all users on Flickr), since a cascade may propagate over the whole network. For practical usage, S can be a large enough subgraph.

Once a user shares a photo, we consider that this user is influenced by all the propagated users that he/she follows. The element t_{ij} of matrix T is the time when user j starts to propagate information after having been influenced by user i. Here, T is called the time matrix, which includes all users corresponding to S. The elements in T that represent currently unpropagated users are set to be infinite. The users in T are corresponding to the users in S. We will further discuss the size of S and T, since including all users is redundant, and is not feasible for practical usage. A large enough subgraph can be used for the prediction. Note that a user j may have



(a) User degree distribution.

(b) Favorite mark distribution.

Figure 8.3: Statistics of the Flickr dataset.

been influenced by multiple users before his/her own propagation at the time t_{ij}. For example, in Fig. 8.1(a), user 5 has been influenced by user 4 since time 3, but he/she finally decides to propagate (i.e., label a "favorite-mark" to the photo) at time 5. Note that time durations of influences can be deducted from T. Therefore, complete information of a photo cascade has been reserved in both S and T. In addition, let τ_0, τ_1 and τ_2, respectively, denote the appearance time of the information source, the current time (i.e., we know the whole cascade process between τ_0 and τ_1), and the future time at which we want to predict the cascade size. In the following cascade examples of this chapter, we set $\tau_0 = 0$, $\tau_1 = 4$, and $\tau_2 = 5$ as a default setting.

8.3.2 Dataset Description

The Flickr dataset is collected by Cha et al. [103] through Flickr APIs. It was collected during the time periods from November 2^{nd} to December 3^{rd}, 2006, and February 3^{rd} to May 18^{th}, 2007. The number of users and their links are growing with respect to time. Note that a user, on average, has less than 15 links: this network is definitely *sparse* (i.e., matrices S and T are sparse). The degree distribution is shown in Fig. 8.3(a), indicating that a few users have very high degrees. 11,267,320 photos

are shared during this period, with 34,734,221 favorite marks in total. 34,484 photos are not marked, but are recorded in the system. Most photos (11,218,834 photos) are marked no more than 100 times, while only 25 photos are marked more than 1,000 times. The distribution of photos, in terms of times marked, is shown in Fig. 8.3(b). Since photos of different popularity stand for cascades of different types, we choose *popular photos* (defined as the photos that are shared more than 100 times) for further analysis in the following part. We consider that popular photos have similar cascade properties. The other dataset statistics are shown in Table 8.1, and all the notations are shown in Table 8.2.

8.4 Tuning The Spatiotemporal Information

The first stage of our prediction scheme is introduced in this section, where we show the mapping process that tunes the weights of space and time information.

8.4.1 Mapping Process

We independently map each element in T to the element in M of the same position. Then, the mapping function is defined as $f : t_{ij} \rightarrow m_{ij}$, or $m_{ij} = f(t_{ij})$, over real positive numbers. Since earlier propagations are more important, we have the following mapping rule:

Principal 8.1. *The function $f(t)$ is strictly decreasing with respect to t. When $t \rightarrow \infty$, we have $f(t) \rightarrow 0$.*

Another concern is that the starting time of the cascade is not important: the cascade in Fig. 8.1(a) can be viewed as starting at $\tau_0 = 0$ and finishing at $\tau_2 = 5$; however, it can also be viewed as starting at $\tau_0 = 1$ and finishing at $\tau_2 = 6$ (i.e., a position translation of 1 on the time domain). Obviously, this translation should not influence mapping results (relationships among elements m_{ij}). Therefore, we have:

$$T(\tau_1 = 4) = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 3 & \infty \\ 4 & \infty & \infty & 3 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \quad M(\tau_1 = 4) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.67 & 0.55 & 0 \\ 0.45 & 0 & 0 & 0.55 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) The time matrix at $\tau_1 = 4$. (b) The corresponding mapping result.

Figure 8.4: Mapping T to M through $f(t) = e^{-t/5}$, where $c = \frac{1}{5} = \frac{1}{\tau_2}$.

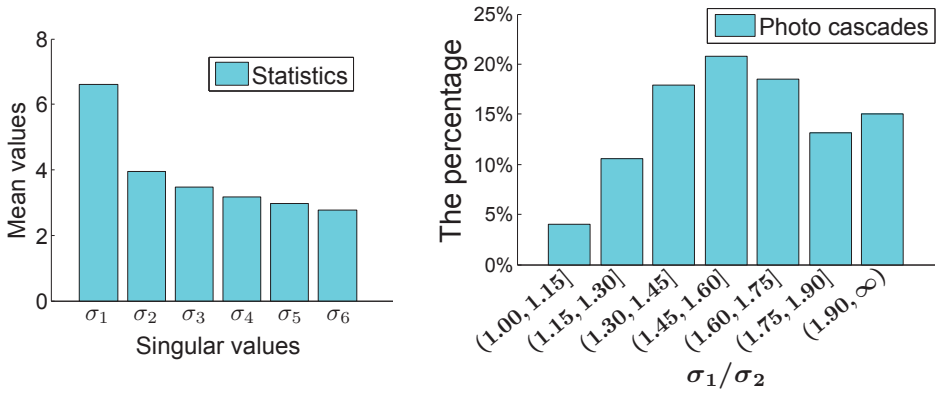
Principal 8.2. *The function $f(t)$ satisfies $\frac{f(t+\tau)}{f(\tau)} = f(t)$, i.e., $f(t+\tau) = f(t)f(\tau)$. Here, τ is a parameter for tuning the starting time of the cascade.*

An interesting phenomenon is that Guiding Rules 8.1 and 8.2 have determined the function form of $f(t)$, as follows:

Theorem 8.3. *The only feasible family of solutions for the above guiding rules are exponential functions, i.e., $f(t) = e^{-ct}$ where c is an arbitrary positive number.*

Proof: Let us start with Guiding Rule 8.2, where $f(t+\tau) = f(t)f(\tau)$. If $\tau = t$, then $f(t+t) = f(2t) = f(t)^2$. If we do this iteratively, then we can have $f(Ct) = f(t)^C$, where C is a parameter. Exchanging t and C, we have $f(Ct) = f(t)^C = f(C)^t$. Let $C = 1$, and then we have $f(t) = f(1)^t$. Obviously, $f(1)$ is an arbitrary constant. If we replace $f(1)$ with e^c, then the result is $f(t) = e^{ct}$. Here, c is an arbitrary real number. According to Guiding Rule 8.1, the function $f(t)$ is strictly decreasing. Therefore, we change the result $f(t) = e^{ct}$ to be $f(t) = e^{-ct}$, and restrict c to be a positive number. In addition, the result can also be proved through (1) operating a logarithm on $f(t+\tau) = f(t)f(\tau)$ to be $\ln f(t+\tau) = \ln f(t) + \ln f(\tau)$, and then (2) using Cauchy's functional equation. □

Here, parameter c's insight is its functionality for tuning time scales (e.g., 1 hour is equivalent to 60 minutes): time scales should not change the mapping result. Generally speaking, the value of c is determined empirically. The value c can be set in the range of $[\frac{1}{\tau_2}, \frac{1}{\tau_1}]$. In addition, the corresponding mapping process of the cascade (only the first four time slots) in Fig. 8.1(a) is shown in Fig. 8.4.



(a) Statistics on σ_i.

(b) σ_1/σ_2 of photo cascades.

Figure 8.5: Statistics on singular values of photo cascades.

8.4.2 Mapping Insights

As previously mentioned, we consider cascades spatiotemporally, which includes a large amount of data that crosses both space and time. Meanwhile, the persuasiveness and receptiveness is used to capture characteristics of followees and followers: the persuasiveness includes followees' capacities to propagate information; the receptiveness includes followers' willingness to accept information. Nodes' persuasiveness and receptiveness are considered to be spatiotemporally-sensitive: if a node with a high out-going degree is spatially far away from the information source, it may not be propagated, and thus it cannot positively propagate the information further. A temporal remote node also has low persuasiveness, since its followers may have been propagated by other nodes. The same rule works for the receptiveness: propagations that fail to reach the sources' neighbors may lead to a premature abortion of further information propagations; a successful propagation of a remote node does not change the overall cascade trend. Therefore, in terms of the distribution, nodes' persuasiveness and receptiveness should decay with respect to their spatiotemporal distances to the information source, as previously mentioned in Fig. 8.2. That is the reason why we highlight earlier propagations in the Guiding Rule 1. Moreover, the decay pattern of nodes' persuasiveness and receptiveness reveals

$$U = \begin{bmatrix} \mathbf{0.00} & 0.00 & 1.00 & 0.00 & 0.00 \\ \mathbf{0.83} & -0.56 & 0.00 & 0.00 & 0.00 \\ \mathbf{0.56} & 0.83 & 0.00 & 0.00 & 0.00 \\ \mathbf{0.00} & 0.00 & 0.00 & -1.00 & 0.00 \\ \mathbf{0.00} & 0.00 & 0.00 & 0.00 & -1.00 \end{bmatrix} \quad \Sigma = \begin{bmatrix} \mathbf{0.98} & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.55 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$

$$V = \begin{bmatrix} \mathbf{0.26} & 0.68 & 0.69 & 0.00 & 0.00 \\ \mathbf{0.00} & 0.00 & 0.00 & -1.00 & 0.00 \\ \mathbf{0.57} & -0.68 & 0.46 & 0.00 & 0.00 \\ \mathbf{0.78} & 0.27 & -0.56 & 0.00 & 0.00 \\ \mathbf{0.00} & 0.00 & 0.00 & 0.00 & -1.00 \end{bmatrix}$$

Figure 8.6: The corresponding SVD result (U, Σ, and V) for the mapped matrix M in Fig. 8.4(b).

boundaries for further propagations: the cascade terminates when nodes have low persuasiveness and receptiveness.

Let us go back to the element m_{ij} in M. Obviously, m_{ij} is a joint result of followee i's persuasiveness and follower j's receptiveness. Note that a larger value of m_{ij} means an earlier propagation, i.e., a larger persuasiveness of followee i, and a larger receptiveness of follower j. Meanwhile, matrices T and M have included complete time information and partial space information of the cascade: nodes that are closer within the social topology are more likely to be propagated at closer times. Now, the parameter c in the mapping function $f(t) = e^{-ct}$ has another insight meaning: it balances the weights of space and time information. If $c \to 0$, then M is composed of zeros and ones: we only focus on the space information, regardless of time sequences. On the other hand, if c is large, the time information is highlighted. Therefore, M can be viewed as a tuned spatiotemporal information matrix. In the next section, we show the decomposition process through SVD operations, where we extract nodes' persuasiveness and receptiveness from the tuned spatiotemporal information matrix.

$$M = \begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.67 & 0.55 & 0.00 \\ 0.45 & 0.00 & 0.00 & 0.55 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$

$$M_1 = \sigma_1 u_1 v_1^* = \mathbf{0.98} \cdot \begin{bmatrix} \mathbf{0.00} \\ \mathbf{0.83} \\ \mathbf{0.56} \\ \mathbf{0.00} \\ \mathbf{0.00} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{0.26} \\ \mathbf{0.00} \\ \mathbf{0.57} \\ \mathbf{0.78} \\ \mathbf{0.00} \end{bmatrix}^* = \begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.21 & 0.00 & 0.46 & 0.63 & 0.00 \\ 0.14 & 0.00 & 0.31 & 0.42 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$

Figure 8.7: The rank-1 approximation of the matrix M. There is a bounded information loss from M to M_1.

8.5 Spatiotemporal Decomposition

The second stage of our prediction scheme is shown in this section, where we introduce the SVD operation on the weighted matrix M to extract information on nodes' persuasiveness and receptiveness.

8.5.1 SVD Preliminaries and Dataset Verification

In the SVD, M is factorized to a product of three matrices: U, Σ, and V ($M = U\Sigma V^*$, where $*$ is a transpose). The matrix Σ is a diagonal matrix with nonnegative real numbers on the diagonal (generally in descending order), while the diagonal entries σ_i of Σ are known as the singular values of M. The number of singular values equal the matrix rank of M. We focus on the SVD's functionality of low-rank approximations, i.e., the matrix M is approximated by vectors. Let u_i and v_i denote the i^{th} columns of matrices U and V, respectively. Assuming r is the rank of M, we select the largest k ($k < r$) singular values to approximate M:

$$M_k = \sum_{i=1}^{k} \sigma_i u_i v_i^* \tag{8.5-1}$$

where M_k is the approximated M through the k largest singular values. Moreover, the difference between matrices M_k and M is bounded by $||M_k - M||_2 = \sigma_{k+1}$, where $|| \cdot ||_2$ denotes the 2^{nd} order Frobenius norm. In addition, M can also be accurately represented as $\sum_{i=1}^{r} \sigma_i u_i v_i^*$.

We then conduct experiments on the Flickr dataset, as to verify the effectiveness of this decomposition. The corresponding time matrices of popular photos (i.e., photos that are shared more than 100 times) are mapped by $f(t) = e^{-ct}$ with the parameter c as the reciprocal of the cascade duration, i.e., $c = 1/(\tau_2 - \tau_0)$. Singular values are averaged with respect to different photos, and the result is shown in Fig. 8.5(a). It can be seen that the difference of σ_1 and σ_2 is much larger than the differences of other consecutive singular values (such as σ_2 and σ_3). The distribution of σ_1/σ_2 is shown in Fig. 8.5(b). It means that the main pattern of M is highlighted (note that $||M_k - M||_2 = \sigma_{k+1}$), i.e., the cascade information is greatly concentrated in σ_1 and its corresponding vectors (u_1 and v_1). In addition, for further analysis, the corresponding SVD for the mapped matrix M in Fig. 8.4(b) is shown in Fig. 8.6. The low rank ($k = 1$) approximation of M is shown in Fig. 8.7.

8.5.2 Information Decomposition

As discussed in Section 8.3, the matrices T and M are sparse matrices, which in general have low ranks. The relationship between matrix sparsity and rank has been studied in [154]. Moreover, experiments in Fig. 8.5 show that the largest singular value is a concentration of the cascade information. Therefore, we use $\sigma_1 u_1 v_1^*$ (i.e., M_1) as the compressed cascade information for further processing: predictions are based on u_1 and v_1. Note that this information compression has limited information loss. We can also use more singular values (rather than only using σ_1), which can bring more accurate predictions at the cost of higher time complexities (a tradeoff

between accuracy and complexity). Let $\hat{u}_1$ and $\hat{v}_1$ denote the predicted vectors in the future cascade, then M can be reconstructed through $\hat{M} = \sigma_1 \hat{u}_1 \hat{v}_1^*$. The predicted number of propagated users can be obtained by mapping M back to the predicted time matrix (i.e., reconstruction).

The decomposition can reduce the difficulties of cascade predictions, since it reduces dimensions for describing cascades. Spatiotemporal cascades are compressed. Instead of matrix operations, vector operations are used to reduce prediction time complexities. According to [155], a centralized SVD of an r-rank $n \times n$ matrix takes a time complexity of $O(rn^2)$.

Moreover, vectors u_1 and v_1 have their insights: u_1 shows nodes' persuasiveness; v_1 represents nodes' receptiveness. As previously mentioned, m_{ij} is a joint result of followee i's persuasiveness and follower j's receptiveness. Meanwhile, the element corresponding to m_{ij} in $\sigma_1 u_1 v_1^*$ (i.e., M_1) is the product of the i^{th} element in u_1 (persuasiveness) and the j^{th} element in v_1 (receptiveness). Note that a larger value in u_1 and v_1 means a larger persuasiveness and receptiveness, respectively, since they would lead to an earlier propagation, i.e., a larger corresponding element in M. The example in Fig. 8.7 shows the SVD for the cascade in Figs. 8.1 and 8.4, while u_1 and v_1 have been shown in Fig. 8.1(d). Note that only the information on the first four time slots is available now, and we want to predict the cascade of the following time slots. As mentioned in Fig. 8.1(d), u_1 shows that nodes 2 and 3 are key spreaders, which conforms to Fig. 8.1(a). v_1 shows that nodes 3 and 4 are more important receivers than node 1, which is also consistent with Fig. 8.1(a).

8.5.3 SVD Insights and Personalities

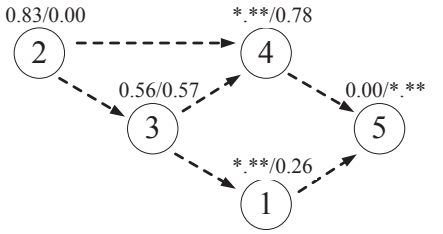
As previously mentioned, the cascade information is greatly concentrated with respect to the largest singular value, while σ_1 is almost twice that of σ_2 in Fig. 8.5. A

reasonable explanation for this phenomenon is that *each singular value represents a cascade mode*: σ_1 shows a general global mode, e.g., almost all users enjoy beautiful high-definition photos rather than normal low-definition ones; σ_2 shows a popular mode, e.g., lots of users share beautiful high-definition photos on landscapes; σ_3 shows a comparatively local mode, e.g., a small group of users like landscape photos on mountains; so on so forth. SVD extracts global and common photo cascade modes into larger singular values, while it leaves local and personal photo cascade modes as smaller singular values.

Therefore, our scheme can utilize the information on user personalities to pursue better performances. Let $\bar{\sigma}$, $\bar{u}$, and $\bar{v}$ respectively denote the weight, the additional persuasiveness, and the additional receptiveness brought by user personalities. $\bar{\sigma}$, $\bar{u}$, and $\bar{v}$ of each user can be concluded from the gender, the age, the total number of shared photos, the total online time, and so on. Then, we can revise our prediction through $\hat{M} = \sigma_1 \hat{u}_1 \hat{v}_1^* + \bar{\sigma}\bar{u}\bar{v}^*$. Therefore, our model can easily be extended by considering user personalities.

8.5.4 Parallel SVD

Another advantage of our scheme is its parallelism. First, mapping matrices T to M can be done in parallel, since mapping elements in T are independent of each other. Meanwhile, SVD also has parallel methods [156, 157]. Given p processors, SVD of a $n \times n$ matrix can be done [156] within a time complexity of $O(n^3/p)$. The centralized method takes $O(rn^2)$, where r is the rank of the matrix. Note that both the centralized and distributed methods target the complete SVD, while we only need the largest singular value σ_1 and its corresponding vectors. Therefore, there exist possibilities to further reduce time complexities. Since SVD is a standard tool, we do not focus on further improving its efficiency.



(a) Persuasiveness and receptiveness.

$$\hat{u}_1 = \begin{bmatrix} \mathbf{0.37} \\ 0.83 \\ 0.56 \\ \mathbf{0.83} \\ 0.00 \end{bmatrix} \quad \hat{v}_1 = \begin{bmatrix} 0.26 \\ 0.00 \\ 0.57 \\ 0.78 \\ \mathbf{0.78} \end{bmatrix}$$

(b) Predicting u_1 and v_1 in (a).

Figure 8.8: The corresponding nodes' characteristics of Fig. 8.7. In (a), dashed directional edges show the cascade process, while numbers within nodes are their IDs. Labels on top of the nodes are persuasiveness/receptiveness, which are extracted from u_1 and v_1 in Fig. 8.7. The symbol *.** means needs to be predicted, the results of which are shown in bold font in (b).

8.6 Information Cascade Prediction

The third stage of our prediction scheme is described in this section, where we conduct predictions through extracting patterns of u_1 and v_1. Then, we construct the spatiotemporal cascade information as the final prediction.

8.6.1 Non-historical Prediction

In this subsection, we predict $\hat{u}_1$ and $\hat{v}_1$ based on the current cascade (called non-historical prediction), i.e., the historical data of former cascades is not utilized. The persuasiveness and receptiveness of unpropagated nodes are predicted based on their *shortest path* to the information source. Here, nodes' persuasiveness and receptiveness are considered as node weights in the shortest path algorithm, while all edge weights are 0. Nodes with known persuasiveness (non-zero elements in u_1) use their persuasiveness as node weights, while nodes with unknown persuasiveness (i.e., need to be predicted) use constant units as their weights. The receptiveness predictions are similar. The reason for the shortest path is that it has a relatively-high probability (among all paths) of gradually propagating the information from the source to the node. Note that, an information source's receptiveness is 0 (it only

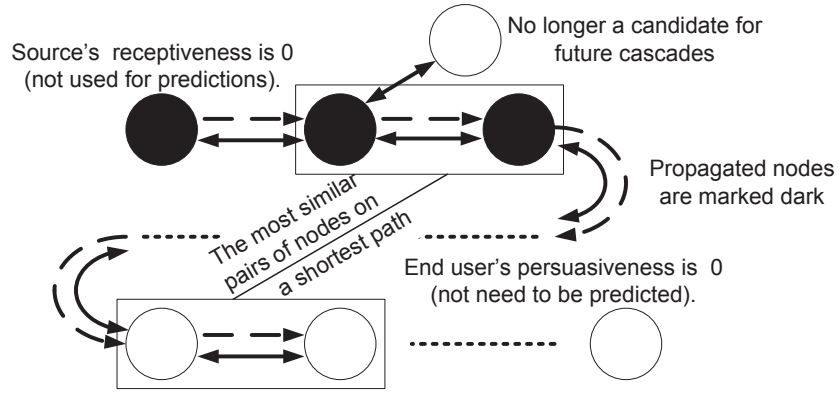


Figure 8.9: Three rules for non-historical predictions.

spreads the information out), and the persuasiveness of the end user of a propagation is also 0 (it only receives the information without further propagations).

A simple but effective method is to use the decay of propagated nodes' persuasiveness and receptiveness for predicting that of unpropagated nodes, and an example is shown in Fig. 8.8, which corresponds to the example in Fig. 8.7. In Fig. 8.8(a), the labels on top of nodes represent their persuasiveness and receptiveness (extracted from u_1 and v_1), where the symbol *.** means needs to be predicted. Let us start with the persuasiveness of node 4. Its shortest path to the information source is from node 4 to node 2 directly; therefore, 0.83 is predicted as the persuasiveness of node 4, since no decay pattern exists on this path. Then, the persuasiveness of node 1 can be calculated through the path of nodes 2, 3, and 1. The persuasiveness decay from node 2 to node 3 is $\frac{0.56}{0.83}$, therefore, node 1's persuasiveness is predicted as $\frac{0.56}{0.83} \times 0.56 = 0.37$. Note that, node 5's persuasiveness is predicted to be 0, since it is the end of a propagation path. As for node 5's receptiveness, it is predicted through the path of nodes 2, 4, and 5. Since the source only spreads information, node 5's receptiveness is predicted to be the same as node 4's receptiveness. The prediction results of $\hat{u}_1$ and $\hat{v}_1$ are shown in Fig. 8.8(b). Then, $\hat{u}_1$ and $\hat{v}_1$ are normalized to be [0.27,0.61,0.41,0.61,0.00]* and [0.20,0.00,0.45,0.62,0.62]*, respectively. Then, we

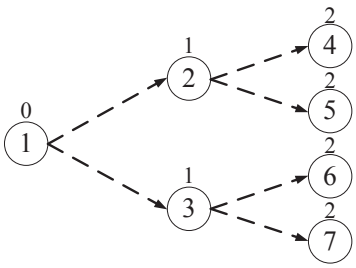


Figure 8.10: A case study (the same notation with Fig. 8.1).

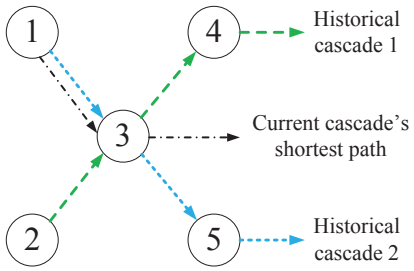


Figure 8.11: Historical predictions.

reconstruct $\hat{M} = \sigma_1 \hat{u}_1 \hat{v}_1^*$. The predicted time matrix, $\hat{T}$, can then be obtained through mapping $\hat{M}$ back. Elements $\hat{t}_{15} = 5$ and $\hat{t}_{45} = 9$ in $\hat{T}$ show two predicted propagation times of node 5. We use the minimum values of 5 and 9 as the final prediction (i.e., node 5 will be propagated at time 5), which is the same as the actual cascade in Fig. 8.1(a).

In the above example, we have not considered the case where the length of the shortest path is longer than three. Instead of using the averages of former decays, we use *the decay of the most similar pair of former nodes* for predictions in a shortest path with larger length. The similarity is defined as *the summation of squared social topological degree differences of followees and followers*. Here, the degree can be either in-degree, out-degree, or both. If the path length is too short to extract the decay information, the followee's persuasiveness and receptiveness are used directly, as shown for predicting node 4's persuasiveness in Fig. 8.8. The pairwise similarities enable different followers of the same followee to have different predictions. To further

reveal directions of cascades, unpropagated nodes within a certain number of hops to the information source are kicked out. In other words, unpropagated nodes near to the source are not receptive, and thus they are no longer candidates for future cascades. This hop-count threshold is empirically determined based on the number of nodes currently propagated. Rules for non-historical predictions are shown in Fig. 8.9 (currently propagated nodes are marked dark while the remaining nodes are unpropagated at present) and are summarized as follows:

- The information source's receptiveness is 0, and is not used for receptiveness predictions along the shortest path. Meanwhile, the persuasiveness of the user at the end of the shortest path is fixed to be 0.
- Predictions are based on shortest paths. Along a shortest path, the persuasiveness and receptiveness decay between a pair of nodes are predicted as the corresponding decay between the most similar (in terms of degree differences) pair of currently propagated nodes. Nodes' persuasiveness and receptiveness can be derived from decays.
- Unpropagated nodes within a certain number of hops to the information source are kicked out for being propagated in the future. They are not receptive, and thus, are no longer candidates for future cascades, i.e., they are influenced by the cascade without further propagations (influenced but not propagated).

A learning process on the pattern of nodes' persuasiveness and receptiveness should bring a better prediction. However, it also has a higher time complexity as a tradeoff. Since the current method has obtained a good result, we do not further explore learning-based methods. To better understand decay patterns, a case study on "branching" cascades is conducted, as shown in Fig. 8.10. This type of cascade spreads without resistances, where the number of propagated nodes increases exponentially. Assuming the usage of $c = \frac{1}{2}$ for the mapping process, the

decomposition result for the cascade in Fig. 8.10 is $u_1 = [1, 0, 0, 0, 0, 0, 0]^*$ and $v_1 = [0, 0.71, 0.71, 0, 0, 0, 0]^*$. In other words, the cascade is compressed into relationships among nodes 1, 2, and 3, since the later cascade repeats their propagation mode. Therefore, the pattern of this "branching" cascade can be captured.

8.6.2 Historical Prediction

In the previous subsection, we predicted a cascade without historical information. Predictions are conducted based on former cascades, i.e., the decay patterns of nodes' persuasiveness and receptiveness in former cascades are used (called historical prediction). The prerequisite of historical predictions is that former cascades are homogenous with the current one: cascades of popular photos are different than unpopular ones; therefore, we should not use the historical data on cascades of popular photos to predict cascades of unpopular ones.

Shortest paths of the current cascade are cooperatively used with the historical data. Instead of calculating decays of nodes' persuasiveness and receptiveness based on currently propagated nodes, we use decays of former cascades as predictions. An example is shown in Fig. 8.11: the black dashed directional path indicates a shortest path of the current cascade. The historical decay of cascade 2 is used to predict the decay from node 1 to 3, while cascade 1 is not used, since it does not have an intersection with the decay from nodes 1 to 3. In the case of multiple available historical cascades, the decay of the current cascade is predicted to be their average decay.

8.6.3 Algorithm Complexities

As previously mentioned, S and T include all users in the network. However, this is unnecessary, since most cascades only influence a very small portion of users in the

network. Therefore, for practical usage, we can have a subgraph just large enough for predictions.

We have used shortest paths with node weights in predictions; however, this can be solved by slightly modifying Dijkstra's algorithm. Therefore, it has the same time complexity with the normal Dijkstra's algorithm. Let N_p and N ($N_p \ll N$), respectively, denote the number of currently propagated nodes and total nodes (E_p and E to represent the number of corresponding edges). Then, the centralized Dijkstra's algorithm takes $O(E + N \log N)$ through a Fibonacci heap. Calculating decays (and nodes' persuasiveness or receptiveness) can follow the same order of the shortest path. Since the path length is bounded by the network diameter D, the decay calculation takes at most $O((E + N \log N)D)$. The mapping and its reversion (mapping $\hat{M}$ back to $\hat{T}$) takes $O(N^2)$. The centralized SVD takes $O(rN_p^2)$, where r is the rank of M. Here, we do not need $O(rN_p^2)$ for the SVD, since the decomposition results for currently unpropagated nodes are useless; the persuasiveness and receptiveness corresponding to unpropagated nodes are 0, and we only need to decompose the cascade information among currently propagated nodes. Therefore, the total time complexity is $O(N^2 + DN \log N + rN_p^2)$ in a centralized calculation method for a sparse graph.

According to [158], Dijkstra's algorithm can be done in parallel. The idea is to divide the graph into pieces for each processor. Given p processors, the time complexity can be brought down to $O(N^3/p)$ (a more accurate description is given in [158]). The SVD takes $O(N_p^3/p)$. Mapping and its reversion can be solved in parallel, since each element's map is independent from the others. So the total time complexity of our scheme is $O((N^3 + N_p^3)/p)$ in parallel.

If we conduct predictions on S and T directly, then the time complexity will not be acceptable. For each unpropagated node, we need to scan and process

S and T, which takes at least $O(N^2)$. Therefore, at least $O(N^3)$ is needed for a centralized method. Even if direct predictions can be implemented in parallel, they should have a higher time complexity than $O(N^3/p)$ due to the overhead. Meanwhile, our decomposition method has compressed all cascade information into nodes' persuasiveness and receptiveness with limited information loss, resulting in a reduced time complexity.

8.7 Evaluation

This section conducts evaluations. After presenting the basic settings, we show baseline algorithms and evaluation metrics. Finally, the evaluation results are shown from different perspectives to provide insightful conclusions.

8.7.1 System Settings

Our evaluations focus on cascades of popular photos that are marked "favorite" more than 100 times, since photos of different levels of popularity stand for cascades of different types. However, each photo may be involved in multiple cascades: unconnected users may share the same photo coincidentally, leading to different cascades in the network. Therefore, for each popular photo, we select its earliest cascade (in the sense of the earliest appearance time of the information source). The earliest cascade is generally the largest one.

For a photo cascade predition, it is almost impossible to take all unpropagated nodes into consideration, since we have 2,302,925 users in total. Meanwhile, only 25 photos are propagated over more than 1,000 users. Therefore, a subgraph is expected to improve the prediction efficiency. This subgraph is constructed as a combination of (1) all propagated users of the earliest cascade of the photo, (2) three random

out-going neighbors for each of these propagated users, and (3) all social topology relationships between the above users.

The input parameters of our prediction scheme include the complete cascade information (from the time τ_0 to τ_1), while we will predict the size of the cascade at its finishing time τ_2. In the following experiments, we will tune the ratio of τ_1/τ_2 to observe the performance of our scheme. This value stands for the amount of prior knowledge, i.e., a larger τ_1/τ_2 should bring a better prediction. In addition, the parameter c is set to be $1/\tau_1$, while the hot count threshold is empirically set to be 3. As for user personalities, we use normalized out-degree and in-degree to respectively describe $\bar{u}$ and $\bar{v}$, while we set $\bar{\sigma}$ to be $0.1 \times \sigma_1$.

8.7.2 Baseline Algorithms and Evaluation Metrics

Three baseline algorithms (the first two prediction schemes are non-historical schemes, while the last scheme is a historical scheme) are used for comparison as follows. (1) Largest in-degree: among all unpropagated nodes, the node with the largest in-degree (in terms of the social topology) is considered to be the next propagated node. The propagation time delay is considered as the largest propagation time delay of the current cascade. This scheme is based on the observation that a user with a larger in-degree is more likely to accept new information. (2) Most influenced: among all unpropagated nodes, the node that has the largest number of incoming propagated neighbors is considered to be the next propagated node. The propagation time delay is also the largest propagation time delay of the current cascade. This scheme is based on the observation that a user with more in-neighbors in the cascade is more likely to be influenced. (3) Most active: among all unpropagated nodes that are outgoing neighbors of propagated nodes, the node that is the most active, in terms of having been propagated by former cascades for the most number of times, is

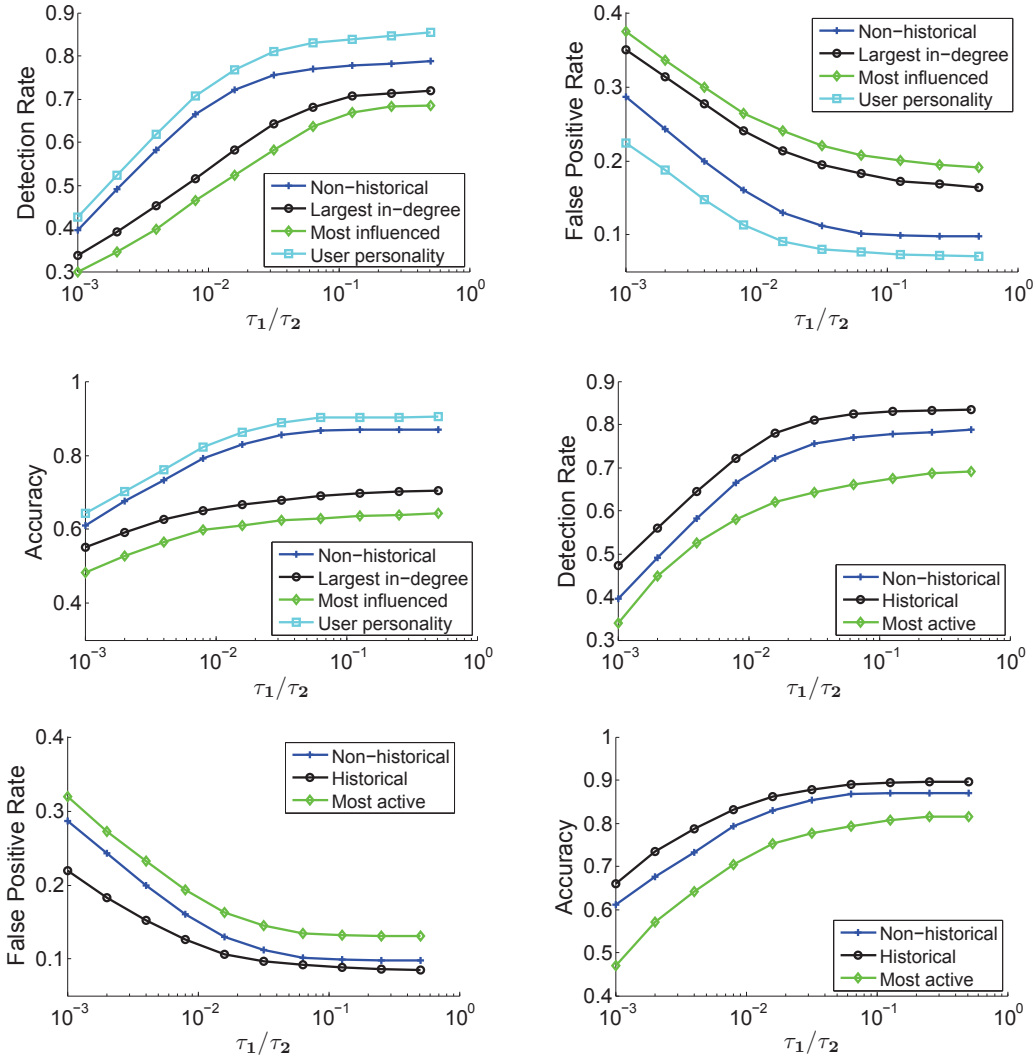


Figure 8.12: The evaluation results. The top row shows non-historical prediction schemes (The algorithm "User personality" is the proposed non-historical scheme with additional considerations on user personalities), while the bottom row consists of historical prediction schemes. Note that the history information has included the information on user personalities. Each of the three columns indicates one of the three metrics (detection rate, false positive rate, accuracy).

considered to be the next propagated node. The propagation time delay is calculated as the historical delay.

As for the evaluation metrics, the standard Receiver Operating Characteristic (ROC) metrics [159] are employed, including the detection rate (the higher the better),

the false positive rate (the lower the better), and the accuracy (the higher the better). More details can be found in [159].

8.7.3 Evaluation Result

The evaluation result is shown in Fig. 8.12, in terms of non-historical (top row) and historical (bottom row) prediction schemes. Each column corresponds to one of the three ROC metrics. For the non-historical schemes, the proposed algorithm outperforms the two naive baselines, among all three metrics. This is beacuase our algorithm considers spatiotemporal information, while the two naive algorithms mainly focus on the space information. Overall, our algorithms get about 20% higher accuracy than the two baselines. Another observation is that all these schemes have diminishing return effects: the increasing rate of the accuracy decreases with respect to τ_1/τ_2. This is because the early propagations are more important and more deterministic for the future trend of the cascade, and thus the amount of information contributed by a early propagation is larger than that by a late propagation. The initial information helps predict the cascade framework, while the following information just fulfills predicting details of the cascade. The prediction gain is marginal when $\tau_1/\tau_2 \geq 0.1$.

As for the historical predictions (bottom line), it can be seen that they perform better than non-historical schemes, since additional information is utilized. Meanwhile, the baseline algorithm (i.e., most active) does not have a very good performance, since it relies on the user histories too much, without considerations of the spatiotemporal propagations of the current cascade. Our algorithm extracts users' persuasiveness and receptiveness from former cascades, and then combines that information with the spatiotemporal information of the current cascade to obtain a better result. It can been seen that the historical prediction has an accuracy of about

0.9 when $\tau_1/\tau_2 = 0.1$. The corresponding detection rate and false positive rate is more than 0.8, and less than 0.1, respectively.

8.8 Summary

Information cascade predictions are important, due to their functionalities of detecting bad cascades. Given the current cascade and the social topology, we want to predict the cascade size at a future time slot. A cascade can be described by space and time dimensions. The SVD operation is used to decompose the spatiotemporal cascade information into the users' persuasiveness and receptiveness. Predictions are conducted based on the decomposed information, as to have a low time complexity. User personalities can also be incorporated into our scheme. Furthermore, our prediction scheme can be implemented in parallel. Finally, real-data driven evaluations verify the competitive performance of the proposed scheme.

CHAPTER 9

CONCLUSION

9.1 Summary of Results

The components of complex network systems satisfy a scale-free architecture, in which the node degree distribution follows power-law. While social networks are generally scale-free, it is natural to utilize their structural properties in some social network applications. As a result, this dissertation explores social network architectures, and in turn, leverages these architectures to facilitate some influence and information propagation applications. Social network architectures are analyzed in two aspects. The first aspect focuses on the node degree snowballing effects, based on an age-sensitive preferential attachment model. The second aspect studies NSFAs for social networks. 'Nested' indicates that the scale-free architecture is preserved when low-degree nodes and their associated connections are removed. NSFAs indicate that social networks have an onion architecture. Based on the social network structure, this dissertation explores two influence propagation applications for the SIMP. The first application is a friend recommendation strategy with the perspective of social influence maximization. The second application focuses on the SIMP with the crowd influence, since the crowd influence surpasses the combination of the independent influence from each person in the crowd. Beside the influence propagation applications, this dissertation further explores three different information propagation applications (epidemic propagation, trust propagation, and cascade propagation). The first application is a social network quarantine strategy, which can

eliminate epidemic outbreaks with minimal isolation costs. The second application is a rating prediction application, called DynFluid, based on the fluid dynamics. The third application is a cascade prediction application: given the social current cascade and social topology, the number of propagated users at a future time slot is predicted through decomposing the spatiotemporal cascade information. Real data-driven experiments demonstrate the efficiency and effectiveness of our applications.

9.2 Future Research

Our future research directions can be rich. One of the most important direction is to further understand the dynamics of social networks over time. For example, how does the social network architecture changes over time, especially when people join and leave? Will the social network properties, such as diameter, clustering coefficient, and scale-free exponent, change over time? Moveover, different people have different impacts over time. What happens if a high-degree node leaves the social network? In contrast, what happens if a low-degree node leaves the social network? The above questions need to be explored.

Another future research direction focuses on the influence and information propagation models. For the influence propagation models, this dissertation uses the independent cascade model, while the linear threshold model is not explored. Can our results be generalized to both independent cascade and linear threshold models? One step further, which property of the influence propagation model can be incorporated into our results? The impact of the crowd influence needs to be further studied in other influence propagation models, especially for non-submodular models. For the information propagation models, this dissertation only focuses on the epidemic propagation, the trust propagation, and the cascade propagation. What are the other models for the information propagation? Note that the information

propagation models heavily depends on the type of the information, i.e., different types of information can have completely different propagation models.

Finally, our future work can involve more practical implementations, especially for OSNs such as Facebook and Twitter. More real data can be collected from the company. Meanwhile, user surveys can be very helpful to understand how the influence and information propagate among users. User characteristics (e.g., age, address, education level, interest, marriage status, and so on) can be critical. Given a specified type of influence or information, which user characteristic is more important with respect to the propagations? Can we predict the user's behavior upon receiving the influence or the information? Practical implementations in the company can help to figure out the answers.

PUBLICATIONS

Conference:

1. **H. Zheng** and J. Wu, Social Influence Maximization in Hypergraphs: Non-Submodularity and Approximability, submitted to the IEEE International Conference on Computer Communications (INFOCOM), 2018.

2. J. Wu, Y. Chen, and **H. Zheng**, An Approximation for Dependency-Aware Rule-Caching in Software-Defined Networks, submitted to the IEEE International Conference on Communications (ICC), 2018.

3. J. Wu, S. Lu, and **H. Zheng**, On Maximum Elastic Scheduling in Virtual Private Networks with the Hose Model, submitted to the IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2017.

4. Y. Chen, **H. Zheng**, and J. Wu, Optimizing Software Defined Network Updates through Sparse Links, in the IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA), 2017.

5. **H. Zheng** and J. Wu, Cooperative Wireless Charging Vehicle Scheduling in Wireless Sensor Networks, in the IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS), 2017.

6. T. Mosharraf, J. Wu, and **H. Zheng**, Vehicle Routing with Pickup and Delivery: A Greedy Approach, in the ACM MobiCom Workshop on Challenged Networks (CHANTS), 2017. Poster paper.

7. **H. Zheng** and J. Wu, Friend Recommendations in OSNs: Perspective of Social Influence Maximization, in the International Conference on Computer Communications and Networks (ICCCN), 2017.

8. W. Chang, **H. Zheng**, and J. Wu, On the RSU-based Secure Distinguishability Among Vehicular Flows, in the IEEE/ACM International Symposium on Quality of

Service (IWQoS), 2017. Short paper.

9. **H. Zheng** and J. Wu, Online to Offline Business: Urban Taxi Dispatching with Passenger-Driver Matching Stability, in the IEEE International Conference on Distributed Computing Systems (ICDCS), 2017.

10. **H. Zheng** and J. Wu, Connected Placement of Disaster Shelters in Modern Cities, in the ACM MobiCom Workshop on Challenged Networks (CHANTS), 2016.

11. **H. Zheng** and J. Wu, NSFA: Nested Scale-Free Architecture for Scalable Publish/Subscribe over P2P Networks, in the IEEE International Conference on Network Protocols (ICNP), 2016.

12. **H. Zheng**, Z. Wan, and J. Wu, Optimizing MapReduce Framework through Joint Scheduling of Overlapping Phases, in the International Conference on Computer Communications and Networks (ICCCN), 2016. **Best paper runner-up**.

13. **H. Zheng**, W. Chang, and J. Wu, Coverage and Distinguishability Requirements for Traffic Flow Monitoring Systems, in the IEEE/ACM International Symposium on Quality of Service (IWQoS), 2016. **Best paper award**.

14. **H. Zheng** and J. Wu, Effective Social Network Quarantine with Minimal Isolation Costs, in the IEEE International Conference on Communications (ICC), 2016.

15. **H. Zheng** and J. Wu, DynFluid: Predicting Time-Evolving Rating in Recommendation Systems via Fluid Dynamics, in the IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2015.

16. **H. Zheng** and J. Wu, Snowballing Effects in Preferential Attachment: The Impact of The Initial Links, in the International Conference on Computer Communications and Networks (ICCCN), 2015.

17. Z. Wan, J. Wu, and **H. Zheng**, Utility-based Uploading Strategy in Cloud Scenarios, in the International Conference on Computer Communications and Networks (ICCCN), 2015.

18. **H. Zheng** and J. Wu, Optimizing Roadside Advertisement Dissemination in

Vehicular Cyber-Physical Systems, in the IEEE International Conference on Distributed Computing Systems (ICDCS), 2015.

19. **H. Zheng** and J. Wu, Data Collection and Event Detection in the Deep Sea with Delay Minimization, in the IEEE International Conference on Sensing, Communications, and Networking (SECON), 2015.

20. J. Wu and **H. Zheng**, On Efficient Data Collection and Event Detection with Delay Minimization in Deep Sea, in the ACM MobiCom Workshop on Challenged Networks (CHANTS), 2014. Short Paper.

21. R. Beigel, J. Wu, and **H. Zheng**, On Optimal Scheduling of Multiple Mobile Chargers in Wireless Sensor Networks, in the ACM MobiHoc Workshop on Mobile Sensing, Computing and Communication (MSCC), 2014. **Best paper award**.

22. **H. Zheng** and J. Wu, Spatiotemporal Cascades in Online Social Networks, in the SIAM International Conference on Data Mining (SDM), 2014. Doctoral Forum.

23. **H. Zheng** and J. Wu, Fast Information Cascade Prediction Through Spatiotemporal Decompositions, in the IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS), 2014.

24. **H. Zheng** and J. Wu, Up-and-Down Routing in Mobile Opportunistic Social Networks with Bloom-Filter-Based Hints, in the IEEE/ACM International Symposium on Quality of Service (IWQoS), 2014.

25. **H. Zheng**, Y. Wang, and J. Wu, Optimizing Multi-copy Two-hop Routing in Mobile Social Networks, in the IEEE International Conference on Sensing, Communication, and Networking (SECON), 2014.

26. W. Jiang, J. Wu, G. Wang, and **H. Zheng**, FluidRating: A Time-Evolving Rating Scheme in Trust-based Recommendation Systems Using Fluid Dynamics, in the IEEE International Conference on Computer Communications (INFOCOM), 2014.

27. K. Li, **H. Zheng**, and J. Wu, Migration-based Virtual Machine Placement in Cloud Systems, in the IEEE International Conference on Cloud Networking (CloudNet),

2013.

28. **H. Zheng** and J. Wu, Spectral Graph Multisection Through Orthogonality, in the ACM SIGKDD Workshop on Multiple Clusterings, Multi-view Data, and Multi-source Knowledge-driven Clustering (MultiClust), 2013.

29. H. Zhou, **H. Zheng**, J. Wu, and J. Chen, Energy-efficient Contact Probing in Opportunistic Mobile Networks, in the International Conference on Computer Communications and Networks (ICCCN), 2013.

30. **H. Zheng**, K. Li, C. Tan, and J. Wu, User-based CPU Verification Scheme for Public Cloud Computing, in the IEEE International Conference on Cloud Computing (CLOUD), 2013.

31. S. Hou, X. Zhang, **H. Zheng**, L. Zhao, and F. Wang, An Effective Interference Management Framework to Achieve Energy-Efficient Communications for Heterogeneous Network through Cognitive Sensing, in the International Conference on Communications and Networking in China (ChinaCom), 2012.

32. **H. Zheng**, An Improved Niche Genetic Algorithm Based On Simulated Annealing: SANGA, in the International Conference on Computational Problem-Solving (ICCP), 2011.

Journal:

33. **H. Zheng**, Y. Chen, and J. Wu, Joint Scheduling of Overlapping MapReduce Phases: Pair Jobs for Optimization, submitted to IEEE Transactions on Services Computing, 2017.

34. **H. Zheng** and J. Wu, Optimizing Colors in Online Social Networks through Partial Multi-interval Graphs, submitted to Science China Information Sciences, 2017.

35. **H. Zheng**, Y. Wang, and J. Wu, Optimizing Opportunistic Multi-copy Two-hop Routing in Mobile Social Networks, submitted to Journal of Parallel and Distributed Computing, 2017.

36. **H. Zheng**, W. Chang, and J. Wu, Traffic Flow Monitoring Systems in Smart Cities:

Coverage and Distinguishability Among Vehicles, submitted to Journal of Parallel and Distributed Computing, 2017.

37. **H. Zheng** and J. Wu, Smart City Advertisement Dissemination using Vehicular Cyber-Physical Systems, submitted to IEEE Transactions on Vehicular Technology, 2017.

38. **H. Zheng**, N. Wang, and J. Wu, Minimizing Deep Sea Data Collection Delay with Autonomous Underwater Vehicles, Journal of Parallel and Distributed Computing, 2017.

39. C. Song, J. Wu, M. Liu, and **H. Zheng**, Efficient Routing through Discretization of Overlapped Road Segments in VANETs, Journal of Parallel and Distributed Computing, 2017.

40. **H. Zheng** and J. Wu, Up-and-Down Routing through Nested Core-periphery Hierarchy in Mobile Opportunistic Social Networks, IEEE Transactions on Vehicular Technology, 2017.

41. H. Yao, H. Zhang, C. Zhang, D. Zeng, J. Wu, and **H. Zheng**, Data or Index: A Trade-off in Mobile Delay Tolerant Networks, International Journal of Computational Science and Engineering, 2017.

42. **H. Zheng** and J. Wu, Which, When, and How: Human-Machine Cooperations in Hierarchical Clusterings, MDPI Algorithms, 2016.

43. G. Wang, W. Jiang, J. Wu, F. Li, and **H. Zheng**, Trust Evaluation in Online Social Networks using Generalized Network Flow, IEEE Transactions on Computers, 2016.

44. W. Jiang, J. Wu, G. Wang, and **H. Zheng**, Forming Opinions via Trusted Friends: Time-evolving Rating Prediction Using Fluid Dynamics, IEEE Transactions on Computers, 2016.

45. **H. Zheng** and J. Wu, Effective Network Quarantine with Minimal Restrictions on Communication Activities, IEEE Transactions on Network Science and Engineering, 2016.

46. H. Zhou, J. Chen, **H. Zheng**, and J. Wu, Energy Efficiency and Contact Opportunities Trade-off in Opportunistic Mobile Networks, IEEE Transactions on Vehicular Technology, 2016.

47. W. Jiang, J. Wu, G. Wang, and **H. Zheng**, Blood Typing for Families: A Novel Hybrid Human-Computer Application, International Journal of Parallel, Emergent and Distributed Systems, 2015.

48. **H. Zheng**, K. Li, A. Blaisse, C. Tan, and J. Wu, Cloud CPU Verification Scheme for Individual End Users, International Journal of Cloud Computing and Services Science, 2014.

49. K. Li, **H. Zheng**, J. Wu, and X. Du, Virtual Machine Placement in Cloud Systems through Migration Process, International Journal of Parallel, Emergent and Distributed Systems, 2014.

50. J. Li and **H. Zheng**, Determinate Joint Remote Preparation of An Arbitrary W-class Quantum State, Chinese Physics C, 2012.

51. **H. Zheng**, Coverage Optimization Methods in Wireless Homo-Sensor Network Based on Guided Swarms, Applied Mechanics and Materials, 2011.

Book Chapter:

52. W. Chang, **H. Zheng**, J. Wu, C. Tan, and H. Ling, Environmental-Assisted Vehicular Data in Smart Cities, Smart Cities: Foundations, Principles, and Applications, 2017.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] H. Jeong, Z. Neda, and A.-L. Barabási, "Measuring preferential attachment in evolving networks," *Europhysics Letters*, vol. 61, no. 4, pp. 567–572, 2007.

[2] A. Mahanti, N. Carlsson, M. Arlitt, and C. Williamson, "A tale of the tails: Power-laws in internet measurements," *IEEE Network*, vol. 27, no. 1, pp. 59–64, 2013.

[3] H.-J. Hung et al., "When social influence meets item inference," in *ACM KDD*, 2016, pp. 1–12.

[4] E. Bulut and B. K. Szymanski, "Constructing limited scale-free topologiesover peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 4, pp. 919–928, 2014.

[5] J. Tan, B. Swapna, and N. B. Shroff, "Retransmission delays with bounded packets: Power-law body and exponential tail," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 27–38, 2014.

[6] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *IEEE Internet Computing*, 2002.

[7] M. E. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E*, vol. 64, no. 2, p. 25102, 2001.

[8] https://www.facebook.com/advertising.

[9] S. N. Dorogovtsev and J. F. Mendes, "Scaling properties of scale-free evolving networks: Continuous approach," *Physical Review E*, vol. 63, no. 5, p. 56125, 2001.

[10] http://tuvalu.santafe.edu/%7Eaaronc/powerlaws/.

[11] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[12] D. Wang, C. Song, and A.-L. Barabási, "Quantifying long-term scientific impact," *Science*, vol. 342, pp. 127–132, 2013.

[13] X. Zhao, A. Sala, C. Wilson, X. Wang, S. Gaito, H. Zheng, and B. Y. Zhao, "Multi-scale dynamics in a massive online social network," in *ACM IMC*, 2012, pp. 171–184.

[14] Y. Wu, T. Z. Fu, and D. M. Chiu, "Generalized preferential attachment considering aging," *Journal of Informetrics*, vol. 8, no. 3, pp. 650–658, 2014.

[15] M. Wang, G. Yu, and D. Yu, "Effect of the age of papers on the preferential attachment in citation networks," *Physica A*, vol. 388, no. 19, pp. 4273–4276, 2009.

[16] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *ACM IMC*, 2007, pp. 1–14.

[17] J. Liu, Y. Dang, Z. Wang, and T. Zhou, "Relationship between the in-degree and out-degree of WWW," *Physica A*, vol. 371, no. 2, pp. 861–869, 2006.

[18] M. Zanin, P. Cano, O. Celma, and J. M. Buldu, "Preferential attachment, aging and weights in recommendation systems," *International Journal of Bifurcation and Chaos*, vol. 19, no. 02, pp. 755–763, 2009.

[19] A. M. Petersen, W.-S. Jung, J.-S. Yang, and H. E. Stanley, "Quantitative and empirical demonstration of the matthew effect in a study of career longevity," *Proceedings of the National Academy of Sciences*, vol. 108, pp. 18–23, 2011.

[20] C. Watts and N. Gilbert, "Does cumulative advantage affect collective learning in science? an agent-based simulation," *Scientometrics*, vol. 89, no. 1, pp. 437–463, 2011.

[21] R. Kumar, Y. Lifshits, and A. Tomkins, "Evolution of two-sided markets," in *ACM WSDM*, 2010, pp. 311–320.

[22] D. Braha, B. Stacey, and Y. Bar-Yam, "Corporate competition: A self-organized network," *Social Networks*, vol. 33, no. 3, pp. 219–230, 2011.

[23] M. Kas, K. M. Carley, and L. R. Carley, "Trends in science networks: understanding structures and statistics of scientific networks," *Social Network Analysis and Mining*, vol. 2, no. 2, pp. 169–187, 2012.

[24] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *ACM SIGKDD*, 2005, pp. 177–187.

[25] M. Cha, A. Mislove, and K. P. Gummadi, "A measurement-driven analysis of information propagation in the Flickr social network," in *ACM WWW*, 2009, pp. 721–730.

[26] B. F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H.-A. Jacobsen, N. Puz, D. Weaver, and R. Yerneni, "PNUTS: Yahoo!'s hosted data serving platform," *VLDB Endowment*, vol. 1, no. 2, pp. 1277–1288, 2008.

[27] C. Chen, R. Vitenberg, and H.-A. Jacobsen, "A generalized algorithm for publish/subscribe overlay design and its fast implementation," in *DISC*, 2012, pp. 76–90.

[28] V. Muthusamy and H.-A. Jacobsen, "Infrastructure-free content-based publish/subscribe," *IEEE/ACM Transactions on Networking*, vol. 22, no. 5, pp. 1516–1530, 2014.

[29] M. A. Tariq, B. Koldehofe, and K. Rothermel, "Securing broker-less publish/subscribe systems using identity-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 518–528, 2014.

[30] S. Voulgaris, E. Rivière, A.-M. Kermarrec, and M. Van Steen, "Sub-2-Sub: Self-organizing content-based publish and subscribe for dynamic and large scale collaborative networks," in *ACM IPTPS*, 2006, pp. 16–35.

[31] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Computer Systems*, vol. 19, no. 3, pp. 332–383, 2001.

[32] W. W. Terpstra, S. Behnel, L. Fiege, A. Zeidler, and A. P. Buchmann, "A peer-to-peer approach to content-based publish/subscribe," in *ACM DEBS*, 2003, pp. 1–8.

[33] I. Aekaterinidis and P. Triantafillou, "PastryStrings: A comprehensive content-based publish/subscribe DHT network," in *IEEE ICDCS*, 2006, pp. 23–23.

[34] Y. Mi, X. Liao, X. Huang, L. Zhang, W. Gu, G. Hu, and S. Wu, "Long-period rhythmic synchronous firing in a scale-free network," *Proceedings of the National Academy of Sciences*, vol. 110, no. 50, pp. E4931–E4936, 2013.

[35] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, p. 2, 2007.

[36] R. Daft, *Organization theory and design*. Cengage learning, 2012.

[37] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1576–1589, 2011.

[38] J. F. Rodrigues Jr, H. Tong, J.-Y. Pan, A. J. Traina, C. Traina, and C. Faloutsos, "Large graph analysis in the gmine system," *ACM Transactions on Knowledge Discovery from Data*, vol. 25, no. 1, pp. 106–118, 2013.

[39] M. Takaffoli, O. R. Zaïane *et al.*, "Social network analysis and mining to support the assessment of on-line student participation," *ACM SIGKDD Explorations Newsletter*, vol. 13, no. 2, pp. 20–29, 2012.

[40] S. Yang, Q. Sun, S. Ji, P. Wonka, I. Davidson, and J. Ye, "Structural graphical lasso for learning mouse brain connectivity," in *ACM KDD 2015*, 2015, pp. 1385–1394.

[41] B. Corominas-Murtra, J. Goñi, R. V. Solé, and C. Rodríguez-Caso, "On the origins of hierarchy in complex networks," *Proceedings of the National Academy of Sciences*, vol. 110, no. 33, pp. 13 316–13 321, 2013.

[42] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *IEEE SocialCom*, 2011, pp. 1318–1326.

[43] F. Rahimian, S. Girdzijauskas, A. H. Payberah, and S. Haridi, "Vitis: A gossip-based hybrid overlay for internet-scale publish/subscribe enabling rendezvous routing in unstructured overlay networks," in *IEEE IPDPS*, 2011, pp. 746–757.

[44] V. Setty, M. Van Steen, R. Vitenberg, and S. Voulgaris, "Poldercast: Fast, robust, and scalable architecture for P2P topic-based pub/sub," in *ACM/IFIP/USENIX Middleware*, 2012, pp. 271–291.

[45] S. Qian, J. Cao, Y. Zhu, and M. Li, "REIN: A fast event matching approach for content-based publish/subscribe systems," in *IEEE INFOCOM*, 2014, pp. 2058–2066.

[46] A. Margara and G. Cugola, "High-performance publish-subscribe matching using parallel hardware," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 126–135, 2014.

[47] X. Xu, R. Ansari, A. Khokhar, and A. V. Vasilakos, "Hierarchical data aggregation using compressive sensing (hdacs) in wsns," *ACM Transactions on Sensor Networks*, vol. 11, no. 3, p. 45, 2015.

[48] K. Jayaram, P. Eugster, and C. Jayalath, "Parametric content-based publish/subscribe," *ACM Transactions on Computer Systems*, vol. 31, no. 2, p. 4, 2013.

[49] M. A. Tariq, B. Koldehofe, and K. Rothermel, "Efficient content-based routing with network topology inference," in *ACM DEBS*, 2013, pp. 51–62.

[50] E. Fidler, H.-A. Jacobsen, G. Li, and S. Mankovskii, "The PADRES distributed publish/subscribe system," in *International Conference on Feature Interactions in Telecommunications and Software Systems*, 2005, pp. 12–30.

[51] P. R. Pietzuch and J. M. Bacon, "Hermes: A distributed event-based middleware architecture," in *IEEE ICDCS*, 2002, pp. 611–618.

[52] A. Gupta, O. D. Sahin, D. Agrawal, and A. E. Abbadi, "Meghdoot: content-based publish/subscribe over P2P networks," in *ACM/IFIP/USENIX Middleware*, 2004, pp. 254–273.

[53] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001.

[54] http://www.similarweb.com/global.

[55] http://www.pewinternet.org/fact-sheets.

[56] https://www.facebook.com/notes/facebook/people-you-may-know/15610312130.

[57] J. Bu, S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X. He, "Music recommendation by unified hypergraph: combining social media information and music content," in *ACM Multimedia*, 2010, pp. 391–400.

[58] M. Ye, P. Yin, and W.-C. Lee, "Location recommendation for location-based social networks," in *ACM SIGSPATIAL*, 2010, pp. 458–461.

[59] D. Carnegie, *How to win friends and influence people*. Simon and Schuster, 2010.

[60] https://www.facebook.com/business/success/.

[61] https://business.twitter.com/success-stories/rukes.

[62] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *ACM SIGKDD*, 2003, pp. 137–146.

[63] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *ACM SIGKDD*, 2010, pp. 1029–1038.

[64] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *IEEE ICDM*, 2010, pp. 88–97.

[65] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *ACM WWW*, 2011, pp. 665–674.

[66] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *ACM-SIAM SODA*, 2014, pp. 946–957.

[67] X. Yang, H. Steck, and Y. Liu, "Circle-based recommendation in online social networks," in *ACM SIGKDD*, 2012, pp. 1267–1275.

[68] Y. Zhang, B. Cao, and D.-Y. Yeung, "Multi-domain collaborative filtering," in *UAI*, 2010, pp. 1–10.

[69] Y. Zheng, L. Zhang, Z. Ma, X. Xie, and W.-Y. Ma, "Recommending friends and locations based on individual location history," *ACM Transactions on the Web (TWEB)*, vol. 5, no. 1, p. 5, 2011.

[70] M. B. Zafar, P. Bhattacharya, N. Ganguly, K. P. Gummadi, and S. Ghosh, "Sampling content from online social networks: Comparing random vs. expert sampling of the twitter stream," *ACM Transactions on the Web*, vol. 9, no. 3, p. 12, 2015.

[71] M. Ye, X. Liu, and W.-C. Lee, "Exploring social influence for recommendation: a generative model approach," in *ACM SIGIR*, 2012, pp. 671–680.

[72] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *ACM SIGIR*, 2013, pp. 363–372.

[73] V. V. Vazirani, *Approximation algorithms*. Springer Science & Business Media, 2013.

[74] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[75] S. Cheng, H. Shen, J. Huang, W. Chen, and X. Cheng, "IMRank: influence maximization via finding self-consistent ranking," in *ACM SIGIR*, 2014, pp. 475–484.

[76] G. Mao and N. Zhang, "Analysis of average shortest-path length of scale-free network," *Journal of Applied Mathematics*, vol. 2013, no. 1, pp. 1–5, 2013.

[77] G. Song, X. Zhou, Y. Wang, and K. Xie, "Influence maximization on large-scale mobile social network: a divide-and-conquer method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1379–1392, 2015.

[78] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[79] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *ACM WOSN*, 2009, pp. 37–42.

[80] J. Tang, H. Gao, H. Liu, and A. Das Sarma, "etrust: Understanding trust evolution in an online world," in *ACM SIGKDD*, 2012, pp. 253–261.

[81] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *ACM WWW*, 2010, pp. 641–650.

[82] P. Chalermsook, A. Das Sarma, A. Lall, and D. Nanongkai, "Social network monetization via sponsored viral marketing," *ACM SIGMETRICS Performance Evaluation Review*, vol. 43, no. 1, pp. 259–270, 2015.

[83] B. Bahmani, K. Chakrabarti, and D. Xin, "Fast personalized pagerank on mapreduce," in *ACM SIGMOD*, 2011, pp. 973–984.

[84] H. Nguyen and R. Zheng, "On budgeted influence maximization in social networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 6, pp. 1084–1094, 2013.

[85] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM SIGMOD Record*, vol. 42, no. 2, pp. 17–28, 2013.

[86] M. Edelson, T. Sharot, R. J. Dolan, and Y. Dudai, "Following the crowd: brain substrates of long-term memory conformity," *Science*, vol. 333, no. 6038, pp. 108–111, 2011.

[87] H. Zhang, D. T. Nguyen, H. Zhang, and M. T. Thai, "Least cost influence maximization across multiple social networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 929–939, 2016.

[88] J. L. Z. Cai, M. Yan, and Y. Li, "Using crowdsourced data in location-based social networks to explore influence maximization," in *IEEE INFOCOM*, 2016, pp. 1–9.

[89] G. Tong, W. Wu, S. Tang, and D.-Z. Du, "Adaptive influence maximization in dynamic social networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 112–125, 2017.

[90] S. Galhotra, A. Arora, S. Virinchi, and S. Roy, "Asim: A scalable algorithm for influence maximization under the independent cascade model," in *ACM WWW*, 2015, pp. 35–36.

[91] W. Chen et al., "Influence maximization in social networks when negative opinions may emerge and propagate," in *SIAM SDM*, 2011, pp. 379–390.

[92] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *VLDB Endowment*, vol. 5, no. 1, pp. 73–84, 2011.

[93] M. Feldman and R. Izsak, "Constrained monotone function maximization and the supermodular degree," in *ACM-SIAM SODA*, 2014, pp. 1–10.

[94] S. Dughmi, "Algorithmic information structure design: a survey," *ACM SIGecom Exchanges*, vol. 15, no. 2, pp. 2–24, 2017.

[95] S. Fujishige and S. Isotani, "A submodular function minimization algorithm based on the minimum-norm base," *Pacific Journal of Optimization*, vol. 7, no. 1, pp. 3–17, 2011.

[96] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," in *ACM-SIAM SODA*, 2015, pp. 1–10.

[97] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *ACM SIGCOMM*, 2007, pp. 29–42.

[98] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," *Link Mining: Models, Algorithms, and Applications*, pp. 337–357, 2010.

[99] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *ACM SIGKDD*, 2010, pp. 1039–1048.

[100] T. Gradowski and A. Krawiecki, "Majority-vote model on scale-free hypergraphs," *Acta Physica Polonica A*, vol. 127, no. 3A, pp. 1–4, 2015.

[101] M. Molloy and B. Reed, "The size of the giant component of a random graph with a given degree sequence," *Combinatorics, probability and computing*, vol. 7, no. 3, pp. 295–305, 1998.

[102] U. Feige and R. Izsak, "Welfare maximization and the supermodular degree," in *ACM ITCS*, 2013, pp. 247–256.

[103] https://toreopsahl.com/datasets/#newman2001.

[104] F. D. Sahneh, C. Scoglio, and P. Van Mieghem, "Generalized epidemic mean-field model for spreading processes over multilayer complex networks," *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1609–1620, 2013.

[105] H. K. Lee, P.-S. Shim, and J. D. Noh, "Epidemic threshold of the susceptible-infected-susceptible model on complex networks," *Physical Review E*, vol. 87, no. 6, p. 062812, 2013.

[106] S. Wen, W. Zhou, J. Zhang, Y. Xiang, W. Zhou, and W. Jia, "Modeling propagation dynamics of social network worms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1633–1643, 2013.

[107] D. Chen, "Modeling the spread of infectious diseases: A review," *Analyzing and Modeling Spatial and Temporal Dynamics of Infectious Diseases*, pp. 19–42, 2014.

[108] K. Lewis, M. Gonzalez, and J. Kaufman, "Social selection and peer influence in an online social network," *Proceedings of the National Academy of Sciences*, vol. 109, no. 1, pp. 68–72, 2012.

[109] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley, "Classes of small-world networks," *Proceedings of the National Academy of Sciences*, vol. 97, no. 21, pp. 11 149–11 152, 2000.

[110] B. Proulx and J. Zhang, "Modeling social network relationships via t-cherry junction trees," in *IEEE INFOCOM*, 2014, pp. 2229–2237.

[111] X. Chen, I. Diakonikolas, D. Paparas, X. Sun, and M. Yannakakis, "The complexity of optimal multidimensional pricing," in *ACM-SIAM SODA*, 2014, pp. 1319–1328.

[112] Z. He, J. Cao, and X. Liu, "High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility," in *IEEE INFOCOM*, 2015, pp. 2542–2550.

[113] O. Gurewitz, Y. Sandomirsky, and G. Scalosub, "Cellular multi-coverage with non-uniform rates," in *IEEE INFOCOM*, 2014, pp. 1330–1338.

[114] A. Deshpande, L. Hellerstein, and D. Kletenik, "Approximation algorithms for stochastic boolean function evaluation and stochastic submodular set cover," in *ACM-SIAM SODA*, 2014, pp. 1453–1467.

[115] R. Bar-Yehuda, "Using homogeneous weights for approximating the partial cover problem," *Journal of Algorithms*, vol. 39, no. 2, pp. 137–144, 2001.

[116] S. Khuller, M. Purohit, and K. K. Sarpatwar, "Analyzing the optimal neighborhood: algorithms for budgeted and partial connected dominating set problems," in *ACM-SIAM SODA*, 2014, pp. 1702–1713.

[117] S. Funke, A. Nusser, and S. Storandt, "On k-path covers and their applications," *VLDB Endowment*, vol. 7, no. 10, pp. 893–902, 2014.

[118] J. Zhang and J. M. Moura, "Diffusion in social networks as sis epidemics: beyond full mixing and complete graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 4, pp. 537–551, 2014.

[119] L. Hébert-Dufresne and B. M. Althouse, "Complex dynamics of synergistic coinfections on realistically clustered networks," *Proceedings of the National Academy of Sciences*, vol. 112, no. 33, pp. 10 551–10 556, 2015.

[120] R. Gandhi, S. Khuller, and A. Srinivasan, "Approximation algorithms for partial covering problems," *Journal of Algorithms*, vol. 53, no. 1, pp. 55–84, 2004.

[121] S. Ewen, *Captains of consciousness: Advertising and the social roots of the consumer culture*. Basic Books, 2008.

[122] J. Tang, H. Gao, and H. Liu, "mtrust: discerning multi-faceted trust in a connected world," in *ACM WSDM*, 2012, pp. 93–102.

[123] P. Massa and P. Avesani, "Trust-aware recommender systems," in *ACM RecSys*, 2007, pp. 17–24.

[124] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[125] J. A. Golbeck, "Computing and applying trust in web-based social networks," *PhD thesis, University of Maryland*, 2005.

[126] P. Massa and P. Avesani, "Trust metrics on controversial users: Balancing between tyranny of the majority," *IJSWIS*, vol. 3, no. 1, pp. 39–64, 2007.

[127] M. Jamali and M. Ester, "Trustwalker: a random walk model for combining trust-based and item-based recommendation," in *ACM SIGKDD*, 2009, pp. 397–406.

[128] M. Sirivianos, K. Kim, and X. Yang, "Socialfilter: introducing social trust to collaborative spam mitigation," in *IEEE INFOCOM*, 2011, pp. 2300–2308.

[129] Y. L. Sun, W. Yu, Z. Han, and K. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 305–317, 2006.

[130] A. Jøsang and S. Pope, "Dempster's rule as seen by little colored balls," *Computer Intelligence*, vol. 28, no. 4, pp. 453–474, 2012.

[131] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *ACSC*, 2006, pp. 85–94.

[132] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *ISWC*, 2003, pp. 351–368.

[133] G. Mahoney, W. J. Myrvold, and G. C. Shoja, "Generic reliability trust model." in *PST*, 2005, pp. 113–120.

[134] M. Taherian, M. Amini, and R. Jalili, "Trust inference in web-based social networks using resistive networks," in *IEEE ICIW*, 2008, pp. 233–238.

[135] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz, "Trust-based recommendation systems: an axiomatic approach," in *ACM WWW*, 2008, pp. 199–208.

[136] H. Zhu, B. Huberman, and Y. Luon, "To switch or not to switch: Understanding social influence in online choices," in *ACM SIGCHI*, 2012, pp. 2257–2266.

[137] W. Jiang, J. Wu, G. Wang, and H. Zheng, "Fluidrating: A time-evolving rating scheme in trust-based recommendation systems using fluid dynamics," in *IEEE INFOCOM*, 2014, pp. 1707–1715.

[138] P. G. Hewitt, *Conceptual physics*. Pearson Educación, 2002.

[139] J. Tang, H. Gao, H. Liu, and A. D. Sarma, "eTrust: Understanding trust evolution in an online world," in *ACM SIGKDD*, 2012, pp. 253–261.

[140] G. C. Cawley and N. L. Talbot, "Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers," *Pattern Recognition*, vol. 36, no. 11, pp. 2585–2592, 2003.

[141] P. S. Chakraborty and S. Karform, "Designing trust propagation algorithms based on simple multiplicative strategy for social networks," *Procedia Technology*, vol. 6, no. 1, pp. 534–539, 2012.

[142] E. Henry and J. Hofrichter, "Singular value decomposition: application to analysis of experimental data," *Essential Numerical Computer Methods*, vol. 210, no. 1, pp. 81–138, 2010.

[143] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *ACM SIGKDD*, 2003, pp. 137–146.

[144] W. Galuba, K. Aberer, D. Chakraborty, Z. Despotovic, and W. Kellerer, "Outtweeting the twitterers - predicting information cascades in microblogs," in *ACM WOSN*, 2010, pp. 3–11.

[145] V. Gómez, H. J. Kappen, and A. Kaltenbrunner, "Modeling the structure and evolution of discussion cascades," in *ACM HT*, 2011, pp. 181–190.

[146] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," *Knowledge-Based Intelligent Information and Engineering Systems*, vol. 5179, no. 1, pp. 67–75, 2008.

[147] G. Ghasemiesfeh, R. Ebrahimi, and J. Gao, "Complex contagion and the weakness of long ties in social networks: revisited," in *ACM EC*, 2013, pp. 507–524.

[148] E. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina, "Correcting for missing data in information cascades," in *ACM WSDM*, 2011, pp. 55–64.

[149] P. Mohan, S. Shekhar, J. A. Shine, and J. P. Rogers, "Cascading spatio-temporal pattern discovery: A summary of results," DTIC Document, Tech. Rep., 2010.

[150] Y. Zhao and J. Wu, "Dache: A data aware caching for big-data applications using the mapreduce framework," *IEEE INFOCOM*, pp. 35–39, 2013.

[151] P. A. Dow, L. A. Adamic, and A. Friggeri, "The anatomy of large facebook cascades," in *AAAI ICWSM*, 2013.

[152] G. Wang, W. Jiang, J. Wu, and Z. Xiong, "Fine-grained feature-based social influence evaluation in online social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2286–2296, 2014.

[153] W. Jiang, G. Wang, and J. Wu, "Generating trusted graphs for trust evaluation in online social networks," *Future generation computer systems*, vol. 31, no. 1, pp. 48–58, 2014.

[154] J.-G. Dumas and G. Villard, "Computing the rank of large sparse matrices over finite fields," *Computer Algebra in Scientific Computing CASC, Technische Universität München*, 2002.

[155] M. Brand, "Fast low-rank modifications of the thin singular value decomposition," *Linear Algebra and its Applications*, vol. 415, no. 1, pp. 20–30, 2006.

[156] P. Shah, C. Wieser, and F. Bry, "Parallel higher-order SVD for tag-recommendations," in *ACM WWW/Internet*, 2012.

[157] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*. Philadelphia, PA, USA: SIAM, 2002.

[158] http://www.mcs.anl.gov/~itf/dbpp/text/node35.html.

[159] D. L. Streiner and J. Cairney, "What's under the ROC? an introduction to receiver operating characteristic curves," *A Guide to the Statistically Perplexed*, vol. 52, no. 4, p. 304, 2013.