AIX-MARSEILLE UNIVERSITE

ED 184 MATHEMATIQUES ET INFORMATIQUE

LABORATOIRE D'INFORMATIQUE ET SYSTEMES (LIS UMR 7020)

Thèse présentée pour obtenir le grade universitaire de Docteur

Discipline : ED 184 MATHEMATIQUES ET INFORMATIQUE

Spécialité : Automatique

Wenjing YANG

Maximisation d'Influence dans les Réseaux Sociaux

Influence Maximization in Social Networks

Soutenance prévue le 29/11/2019 devant le jury composé de :

M. Dimitri LEFEBVRE	Professeur, Université Le Havre Normandie, France	Rapporteur
Mme. Mariagazia DOTOLI	Professeur, Politecnico di Bari, Italy	Rapporteur
M. Paolo FRASCA	Chargé de Recherche CNRS, GIPSA-lab, France	Examinateur
M. Alessandro GIUA	Professeur, Università degli Studi di Cagliari, Italy	Directeur
M. Leonardo BRENNER	Maître de conférences, Aix-Marseille Université, France	Co-Directeur

To my mother, I love you

ABSTRACT

In recent years, a large number of *social network* sites (e.g., Facebook and LinkedIn) have appeared to connect people and groups together. Networks have been proven to be a good tool to share information and communicate ideas. *Influence propagation* in social networks has attracted substantial interests from the fields of complex networks, data mining and algorithmic theories.

Influence propagation occurs when an individual's opinions or behaviors change as a result of interactions with others. For example, one may adopt an information or share a video on Facebook under the influence of his acquaintances and a viral effect is finally triggered through the whole network. It is called the *"word-of-mouth effect"*. A marketing strategy that takes advantage of the effect, which is applied extensively in companies and online softwares, is called *viral marketing*. The influence maximization problem, aiming to identify a subset of initial adopters in a social network to maximize the influence propagation, is an algorithmic problem for viral marketing.

There are two progressive models most used in the analysis of social networks, namely the Independent Cascade model and the Linear Threshold model. As a type of epidemic models, the *Independent Cascade model* assumes that an individual adopts an innovation with a certain probability if at least one of its in-neighbors has adopted it. Differently, the *Linear Threshold model* assumes that an individual adopts an innovation if a certain ratio of its in-neighbors have already adopted it. We apply the Independent Cascade model in the thesis.

The thesis addresses three different problems related to influence maximization in social networks: influence propagation computation, influence maximization by seed selection and influence maximization by link activation. Firstly, the *influence propagation computation* consist in computing the probability that each node can be activated given a certain set of initial adopters. It is a preliminary step for the subsequent achievement of influence maximization. We propose the *Path Method* to give an exact result, the *SSS-Noself* algorithm and the *SSS-Bounded-Path* algorithm to give an approximate result. Secondly, the *influence maximization by seed selection* consist in maximizing the final influence propagation by targeting a seed set of certain cardinality. We use our approaches for influence propagation

computation together with different heuristics for seed selection to reach the goal of influence maximization. Thirdly, we initially propose the problem of *influence maximization by link activation*, which is to activate the most effective links within a limited budget to achieve influence maximization. Various properties of this problem and some sub-optimal solutions such as SimCD, MulCD, SimID and MulID are given.

Keywords: social networks, Independent Cascade models, influence propagation estimation, influence maximization, seed selection, link activation

RÉSUMÉ

Récemment, un grand nombre de sites de réseaux sociaux (Facebook et LinkedIn, par exemple) sont apparu pour relier des personnes et des groupes. Les réseaux sociaux sont des bons outils pour obtenir des informations et communiquer des idées. *La propagation de l'influence* dans les réseaux sociaux a attiré l'intérêt dans les domaines des réseaux complexes, de l'exploration de données et des théories algorithmiques.

La propagation de l'influence se produit lorsque les opinions ou les comportements d'un individu changent en conséquence des interactions avec les autres. Par exemple, on peut adopter une information ou partager une vidéo sur Facebook sous l'influence de ses connaissances. Un effet viral est enfin déclenché à sur tout le réseau. Cet effet est appellé "*l'effet de bouche à oreille*". Une stratégie marketing qui profite de l'effet est appelée *marketing viral*. Elle est largement appliqué dans les entreprises et en ligne logiciels. Le problème de la maximisation de l'influence vise à identifier un sous-ensemble d'adopteurs initiaux dans un réseau social afin de maximiser la propagation de l'influence. Dans le marketing viral, le calcul pour la maximisation de l'influence d'une façon efficace est un problème algorithmique encore ouvert.

Deux modèles progressifs sont principalement utilisés dans l'analyse des réseaux sociaux, à savoir le modèle à cascade indépendante et le modèle à seuil linéaire. Le modèle cascade indépendante suppose qu'un individu adopte une innovation avec une certaine probabilité si au moins un de ses voisins l'a adoptée. Autrement, le modèle à seuil linéaire suppose qu'un individu adopte une innovation si un certain ratio de ses voisins l'avons déjà adopté. Nous nous intéressons au modèle à cascade indépendante dans notre thèse.

Cette thèse aborde trois problèmes différents liés à la maximisation de l'influence dans les réseaux sociaux: l'estimation de l'influence, la maximisation de l'influence par la sélection des diffuseurs initiaux, et la maximisation de l'influence par l'activation des liens. D'abord, l'estimation de l'influence consiste à calculer la probabilité pour que chaque noeud puisse être activé par un certain ensemble de diffuseurs initiaux. C'est une étape préliminaire pour la réalisation de la maximisation de l'influence. Nous proposons une méthode appelée *méthode du chemin* pour calculer un résultat exact, en

suite l'algorithme *SSS-Noself* et l'algorithme *SSS-Bounded-Path* pour calculer un résultat approximatif. Deuxièmement, la maximisation de l'influence par la sélection des diffuseurs initiaux consiste à maximiser l'influence finale obtenue par un certain nombre des ces diffuseurs initiaux. Nous utilisons nos approches pour l'estimation d'influence avec différentes heuristiques pour la sélection des diffuseurs initiaux afin d'atteindre l'objectif de maximisation de l'influence. Troisièmement, nous proposons un problème de la maximisation de l'influence par l'activation des liens. Il consiste à activer les liens les plus efficaces avec un budget limité pour maximiser l'influence. Diverses propriétés de ce problème et certaines solutions sous-optimales telles que SimCD, MulCD, SimID et MulID sont données.

Mots-clés: réseaux sociaux, modèles à cascades indépendantes, estimation de l'influence, la maximisation de l'influence, sélection des diffuseurs initiaux, activation des liens

List of Figures

List of Tables

Contents

Chapter 1 Introduction

In recent years, a large number of social network sites (e.g., Facebook and LinkedIn) have appeared to connect individuals and groups of people together. Networks have been proved to be a good tool to obtain information and communicate ideas. They are becoming an effective marketing platform, through which it is possible to spread information or products at a large scale with a high speed. The influence propagation through social networks has attracted substantial interest from the fields of complex networks, data mining and algorithmic theories [1–3]. In the thesis, we also focus on the analysis of influence propagation, in the aspects of influence propagation computation, influence maximization by seed selection and link activation.

1.1 Social Networks and Influence Propagation

A social network can be considered as a directed graph made up of a set of social actors (such as individuals or organizations), sets of directed ties, and other social interactions between actors. This theoretical construct is useful in social sciences to study relationships between individuals, groups, organizations, or even entire societies. For instance, a large number of social network sites that have recently appeared, such as Facebook and Twitter, connect thousands of millions of people and groups together and provide them with an interactive platform to communicate and influence each other.

For the purpose of analysing the influence propagation over social networks, many mathematical models have been proposed. Among them two classic progressive models, namely the Independent Cascade model [4, 5] and the Linear Threshold model [6, 7], are widely used in mathematical sociology, economics and information science. As a type of epidemic models, the Independent Cascade model [8–11] assumes that an individual adopts an innovation with a certain probability each time one of its in-neighbors adopts it. Differently, the Linear Threshold model [11, 12] assumes that an individual adopts an

innovation only if a certain ratio of its in-neighbors have already adopted it.

The adoption of a specific behavior by an individual is highly influenced by his acquaintances, and influence occurs when an individual's opinions or behaviors change as a result of interactions with others [13–19]. We call this the "*word-of-mouth*" effect [20]. Marketing based on the word-of-mouth networks can be more cost-effective than the conventional direct marketing, because it leverages most of the promotional effort of customers in the market. Consider the following marketing example: a company designs a new APP for online users and aims to market it through a social network. Due to limited budget and resources, it will initially address only a limited number of users which it will reward to recommend the APP to their friends. And their friends would use it and recommend to their friends and so on. This strategy is called *viral marketing* [20] since it is similar to the spread of an epidemic.

The research on influence propagation will greatly benefit companies and individuals which aim to distribute their products or spread their ideas through a social network by the "word-of-mouth" effect. Social networks connect a large number of users and thus they are an effective model of interactions to analyze the diffusion of innovations.

1.2 Influence Propagation Analysis

Inspired by the theories of viral marketing, one of the cost-effective ways to promote a new product is to target some users through a social network, and first encourage them to adopt the product, e.g., by giving them a discount or offering them free samples. The goal is that the initial users can drive more users to adopt the new product in the network. To achieve that, we need to research the following problems:

- How to model the process of innovation diffusion with certain parameters;
- How to estimate the influence propagation of the initial users;
- How to choose a set of relatively optimal initial users to maximize the final influence propagation through a social network;

- How to choose the links to be activated such that the influence propagation is maximized regardless of the set of initial adopters.

1.2.1 Influence Diffusion Models

Since the 40's, various models have appeared for the analysis of innovation diffusion. For instance, the Threshold models are based on the threshold effect, i.e., an individual adopts an innovation if a certain ratio of its social contacts have adopted it. Differently, epidemic models, such as Susceptible-Infected-Susceptible (SIS) model and Susceptible-Infected-Recovered (SIR) model [21, 22], assume that individuals can be susceptible with a certain probability.

Kempe et al. [11] initially proposed two classic progressive models, i.e., the Independent Cascade model and the Linear Threshold model. It has been shown that the both models can be generalized and their generalized versions are equivalent. Subsequently many authors have successfully modeled more aspects of network parameters such as positive and negative opinions [23–27], competitions of multiple diffusions [28–33], cooperations between diffusions [34–36] and time-delay propagation [37, 38].

Our work is based on the Independent Cascade model.

1.2.2 Influence Propagation Computation

Influence propagation computation is a vital and essential process for the influence maximization. Only after estimating the final influence propagation, can one select the appropriate initial individuals or activated links according to the influence propagation value.

It has been proved that computing the exact influence propagation is #P-hard by Chen et al. [39] in both the Independent Cascade model and the Linear Threshold model. Thus numerous algorithms aim at approximate efficient computation. Monte Carlo simulation applied in many studies [11, 40–42] is basic and simple but quite time-consuming. Aggarwal et al. [43] gave a more efficient approximate algorithm called *SteadyStateSpread*, of which

the accuracy depends on the network structure. In order to improve the performance of *SteadyStateSpread*, Yang et al. [44] pointed out the scenario of *structural defect* and proposed a *SSSbyStep* algorithm. We also analyze the network structure which leads to the inaccuracy of influence propagation computation and give some improvements to *SteadyStateSpread* in Chapter 3.

1.2.3 Influence Maximization

One of the most important research directions in influence propagation is the influence maximization problem. The classic problem statement is described as targeting a subset of individuals to initially adopt an innovation such as to maximize the influence propagation through a social network. This problem was originally proposed by Domingos and Richardson [20, 45] based on Markov random fields. It was shown to be NP-hard under many stochastic diffusion models [1, 11]. For solving this problem, Kempe et al. [11, 11] first gave a greedy approximation algorithm which guarantees, under certain conditions, that the influence propagation approximates the optimal one within a factor of $(1 - 1/e)$. However, this approach requires long time to run the simulation, thus later much effort was devoted to derive more efficient algorithms [41, 42, 46–50].

In this thesis, we introduce the classic influence maximization problem as *influence maximization by seed selection*. We discuss how to use different influence propagation computation algorithms together with influence maximization heuristics to select a possibly optimal seed set.

Besides the influence maximization by seed selection, there are plenty of work related to link operation. However, most work about link control aims at solving the influence minimization problem [51–58]. A part of our work initially studies on the influence maximization by means of activating links under the Independent Cascade model. We aim to activate the most profitable links to achieve influence maximization within a limited budget.

1.3 Overview of Contributions

Based on the Independent Cascade model, the main contributions of this thesis is summarized as follows.

1. For exact influence propagation computation, we propose *Path Method* that explores all possible evolutions of a model, which can compute the exact solution to the influence propagation in small networks. We point out that, due to its complexity, this method is only viable for small networks but it is useful to test the correctness of different approaches.

2. For approximating influence propagation, we discuss the convergence problem and the multiple solutions problem of *SteadyStateSpread*. Moreover, we point out two factors leading to the gap between the result of *SteadyStateSpread* and the exact solution: the dependency relationship and existence of circuits. To partially overcome the error caused by circuits, we further propose a new *SSS-Noself* algorithm which updates the activation probability of one node assuming that it has not been activated at all before. Besides, another efficient algorithm to compute the influence propagation along paths of bounded length is proposed, namely *SSS-Bounded-Path*.

3. For influence maximization by seed selection, we compare different influence propagation computation algorithms together with *SelectTop*K, *RankedReplace* and greedy algorithm to select seed set for solving the influence maximization problem.

4. We initially propose the problem of influence maximization by link activation and prove that this problem is NP-hard. We analyze the monotonicity and submodularity of the influence propagation function and propose heuristics associated with either cost-degree coefficient or inf-degree coefficient. Emphatically, we prove that the algorithm of SimID can gain an influence propagation which is within $\frac{1}{2}(1-\frac{1}{e})$ of the optimal one under a certain constraint.

1.4 Thesis Organization

The thesis is structured as follows.

Chapter 2 describes formally the general framework of social network and two basic mathematic models, namely the Linear Threshold model and the Independent Cascade model. It is shown that both models can be generalized and their generalized versions are equivalent.

Chapter 3 analyzes different influence propagation computation approaches in an Independent Cascade model. Firstly, *Path Method* can give the exact value of influence propagation, but it is only viable for small networks. Secondly, an approximated method, called *SSS-Noself*, is obtained by modification of the existing *SteadyStateSpread* algorithm, based on fixed-point computation, to achieve a better accuracy. Thirdly, an efficient approach, also based on fixed-point computation, is proposed to compute the probability that a node is activated though a path of minimal length from the seed set. This algorithm, called *SSS-Bounded-Path* algorithm, can provide a lower-bound for the computation of influence propagation.

Chapter 4 applies the influence propagation computation methods proposed in Chapter 3 to the influence maximization problem together with *SelectTop*K algorithm, *RankedReplace* algorithm and greedy algorithm.

Chapter 5 formulates the influence maximization problem by link activation and proves that this problem is NP-hard. The properties of monotonicity and submodularity are discussed afterwards: the influence propagation function is monotone and submodular with respect to the seed set, but monotone and non-submodular with respect to the set of active links. Heuristics based on a cost-degree coefficient or an inf-degree coefficient are proposed to activate the most effective links within a limited budget to achieve influence maximization. It is proved that the algorithm of SimID can achieve an influence propagation within $\frac{1}{2}(1-\frac{1}{e})$ of the optimal solution under a certain constraint.

Chapter 6 proposes a summary of the whole thesis and describes the future work.

Chapter 2 Preliminaries

The basic notions of social networks and diffusion models is given in this chapter. Two different mathematical models for the diffusion of innovation are presented: they are the Linear Threshold model and the Independent Cascade model. It will be shown that both models can be generalized and their generalized versions are equivalent.

2.1 Social Networks

The studies on information propagation through social networks began in the middle of the 20th century [59, 60]. A social network is a graph of interactions and relationships between individuals and groups. Many mathematical models of social networks have been proposed, among which the Linear Threshold model and the Independent Cascade Model [8] are the ones that have received the largest attention.

A social network is represented by a directed graph $G = (V, E)$, in which V is a set of nodes representing individuals in the network. An edge $(i, j) \in E$ denotes that node i influences node j directly. We use the terms *individual* or *node*, and *edge* or *link* interchangeably. To describe all individuals with direct influence on node j, we denote the *in-neighbors* of node j as $N_j^{in} = \{i \in V | (i, j) \in E\}$. The *out-neighbors* of node j denoted as $N_j^{out} = \{i \in V | (j, i) \in E\}$ represent the individuals on which node j has direct influence [61, 62].

For example, in the network in Figure 2.1, it holds that:
$V = \{1, 2, 3, 4\}$; $E = \{(1, 2)\ (1, 3)\ (4, 1)\}$.
$N_1^{in} = \{4\}$, $N_1^{out} = \{2, 3\}$; $N_2^{in} = \{1\}$, $N_2^{out} = \emptyset$;
$N_3^{in} = \{1\}$, $N_3^{out} = \emptyset$; $N_4^{in} = \emptyset$, $N_4^{out} = \{1\}$.

In considering operational models for the spread of an idea or innovation, each node can be either active or inactive. As soon as a node *adopts* the innovation (i.e., it is *activated*),

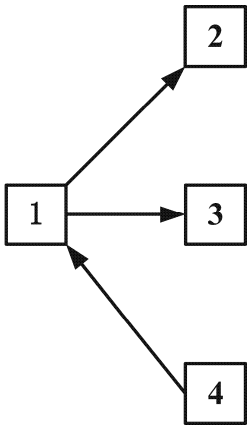

Fig. 2.1 Social network example

it becomes *active*, otherwise is said to be *inactive*. We also assume that nodes can switch from being inactive to being active, but can not switch in the other direction. It means that the adoption of an innovation is permanent and for this reason the model is called progressive. Thus, the diffusion process will look roughly as follows from the perspective of an initially inactive node j: as time unfolds, more and more of node j's in-neighbors become active; at some point, this may cause node j to become active, and node j's decision may in turn trigger further decisions by nodes to which node j is connected [63]. Note that the activation condition of nodes differs in the Linear Threshold model and the Independent Cascade model.

Generally, the information propagation is assumed to proceed in discrete time steps. The set of individuals initially selected to propagate the influence, i.e., initial adopters of the innovation at step $t = 0$, is defined as the *seed set*, represented as ϕ_0. The initial adopters are called the *seed nodes*. Then the innovation propagates from the seed set step by step. We represent the set of nodes activated at step t as ϕ_t, and we denote the set of nodes active at step t, i.e., those have been activated at or before step t as $\Phi_t = \bigcup_{s=0}^{t} \phi_s$. The activation process ends at step t_{max} when no more nodes adopt the innovation, and the set of final active nodes is denoted as $\Phi^* = \bigcup_{s=0}^{t_{max}} \phi_s$. The activation process in different diffusion models is given in Section 2.2.

2.2 Diffusion Models

Granovetter and Schelling first introduced the threshold models [6, 7] in 1978. Their approach was based on the use of node-specific thresholds. Different mathematical diffusion models have been subsequently proposed, among which threshold models and epidemic models are the most popular.

These two models characterize two different aspects of social interaction. The threshold model regards social pressure as the main drive of influence propagation, and assumes that an individual adopts a behavior if a certain ratio of its neighbors has adopted it. The epidemic model focuses on individuals and independent interaction among a social network, and assumes that an individual adopts a behavior with a certain probability if at least one of its neighbors has adopted it. Kempe et al. [11] showed that both models can be generalized and their generalized versions are equivalent. However, the basic Linear Threshold model and Independent Cascade model remain as two distinct models [39].

2.2.1 Linear Threshold Model

Image a situation where an individual needs multiple independent positive reinforcements to change his opinion or behavior. For instance, one may decide to adopt a new product after enough of his friends or acquaintances recommend it to him. Previous works have defined threshold behaviors to model such kind of diffusion [6, 7], i.e., a target individual is activated only when all the positive signals received from his in-neighbors exceed a certain threshold.

The Linear Threshold model, proposed by Kempe et al. [11], describes this type of diffusion. In the Linear Threshold model, each node j has its threshold value $\lambda_j \in [0, 1]$ and is influenced by its in-neighbors N_j^{in}. Each edge is associated with an influence weight $w_{i,j} \in [0, 1]$, denoting the importance of node i on influencing node j. The influence weights are normalized such that for each node j, the sum of influence weights of all edges from node j's in-neighbors to node j is at most 1, i.e., $\sum_{i \in N_j^{in}} w_{i,j} \leq 1$. One assumes $w_{i,j} = 0$ if $(i, j) \notin$

E. An inactive node j is activated at step t if the total influence weights of the edges from its active in-neighbors is no less than its threshold value λ_j, i.e., $\sum_{i \in N_j^{in} \cap \Phi_{t-1}} w_{i,j} \geq \lambda_j$. Thus we represent the Linear Threshold model as a triple $G_{LT} = (V, E, w)$, where $G = (V, E)$ is the social graph and $w_{i,j}$ is the influence weight associated with edge (i, j), denoting the importance of node i on activating node j.

Recall that ϕ_0 denotes the seed set, ϕ_t denotes the set of nodes activated at step t, and $\Phi_t = \bigcup_{s=0}^{t} \phi_s$ denotes the set of nodes active at step t. Given a seed set ϕ_0, at each run, initially each node $j \in V$ chooses its threshold value λ_j uniformly at random from the interval $[0, 1]$. At step $t = 1, 2, ...$, for any inactive node $j \in V \setminus \Phi_{t-1}$, if the total influence weights of the edges associated with its activated in-neighbors is no less than its threshold value λ_j, i.e., $\sum_{i \in N_j^{in} \cap \Phi_{t-1}} w_{i,j} \geq \lambda_j$, then node j is activated at step t, i.e., $j \in \phi_t$, $\Phi_t = \Phi_{t-1} \cup \phi_t$. The evolution process ends at step t_{max} when no more nodes adopt the innovation, and as talked before the set of final activated nodes is denoted as $\Phi^* = \bigcup_{s=0}^{t_{max}} \phi_s$.

Some remarks are emphasized for the Linear Threshold model as follows. First, the threshold λ_j is associated to the degree of stubbornness of node j. A large value of λ_j means that node j cannot be easily influenced by its active in-neighbors, while a small one means that node j is easily influenced by its active in-neighbors. Second, a high influence weight $w_{i,j}$ associated with an edge (i, j) means that node i has a significant influence on node j, i.e., node j can be easily influenced by node i [1].

Note that assuming the threshold value of each node is known, the model is deterministic and there exists a unique possible evolution (run) starting from a given seed set. This is called the *Deterministic Linear Threshold model*, represented by $G_{DLT} = (V, E, w, \lambda)$. As discussed with detail in above, when the threshold values are selected at runtime, then the evolution is stochastic, and the same model may have different runs starting from the same seed set. In general the default Linear Threshold model is the stochastic one without special mention.

Algorithm 1 describes the influence propagation process through the Linear Threshold model in detail.

Algorithm 1 Search final adopters Φ^* in the Linear Threshold model

Input: An Linear Threshold model $G_{IT} = (V, E, w)$, seed set $\phi_0 \subset V$;
Output: The set of final adopters Φ^*;

$t = 0$, $\Phi_t = \phi_0$;
Generate λ_j uniformly at random from the interval $[0, 1]$;
while $\phi_t \neq \emptyset$ **do**
    $t = t + 1$, $\phi_t = \emptyset$;
    for $j \in V \setminus \Phi_{t-1}$ **do**
        if $\lambda_j \geq \sum_{i \in N_j^{in} \cap \Phi_{t-1}} w_{i,j}$ **then**
            $\phi_t = \phi_t \cup \{j\}$;
        end if
    end for
    $\Phi_t = \Phi_t \cup \phi_t$;
end while
$\Phi^* = \Phi_t$.

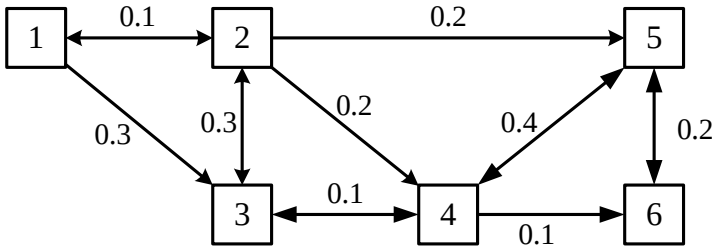

Fig. 2.2 Linear Threshold model example

Example 1. Consider the Linear Threshold model in Figure 2.2 with seed set $\phi_0 = \{1, 2\}$. Table 2.1 shows the threshold value of each node randomly selected from $[0, 1]$ in one simulation. At step $t = 1$, among the out-neighbors of seed set $\{1, 2\}$, i.e., nodes $\{3, 4, 5\}$, only node 3 can be activated, because $w_{1,3} + w_{2,3} > \lambda_3$. Similarly node 4 is activated at step $t = 2$ and node 5 is activated at step $t = 3$. However at step $t = 4$, the total influence weight of edges associated with active in-neighbors of node 6 $\{4, 5\}$ is less than its threshold value. Thus node 6 can not be activated. Since there is no node activated at step $t = 4$, then the evolution process of this model terminates. The set of final adopters is $\Phi^* = \{1, 2, 3, 4, 5\}$. The detailed evolution of is shown in Table 2.2.

TABLE 2.1 Threshold values for the network in Figure 2.2

node	1	2	3	4	5	6
λ_j	0.6	0.6	0.5	0.3	0.6	0.7

TABLE 2.2 Evolution process in Example 1

t	ϕ_k	Φ_k
0	$\{1,2\}$	$\{1,2\}$
1	$\{3\}$	$\{1,2,3\}$
2	$\{4\}$	$\{1,2,3,4\}$
3	$\{5\}$	$\{1,2,3,4,5\}$
4	$\emptyset$	$\{1,2,3,4,5\}$

2.2.2 Independent Cascade Model

Unlike the Linear Threshold model, where an individual may be influenced only if his active in-neighbors have enough "influential power", the Independent Cascade model is suitable to describe the diffusion of innovations which propagates as the spread of a virus. As in all epidemic models [8, 10, 11], it is based on the assumption that a node may adopt an innovation when one of its in-neighbors has adopted the innovation.

In an Independent Cascade model, every edge $(i,j) \in E$ is associated with a *propagation probability* $p : (V \times V) \to (0,1]$, where $p_{i,j}$ represents the probability that node j is influenced by node i through the edge (i,j) at step t when node i is activated at step $t-1$. Informally, if node i is activated at step $t-1$, it will attempt to activate each of its out-neighbors at step t. Note that for each of its out-neighbors, it only has one chance of activating it. If node i fails to activate one of its out-neighbors, it will not attempt to activate the same node at later steps. If many in-neighbors of inactive node j are activated at step $t-1$, the order in which they attempt to influence node j at step t does not affect the probability of node j being active. It is called order-independence. Finally the Independent Cascade model is denoted by a triple $G_{IC} = (V, E, p)$, where $G = (V, E)$ is the social graph and $p : (V \times V) \to (0,1]$ is the propagation probability.

Recall that ϕ_0 denotes the seed set, ϕ_t denotes the set of nodes activated at step t, and $\Phi_t = \bigcup_{s=0}^{t} \phi_s$ denotes the set of nodes active at step t. Given a seed set ϕ_0, at step t each node $i \in \phi_{t-1}$ attempts to influence its inactive out-neighbors, i.e., $j \in N_i^{out} \setminus \Phi_{t-1}$, with probability $p_{i,j}$. If node j is successfully activated at t, then it is added to ϕ_t, i.e., $\phi_t = \phi_t \cup \{j\}$. If at a certain step t_{max} there is no node activated, then the set of active nodes will no longer change, and we say the diffusion evolution terminates, i.e., $\Phi^* = \bigcup_{s=0}^{t_{max}} \phi_s$.

To simulate the stochastic evolution process in the Independent Cascade model, a random number $r_{i,j}$ uniformly distributed in the interval $[0, 1]$ is generated, and we assume that node i successfully influences node j when $r_{i,j} \leq p_{i,j}$, as shown in Figure 2.3. The detailed evolution process is described in Algorithm 2. At step t, we generate random values in $[0, 1]$ for the edges from nodes in ϕ_{t-1} to their inactive out-neighbors. If the random value $r_{i,j}$ is no more than the associated propagation probability $p_{i,j}$, then we say node i successfully influences node j.

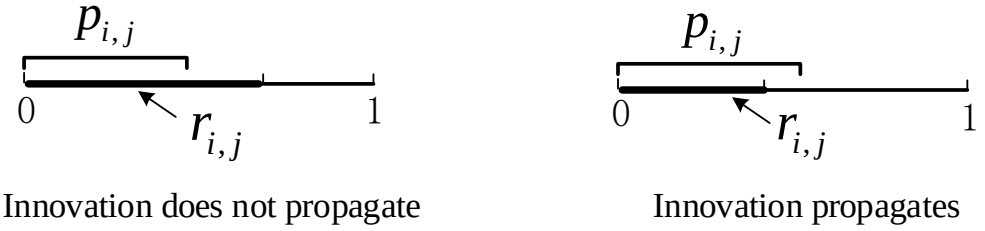

Fig. 2.3 Simulation to the innovation propagation along the edge (i, j)

Some remarks are emphasized for the Independent Cascade model as follows. First, the evolution process in this model is stochastic, and the same model may have different evolutions starting from the same seed set. Second, the propagation probability $p_{i,j}$ represents the propability that node i can influence node j. A large value of $p_{i,j}$ denotes high influence of node i to node j, i.e., node i influences node j with a high probability.

Example 2. Consider the Independent Cascade model in Figure 2.4 with seed set $\phi_0 = \{1, 2\}$. Table 2.3 shows the simulating random numbers for considered edges at each step. At step $t = 1$, from the seed nodes $\{1, 2\}$, there are four edges $\{(1, 3), (2, 3), (2, 4), (2, 5)\}$. Since we only have $r_{2,5} = 0.0922 < p_{2,5}$, then $\phi_1 = \{5\}$. Similarly we have $\phi_2 = \{6\}$. Node 6 does not have inactive out-neighbor, thus the evolution process terminates and $\Phi^* =$

Algorithm 2 Search final adopters Φ^* in the Independent Cascade model

Input: An Independent Cascade model $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$;
Output: The final adopters Φ^*;

$t = 0$, $\Phi_t = \phi_0$;
while $\phi_t \neq \emptyset$ **do**
    $t = t + 1$, $\phi_t = \emptyset$;
    for $i \in \phi_{t-1}$ **do**
        for $j \in N_i^{out} \setminus \Phi_{t-1}$ **do**
            Generate $r_{i,j}$ uniformly at random from the interval $[0, 1]$;
            if $r_{i,j} \leq p_{i,j}$ **then**
                $\phi_t = \phi_t \cup \{j\}$;
            end if
        end for
    end for
    $\Phi_t = \Phi_t \cup \phi_t$;
end while
$\Phi^* = \Phi_t$.

$\Phi_2 = \{1, 2, 5, 6\}$.

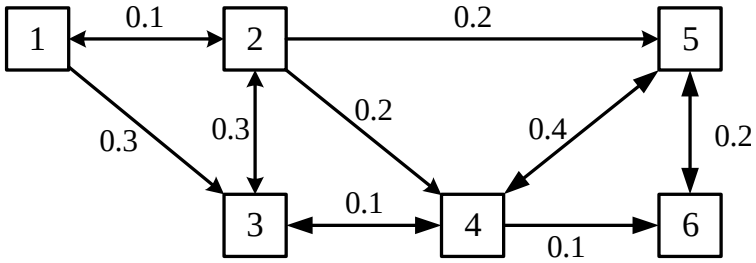

Fig. 2.4 Independent Cascade model example

TABLE 2.3 Simulation results in Example 2

t	edge	$p_{i,j}$	$r_{i,j}$	adopt	ϕ_k	Φ_k
0					$\{1, 2\}$	$\{1, 2\}$
1	$(1, 3)$	0.3	0.6787	N	$\{5\}$	$\{1, 2, 5\}$
	$(2, 3)$	0.3	0.4577	N		
	$(2, 4)$	0.2	0.3431	N		
	$(2, 5)$	0.2	0.0922	Y		
2	$(5, 6)$	0.2	0.1555	Y	$\{6\}$	$\{1, 2, 5, 6\}$

2.3 Equivalence of Models

As pointed by Chen et al. [1], the Linear Threshold model and the Independent Cascade model are not equivalent. However, they can be generalized to the General Threshold model or the General Cascade model and the generalized models are proved to be equivalent. In this section, we show by Example 3 the nonequivalence of the Linear Threshold model and the Independent Cascade model, introduce the definitions of the General Threshold model and the General Cascade model and emphasize the condition of the equivalence between the generalized models. As mentioned before, the Linear Threshold model we talked is associated with stochastic evolutions, not the deterministic model.

As for pointing out the in-equivalence of the Linear Threshold model and the Independent Cascade model, an example is given as follows.

Example 3. The directed network $G = (V, E)$ in Figure 2.5 where $V = \{1, 2, 3\}$ and $E = \{(1, 3), (2, 3)\}$ can be corresponded to a Linear Threshold model or to an Independent Cascade model. We assume there are $w_{1,3}$ and $w_{2,3}$ in the Linear Threshold model and there are $p_{1,3}$ and $p_{2,3}$ in the Independent Cascade model. When seed set $\phi_0 = \{1\}$, for a Linear Threshold model, node 3 is activated when its threshold θ_3 is smaller than $w_{1,3}$. Since θ_3 is a random variable uniformly distributed in $[0, 1]$, the activation occurs with probability $w_{1,3}$. Thus if these two models are equivalent, we must have $w_{1,3} = p_{1,3}$ when seed set $\phi_0 = \{1\}$. Similar conclusions can be taken when the seed set is $\{2\}$ or $\{1, 2\}$: we must have $w_{2,3} = p_{2,3}$ when seed set $\phi_0 = \{2\}$. When seed set $\phi_0 = \{1, 2\}$, node 3 is activated with probability $w_{1,3} + w_{2,3}$ in the Linear Threshold model, and with probability $p_{1,3} + (1 - p_{1,3}) \cdot p_{2,3}$ in the Independent Cascade model. On the condition that $w_{1,3} = p_{1,3}$ and $w_{2,3} = p_{2,3}$, $w_{1,3} + w_{2,3}$ can not equal to $p_{1,3} + (1 - p_{1,3}) \cdot p_{2,3}$ unless $w_{1,3} = p_{1,3} = 0$ or $w_{2,3} = p_{2,3} = 0$. Thus these two models can not be equivalent.

Kempe et al. [11] initially defined the generalized threshold model and the generalized cascade model. We recall the definitions as follows.

Definition 1 (General Threshold model)**.** *In the General Threshold model, every node $j \in V$ has a threshold function $f_j : 2^{N_j^{in}} \to [0, 1]$. f_j is monotone and $f_j(\emptyset) = 0$. Given a seed set*

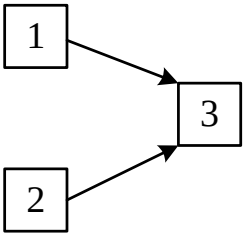

Fig. 2.5 A directed network example

ϕ_0 then the evolution proceeds as follows: every node $j \in V$ chooses a threshold value λ_j uniformly at random from the interval $[0, 1]$. At steps $t \geq 1$, we say node j is activated at step t if the function value f_j based on all active in-neighbors of node j is no less than its chosen threshold value λ_j, i.e., $f_j(N_j^{in} \cap \Phi_{t-1}) \geq \lambda_j$.

Intuitively, the Linear Threshold model is a special case of the General Threshold model where $f_j(\Phi) = \sum_{i \in \Phi} w_{i,j}$ ($w_{i,j}$ is the influence weight in the Linear Threshold model).

Definition 2 (General Cascade model)**.** *In the General Cascade model, every node $j \in V$ has an activation function $p_j : N_j^{in} \times 2^{N_j^{in}} \rightarrow [0, 1]$. $p_j(i, \Phi) \in [0, 1]$, where $\Phi \subset N_j^{in}$ and $i \in N_j^{in} \setminus \Phi$. Given a seed set ϕ_0, then when node i attempts to activate node j, it succeeds with probability $p_j(i, \Phi)$, where Φ is the subset of node j's in-neighbors that have failed to activate node j. The activation function $p_j(i, \Phi)$ is set to be order-independent, i.e., the probability that node j is activated is not affected by the order in which its in-neighbors attempt to activate it.*

Intuitively, the Independent Cascade model is a special case of the General Cascade model where $p_j(i, \Phi)$ equals to a constant $p_{i,j}$, independent of Φ.

Chen et al. [1] pointed that for any general threshold model with threshold function $f_j(\Phi)$ for every node $j \in V$, there is a corresponding general cascade model with a suitable activation function $p_j(i, \Phi)$ for every node $j \in V$, such that the two general models are equivalent and vice versa.

Suppose that there is a general cascade model with activation function $p_j(i, \Phi)$ for every node $j \in V$. Let $\Phi = \{i_1, i_2, ..., i_s\}$ be a set of in-neighbors of node j. Let $S_i = \{i_1, i_2, ..., i_i\}$

and $S_0 = \emptyset$. Given that nodes in Φ are already active, the probability that one of nodes in Φ successfully activates j is $1 - \prod_{i=1}^{s}(1 - p(i_i, S_{i-1}))$. To have this probability, the threshold function $f_j(\Phi)$ is set as

$$f_j(\Phi) = 1 - \prod_{i=1}^{s}(1 - p(i_i, S_{i-1})).$$

Conversely, suppose that there is a general threshold model with threshold function $f_j(\Phi)$ for every node $j \in V$. Given a set of in-neighbors of node j Φ and one of its in-neighbors $i \in N_j^{in} \setminus \Phi$, then $p_j(i, \Phi)$ is the conditional probability that node i activates node j on the condition that none of nodes in the set Φ activates node j. In the general threshold model, the probability that none of nodes in the set Φ activates node j is $1 - f_j(\Phi)$. Then we have node i activates node j if and only if threshold λ_j is between $f_j(\Phi)$ and $f_j(\Phi \cup \{i\})$, i.e., $p_j(i, \Phi)$ is set as

$$p_j(i, \Phi) = \frac{f_j(\Phi \cup \{i\}) - f_j(\Phi)}{1 - f_j(\Phi)}.$$

2.4 Conclusion

In this chapter, two mathematical models describing the influence propagation through social networks, namely the Linear Threshold model and the Independent Cascade model, are discussed. The Linear Threshold model assumes that an individual adopts an innovation if a certain ratio of its in-neighbors have already adopted it. While the Independent Cascade model assumes that an individual adopts an innovation with a certain probability if at least one of its in-neighbors has adopted it. Moreover, these two models can be generalized to be equivalent under a certain set of parameters.

Chapter 3 Influence Propagation Computation

3.1 Introduction

In order to maximize the final influence propagation by determining an optimal set of initial users, a preliminary step is to compute the set of final adopters for any given set of initial users. When the Independent Cascade model is considered, the measure of the influence propagation is given by the total *activation probabilities* of individuals in a network, i.e., the sum of probabilities that the individuals adopt the innovation. It has been proved that computing the exact influence propagation for the Independent Cascade model is #P-hard by Chen et al. [39]. *Monte Carlo simulation* applied in many studies [11, 40–42] is a basic and simple tool but quite time-consuming. Aggarwal et al. [43] gave a more efficient approximate algorithm called *SteadyStateSpread*. However, the computed solution may be far from the exact one, depending on the network structure, and there are no guaranteed bounds. Besides, they did not discuss the convergence of their iterative equation and the uniqueness of the final solution.

In this chapter, focusing on the Independent Cascade model, we analyse different approaches for computing the influence propagation. The main contributions [64] of this chapter can be summarised as follows:

1. To compute the exact value of the influence propagation in small networks, we propose a method that explores all possible evolutions of a model: we call this approach *Path Method*. We point out that, due to its complexity, this method is only viable for small networks but it is useful to test the correctness of different approaches.

2. We discuss the convergence problem [65] and the multiple solutions problem [66] of *SteadyStateSpread*, proving that it converges to a unique solution using fixed-point theory [67]. Moreover, we point out two factors leading to the gap between the result of *SteadyStateSpread* and the exact solution: the dependency relation among

individuals and the existence of circuits in network structures. To partially overcome the error caused by circuits, we further propose a new *SSS-Noself* algorithm which updates the activation probability of one node assuming that it has not been activated at all before.

3. We propose an efficient way to compute the influence propagation along paths of bounded length and provide a lower-bound for the influence propagation by *SSS-Bounded-Path*.

3.2 Related Work

Kempe et al. [11, 40] firstly ran Monte Carlo simulation for 10,000 times to evaluate the influence propagation. However, it is not computational efficient, then some algorithms are generated to improve the efficiency of influence propagation evaluation. Chen et al. [39] constructed the Maximum Influence Arborescence (MIA) model to restrict influence in a local region. They firstly computed Maximum Influence Paths (MIP) via Dijkstra algorithm and ignored MIPs with probability smaller than an influence threshold θ. Then they unioned the MIPs into the arborescence structures and only influence propagated through these local arborescences is considered. Aggarwal et al. [43] gave an efficient approximate algorithm called SteadyStateSpread. However, it does not always perform well specifically in the dependent sub-structures of a network. Besides, as pointed out in [44], Aggarwal et al. did not demonstrate the convergence of the iterative equation and the uniqueness of the final solution.

3.3 Influence Propagation Computation in the Independent Cascade Model

3.3.1 Problem Formulation

The evaluation of influence propagation given a seed set is the first task for the goal of influence maximization. In the thesis, the influence propagation is denoted as the sum

of activation probabilities of all nodes in a network. We give a mathematical description of the problem of influence propagation computation in the Independent Cascade model as follows.

Definition 3 (Activation Probability)**.** *Given an Independent Cascade model* $G_{IC} = (V, E, p)$ *and a seed set* ϕ_0*, the probability that a node* $j \in V$ *is activated during the innovation propagation process is defined as the activation probability of node* j*, denoted as* π_j*.*

The final influence propagation $\sigma(\phi_0)$ is given by the total activation probabilities of nodes in a network, i.e.,

$$\sigma(\phi_0) = \sum_{j \in V} \pi_j.$$

3.3.2 Hardness of Influence Propagation Computation

Chen et al. [1, 39] proved that computing influence propagation based on the Independent Cascade model is #P-hard. The class of #P problems are counting problems associated with decision problem in NP: a problem in NP needs to answer if a problem instance has a solution while a #P problem needs to provide the number of solutions to the problem instance. A problem is #P-complete if it is in class #P and every problem in #P can be reduced to it by a polynomial time reduction. A computation problem is #P-hard if it can be reduced in polynomial time from a #P-complete problem.

Theorem 1. *[39] Computing the influence propagation* $\sigma(\Phi_0)$ *in the Independent Cascade model given a seed set* ϕ_0 *is #P-hard.*

A detailed proof can be found in the paper of Chen et al. [39].

3.3.3 Submodularity and Monotonicity of Influence Propagation Function

The influence propagation function $\sigma(\cdot)$ holds two important properties in the Independent Cascade model: monotonicity and submodularity. These two properties contribute a lot

for the analysis of the greedy algorithm designed for influence maximization problem, which will be discussed in Chapter 4. We recall the definitions of monotonicity and submodularity [68] in Definition 4 and Definition 5.

Definition 4 (Monotonicity)**.** *A set function $f : 2^V \rightarrow \mathbb{R}$ is monotone if for any subset $S \subseteq T \subseteq V$, $f(S) \leq f(T)$.*

Definition 5 (Submodularity)**.** *A set function $f : 2^V \rightarrow \mathbb{R}$ is submodular if for any subset $S \subseteq T \subseteq V$ and any element $v \in V \setminus T$, the marginal gain of adding element v to a set T is no more than the marginal gain of adding element v to a subset $S \subseteq T$, i.e.,*

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T).$$

Kempe et al. [63] pointed that the influence propagation function $\sigma(\cdot)$ is monotone and submodular in the Independent Cascade model. The detailed proof can be found in [63].

Theorem 2. *The influence propagation function $\sigma(\cdot)$ in the Independent Cascade model is monotone and submodular.*

3.4 Methodology

As mentioned above, the preliminary step to determine a seed set for achieving influence maximization is to evaluate the influence propagation, i.e., the activation probabilities of all nodes in a network. The methodology for both exact and approximate influence propagation computation is given in this section.

3.4.1 Exact Influence Propagation Computation

In this part, we propose an algorithm to compute the exact solution to the influence propagation called *Path Method*. The value of activation probability computed by *Path Method* for node $j \in V$ is defined as π_j^p. Since *Path Method* can give an exact value, we have $\pi_j^p = \pi_j$.

Path Method takes into account all evolutions of a model, so that it can offer a precise result of the final influence propagation. Firstly, it creates an evolution graph for a model, which is composed of cells shown in Figure 3.1. Each cell C_k consists of three elements: *past active nodes* is a set A_k^p which contains all the nodes activated before the current step; *current active nodes* is a set A_k^c which contains the nodes activated at the current step; *cell probability* P_k is the probability that the evolution described by A_k^p and A_k^c occurs. A cell whose current active nodes is null is called a *terminal cell*. Every non-terminal cell $C_i = (A_i^p, A_i^c, P_i)$ is connected with each of its successor cells $C_k = (A_k^p, A_k^c, P_k)$ by a directed arc with which is associated the *arc probability* $P_a^{(i,k)}$. It denotes the probability that an evolution reaches cell C_i proceed to reach C_k. Adding together all the cell probabilities of terminal cells whose sets of past active nodes contain node j, the exact activation probability of node j can be computed.

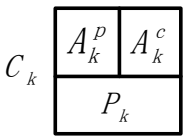

Fig. 3.1 Cell of evolution graph

Algorithm 3 creates the evolution graph of an Independent Cascade model and computes the influence propagation of a network. The symbols used in Algorithm 3 are described in Table 3.1. The algorithm defines as initial cell $C_1 = (\emptyset, \phi_0, 1)$ because initially no node has been previously explored ($A_1^p = \emptyset$), the set of currently active nodes is the seed set ($A_1^c = \phi_0$) and the probability of reaching this condition during a run is 1 ($P_1 = 1$). C_1 is also added to the set new containing cells that need to be explored.

Then, while the set new is not empty a cell $i \in new$ is selected and its child cells are computed as follows. From a cell C_i the innovation can propagate to any subset of

$$D = N_{A_i^c}^{out} - (A_i^p \cup A_i^c),$$

which contains the out-neighbors of A_i^c that have not yet adopted the innovation.

Two cells are called equivalent if both their sets of past active nodes and sets of current

Algorithm 3 *Path Method*

Input: An independent cascade network $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$;
Output: Activation probability π_j^p for all nodes $j \in V$;

/* construct the evolution graph */
Let $A_1^p = \emptyset$, $A_1^c = \phi_0$, $P_1 = 1$;
Add cell $C_1 = (A_1^p, A_1^c, P_1)$ to the graph;
$new = \{C_1\}$, $k = 1$;
while $new \neq \emptyset$ **do**
    Pick $C_i = (A_i^p, A_i^c, P_i) \in new$;
    Let $new = new \setminus \{C_i\}$;
    $D = N_{A_i^c}^{out} - (A_i^p \cup A_i^c)$;
    for all $D' \subseteq D$ **do**
        $k = k + 1$;
        $D_k = D'$, $A_k^c = D_k$;
        $A_k^p = A_i^p \cup A_i^c$;
        $$P_a^{(i,k)} = \prod_{\substack{q \in A_i^c \\ r \in D - A_k^c}} (1 - p(q,r)) \cdot \left(1 - \prod_{\substack{q' \in A_i^c \cap N_{r'}^{in} \\ r' \in A_k^c}} (1 - p(q', r'))\right);$$
        if $\exists C_{k'}$, $s.t.$ $A_{k'}^c = A_k^c$ and $A_{k'}^p = A_k^p$ **then**
            $P_a^{(i,k')} = P_a^{(i,k)}$;
            Add an arc $C_i \to C_{k'}$ with probability $P_a^{(i,k')}$;
            $P_{k'} = P_{k'} + P_i \cdot P_a^{(i,k)}$;
            $k = k - 1$;
        else
            $P_k = P_i \cdot P_a^{(i,k)}$;
            Add cell $C_k = (A_k^p, A_k^c, P_k)$ to the graph;
            Add an arc $C_i \to C_k$ with probability $P_a^{(i,k)}$;
            if $A_k^c \neq \emptyset$ **then**
                $new = new \cup \{C_k\}$;
            end if
        end if
    end for
end while
$n = k$;
/* activation probability computation */
for $j \in V$ **do**
    $\pi_j^p = 0$;
    for $k = 1$ to n **do**
        if $A_k^c = \emptyset$ and $j \in A_k^p$ **then**
            $\pi_j^p = \pi_j^p + P_k$;
        end if
    end for
end for
return π_j^p for $j \in V$.

TABLE 3.1 Symbols used in Algorithm 3.

Symbol	Description
C_k	cell k
A_k^p	past active nodes of cell k
A_k^c	current active nodes of cell k
P_k	cell probability of cell k
$P_a^{(i,k)}$	arc probability of the arc $C_i \to C_k$
k	label of cell
i	label of no-terminal cell
n	number of cells in the evolution graph
new	set of cells to be explored
D	inactive out-neighbors of the current active nodes of a cell
D'	one subset of set D
$N_{A_k^c}^{out}$	union of all out-neighbors of nodes in A_k^c

active nodes are the same. For any subset $D_k \subseteq D$, a new cell $C_k = (A_k^p, A_k^c, P_k)$ is created when it is not equivalent with any other existing cell, with $A_k^p = A_i^p \cup A_i^c$ and $A_k^c = D_k$. The probability of reaching cell C_k from C_i is

$$P_a^{(i,k)} = \prod_{\substack{q \in A_i^c \\ r \in D - A_k^c}} (1 - p(q,r)) \cdot (1 - \prod_{\substack{q' \in A_i^c \cap N_{r'}^{in} \\ r' \in A_k^c}} (1 - p(q',r'))).$$

Hence we have

$$P_k = P_i \cdot P_a^{(i,k)}.$$

If the new cell C_k has active nodes ($A_k^c \neq \emptyset$) then C_k is added to the set new, else it is a terminal cell and will not be explored further. If the cell C_k is equivalent to another cell C_k' already in the graph, we just add an arc from C_i to $C_{k'}$ with $P_a^{(i,k')} = P_a^{(i,k)}$ and increase the value of $P_{k'}$ by the amount $P_i \cdot P_a^{(i,k')}$. Finally after cell C_i has been explored it is removed from set new.

After constructing the evolution graph, we can search for the terminal cells whose sets of past active node contain node $j \in V$. Adding all the cell probabilities of these terminal cells, then π_j^p can be computed.

For example, the evolution graph for the model in Figure 3.2 with seed set $\phi_0 = \{5\}$ is shown in Figure 3.3. We briefly explain how to construct the evolution graph of this model. In C_1, only seed node 5 is activated, thus $A_1^p = \emptyset$ and $A_1^c = \{5\}$ with $P_1 = 1$. Since

the out-neighbor set of node 5 only contains node 3, only C_2 and C_3 can be obtained from C_1: C_2 corresponds to an evolution that does not activate node 3, while C_3 corresponds to an evolution that activates node 3. C_2 is a terminal cell since $A_2^c = \emptyset$. Moreover, $P_a^{(1,3)} = p_{5,3} = 0.4$ and $P_1 = 1$, thus $P_3 = P_1 * P_a^{(1,3)} = 0.4$. Four cells can be reached from C_3 according to which subset of out-neighbors of node 3 will be activated. Based on this procedure, the evolution graph corresponding to the network can be obtained. The value of each terminal cell C_k represents the probability of observing a run whose set of finally active nodes is A_k^p. For example, C_{13} corresponds to the evolution where nodes $\{1, 2, 3, 5\}$ are influenced by the order $5 \to 3 \to \{1, 2\}$ or $5 \to 3 \to 1 \to 2$, and no other node is activated. Thus the terminal cells which contain node j as a past active node describe all final evolutions in which node j can be activated. The sum of the cell probabilities of these terminal cells is the activation probability of node j. For the network in Figure 3.2, the activation probabilities obtained from the evolution graph in Figure 3.3 are shown in the second row of Table 3.2 on page 37.

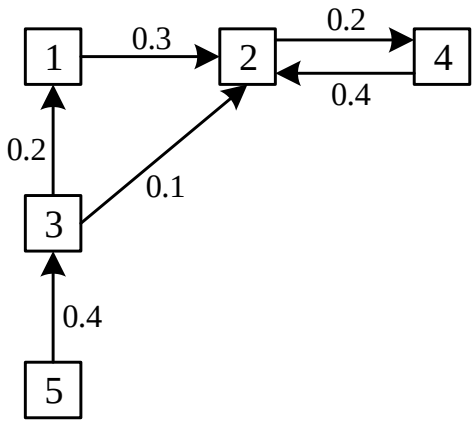

Fig. 3.2 An Independent Cascade model with 5 nodes

As for the time complexity of the *Path Method*, we have Proposition 1. Due to its exponential complexity, this method is only viable for small networks.

Proposition 1. The time complexity of the *Path Method* is $O(6^N)$.

Proof. There are three possible states for each node $j \in V$ in a cell of the evolution graph: belonging to the set of past active nodes, belonging to the set of current active nodes or in neither of these two sets. Thus the maximal number of cells in an evolution graph is 3^N.

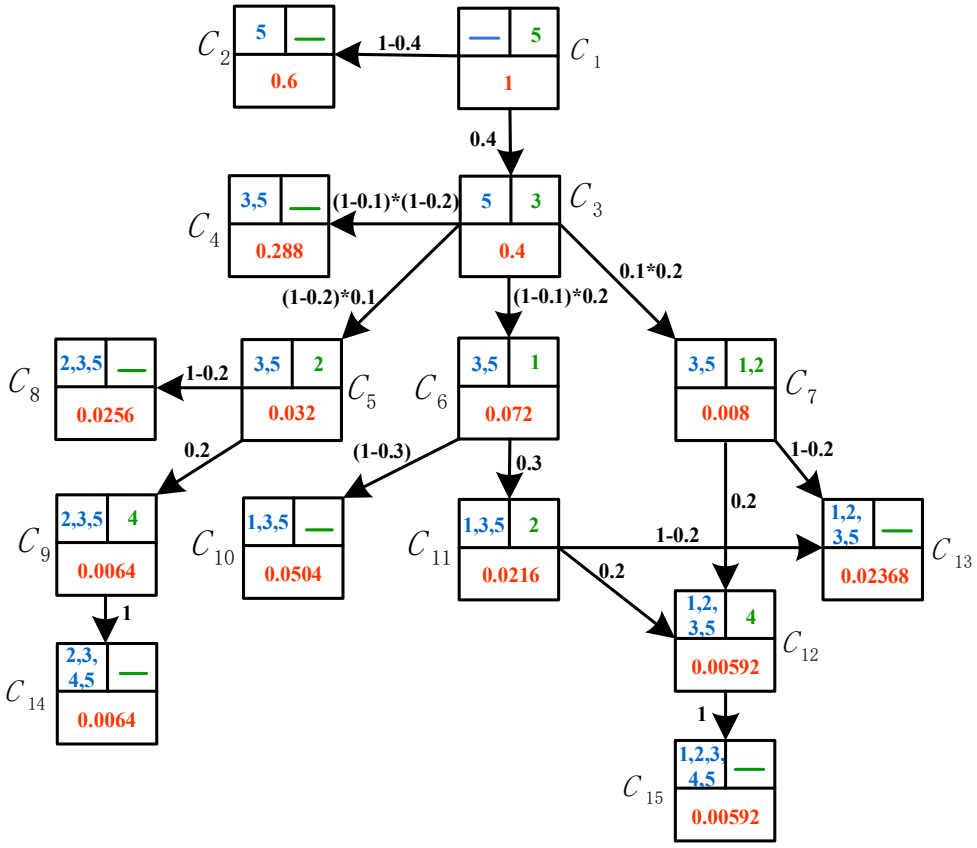

Fig. 3.3 Evolution graph of the network in Figure 3.2

The number of subsets of D is $2^{|D|}$. Since $D \subset V$ it results that the number of subsets of D is bounded by 2^N. Thus the time complexity of the part that constructs the evolution graph is $O(6^N)$. We need one pass of $j \in V$ for each cell to obtain π_j^p. Thus the time complexity of the part that computes the influence propagation is $O(N \cdot 3^N)$. The total complexity of *Path Method* is $O(6^N)$. □

3.4.2 Approximate Influence Propagation Computation by Fixed-Point Approaches

Aggarwal et al. [43] proposed the *SteadyStateSpread* algorithm to evaluate the influence propagation of a network. This iterative method computes an approximated value of the node activation probability by solving a non-linear system of equations. Though Aggarwal et al. [43] also gave an iterative algorithm, as mentioned in a subsequent work [44], they did not prove that their iterative method can converge to only one final result, and ignored

the inaccuracy caused by some specific sub-structures of networks. In order to improve the accuracy of the classic *SteadyStateSpread* algorithm, we propose the *SSS-Noself* approach which takes consideration of some specific sub-structures. Moreover, an efficient approach, also based on fixed-point computation, is proposed to compute the probability that a node is activated though a path of minimal length from the seed set. This algorithm, called *SSS-Bounded-Path* algorithm, can provide a lower-bound for the influence propagation computation.

3.4.2.1 Basic Fixed-Point Approach

In this part, we discuss the *SteadyStateSpread* algorithm based on fixed-point theory.

Definition 6 ([67])**.** *Given a real function of a real variable $f : \mathbb{R} \to \mathbb{R}$, a real number x is a fixed point of f if it satisfies*

$$x = f(x)$$

Given a point x_0 in the domain of f, the fixed-point iteration is

$$x(t+1) = f(x(t)),\ t = 0, 1, 2, \ldots$$

which generates the sequence $x(0)$, $x(1)$, $x(2)$, ... If the sequence converges to a point x and f is continuous, then one can prove that x is a fixed point of f.

We define the approximate value of activation probability computed by *SteadyState-Spread* algorithm for node $j \in V$ as π_j^s. To apply this theory for iterative influence propagation computation, it is necessary to construct a function for computing π_j^s. In the Independent Cascade model, node j can be activated by any of its in-neighbors. Equivalently, in order for node j to not be activated, it must not be activated by any of its in-neighbors. Assuming that the activation of the in-neighbors are independent events and that they do not depend on the activation of node j, the probability of that can be written

as $\prod_{i \in N_j^{in}} (1 - \pi_i^s \cdot p_{i,j})$. Thus the computation function can be constructed as following:

$$\pi_j^s(t+1) = \begin{cases} 1 & \text{if } j \in \phi_0 \\ 1 - \prod\limits_{i \in N_j^{in}} (1 - \pi_i^s(t) \cdot p_{i,j}) & \text{if } j \notin \phi_0 \end{cases} \tag{3-1}$$

The procedure of the *SteadyStateSpread* approach [43] is described in Algorithm 4.

Algorithm 4 *SteadyStateSpread*

Input: An independent cascade network $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$, stopping criterion $\varepsilon^* > 0$;

Output: Activation probability π_j^s for all nodes $j \in V$;

1: $t = 0$;
2: $\varepsilon = \varepsilon^* + 1$;
3: $\pi_j^s(0) = 1, \forall j \in \phi_0$;
4: $\pi_j^s(0) = 0, \forall j \in V \setminus \phi_0$;
5: **while** $\varepsilon \geq \varepsilon^*$ **do**
6: **for** $j \in V$ **do**
7: **if** $j \in \phi_0$ **then**
8: $\pi_j^s(t+1) = 1$;
9: **else**
10: $\pi_j^s(t+1) = 1 - \prod_{i \in N_j^{in}} (1 - p_{i,j} \cdot \pi_i^s(t))$;
11: **end if**
12: **end for**
13: $\varepsilon = \sum_{j \notin \phi_0} |\pi_j^s(t+1) - \pi_j^s(t)|$;
14: $t = t + 1$;
15: **end while**
16: **return** $\pi_j^s = \pi_j^s(t-1)$.

To prove that Equation 3-1 converges, we recall a classic monotone convergence theorem.

Theorem 3 ([69])**.** *If a sequence of real numbers is increasing and bounded above, then its supremum is the limit.*

From Theorem 3 next result follows.

Proposition 2. The sequence $\{\pi_j^s(t)\}$ generated by Equation 3-1 converges to a unique fixed-point.

Proof. Equation 3-1 generates a sequence for each node j, $\pi_j^s(0), \pi_j^s(1), \pi_j^s(2), ...$ in which,

$$\pi_j^s(0) = \begin{cases} 1 & \text{if } j \in \phi_0 \\ 0 & \text{if } j \notin \phi_0 \end{cases} \tag{3-2}$$

Such a sequence is non decreasing since there are more and more in-neighbors of node j that can propagate the innovation as the iteration proceeds. Also this sequence is upper bounded by 1. As the iteration unfolds, the sequence converges to the supremum.

Hence according to the monotone convergence theorem [69], the sequence $\{\pi_j^s(s)\}$ converges to the supremum, i.e., a fixed-point of Equation 3-1. □

In practice, the convergence to the fixed-point is asymptotic. However, we stop the iteration when the absolute difference between the computing results of adjacent iterations drops below a given stopping criterion $\varepsilon^* > 0$, i.e., $\sum_{j \notin \phi_0} |\pi_j^s(t+1) - \pi_j^s(t)| \leq \varepsilon^*$.

As an example, applying *SteadyStateSpread* to the network in Figure 3.2, we can compute the influence propagation shown in the third row of Table 3.2 on page 37.

3.4.2.2 Inaccuracy of the Basic Fixed-Point Approach

SteadyStateSpread is generally not correct because Equation 3-1 holds only assuming that the activation events of node j's in-neighbors are independent events and that these events do not depend on the activation of node j itself. Thus it can not provide exact solution for every structure of networks, especially graphs containing bidirectional edges or dependent sub-structures.

Yang et al. [44] also discussed the inaccuracy problem caused by the bidirectional edges. They gave the Example 4 to explain the scenario called *structural defect*, which corresponds to the situation: nodes $i, j \notin \phi_0$ and every path from ϕ_0 to j has to pass i, nevertheless according to a certain computation algorithm, π_i depends on π_j.

Example 4. Consider the bidirectional network in Figure 3.4, and assume the seed set $\phi_0 = \{1\}$. According to Equation 3-1, the activation probability of node 4 depends on the activation probabilities of node 2, node 3 and node 5. However, node 5 can be activated only

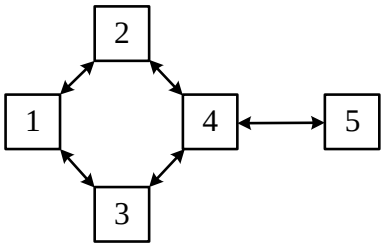

Fig. 3.4 An bidirectional network example

after node 4 having been activated. Node 4 is independent from node 5 and this contradicts with Equation 3-1.

They defined the network structure as shown in Figure 3.4 as structure defect.

Definition 7 ([44]). *Given a network $G = (V, E)$, a seed set ϕ_0 and an influence propagation computation algorithm A, if $i, j \notin \phi_0$ and every path from ϕ_0 to j has to pass i, and according to algorithm A, the value of π_i depends on π_j, then we say that $(i, j)$is a structural defect of algorithm A on G.*

Besides the inaccuracy caused by bidirectional edges discussed above, we illuminate that the dependent sub-structures can lead to the inaccuracy. Comparing the results for the network in Figure 3.2 computed by *Path Method* and *SteadyStateSpread* shown in Table 3.2, we find $\pi_2^s > \pi_2^p$ and $\pi_4^s > \pi_4^p$ (shown in bold in Table 3.2 on page 37). As mentioned by Yang et al. [44], one reason is: in Equation 3-1, π_2^s depends on π_4^s, i.e., node 4 increases π_2^s. However, node 4 can be activated only after node 2's activation. Another reason is that the dependency relation between node 2's in-neighbors (node 1 and node 3) increases the final result of node 2. Ignoring the influence of node 4, the equation to compute the activation probability of node 2 by *SteadyStateSpread* is

$$\begin{aligned} \pi_2^s &= 1 - (1 - \pi_3^s p_{3,2}) \cdot (1 - \pi_1^s p_{1,2}) \\ &= \pi_3^s p_{3,2} + \pi_1^s p_{1,2} - \pi_1^s \pi_3^s p_{3,2} p_{1,2} \end{aligned} \tag{3-3}$$

Nevertheless, the exact equation to compute the activation probability of node 2 should be

$$\begin{aligned} \pi_2 &= \pi_3 \cdot (p_{3,2} + (1 - p_{3,2}) \cdot p_{3,1} p_{1,2}) \\ &= \pi_3 p_{3,2} + \pi_1 p_{1,2} - \pi_3 p_{3,2} p_{1,2} p_{3,1} \end{aligned} \tag{3-4}$$

Moreover, circuits among more than two nodes also contribute to the error in the solution computed by *SteadyStateSpread*.

Overall, circuits and dependency relations among nodes lead to computing influence propagation by *SteadyStateSpread* that are equal to or greater than the exact solution.

3.4.2.3 Improving the Fixed-Point Approach

To limit the bad impact of bidirectional edges, Yang et al. [44] incorporated the Maximum Influence Path (MIP) heuristic into *SteadyStateSpread* to gain the *SSSbyStep* algorithm. The main idea is to set an iteration threshold β_j for node j, i.e., for each node j, π_j is updated only in the first β_j iterations. They denoted the hops of the maximum influence path from seed set ϕ_0 to node j as $step_j$, then chose $step_j + 1$ as the iteration threshold of node j. We represent the activation probability of node $j \in V$ computed by *SSSbyStep* algorithm as π_j^{s-s} and the *SSSbyStep* algorithm is presented in Algorithm 5.

Algorithm 5 *SSSbyStep*

Input: An independent cascade network $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$;
Output: Activation probability π_j^{s-s} for all nodes $j \in V$;

$t = 0$;
$\pi_j^{s-s}(0) = 1, \forall j \in \phi_0$;
$\pi_j^{s-s}(0) = 0, \forall j \in V \setminus \phi_0$;
Compute $step_j$ for $j \in V \setminus \phi_0$;
$stop = 0$;
while $stop = 0$ **do**
    $stop = 1$;
    $t = t + 1$;
    for $j \in V$ **do**
        if $j \notin \phi_0 \cap t \leq step_j + 1$ **then**
            $\pi_j^{s-s}(t+1) = 1 - \prod_{i \in N_j^{in}} (1 - p_{i,j} \cdot \pi_i^{s-s}(t))$;
        end if
    end for
end while
return $\pi_j^{s-s} = \pi_j(t-1)$.

Yang et al. [44] pointed that their *SSSbyStep* algorithm limits the impact of bidirectional edges as well as includes important influence along maximum influence path. However, intuitively, their algorithm can not guarantee a good result for all network structure. Besides, it is highly time-consuming to compute the maximum influence path.

For partly solving the inaccuracy caused by circuits, we propose in the following an improved algorithm to assure that the iteration process to compute activation probability of node j is not influenced by itself. The new algorithm, that we call *SSS-Noself*, updates the activation probability of a node by Equation 3-1 without the influence of the node itself. We define the value of activation probability computed by *SSS-Noself* for node $j \in V$ as π_j^n.

Given an Independent Cascade model $G_{IC} = (V, E, p)$, a new network $G^{[q]} = (V, E^{[q]}, p^{[q]})$ is obtained from G by removing the input and output arcs of node q. The total number of these new nets is N', where $N' = N - |\phi_0|$, $|\phi_0|$ is the number of nodes in a seed set. For node $j \in V$ one can proceed to compute at each step s the activation probability of j assuming that q has not been activated $\pi_j^{[q]}(s)$. Finally, when updating the activation probability of node j by Equation 3-1, the used value of every j's in-neighbor i is $\pi_i^{[j]}$ obtained by previous iterations. Let us define the activation probability vector of network $G^{[q]}$ $\vec{p^{[q]}}$ as:

$$p_{i,j}^{[q]} = \begin{cases} 0 & \text{if } q = i \text{ or } q = j \\ p_{i,j} & otherwise \end{cases} \tag{3-5}$$

Then the computation function for $\pi_j^{[q]}(s+1)$ is constructed as:

$$\pi_j^{[q]}(s+1) = \begin{cases} 1 & \text{if } j \in \phi_0 \\ 1 - \prod_{i \in N_j^{in}} (1 - \pi_i^{[q]}(s) \cdot p_{i,j}^{[q]}) & \text{if } j \notin \phi_0 \end{cases} \tag{3-6}$$

Algorithm 6 is a modified version of *SteadyStateSpread* where the activation probability of node j is computed disregarding the influence of itself. The computation result for the network in Figure 3.2 by Algorithm 6 is shown in the fourth row of Table 3.2 on page 37. The results of node 2 and node 4 are in bold in Table 3.2 to highlight that they are different from their results by *Path Method* and *SteadyStateSpread*. It is obvious that the result of *SSS-Noself* is closer to the result of *Path Method* than *SteadyStateSpread*, i.e., *SSS-Noself* is more precise than *SteadyStateSpread*. In fact, *SSS-Noself* always gives a result between the result of *Path Method* and the result of *SteadyStateSpread*.

Algorithm 6 *SSS-Noself*

Input: An independent cascade network $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$, stopping criterion $\varepsilon^* > 0$;

Output: Activation probability π_j^n for all nodes $j \in V$;

$t = 0$;
$\varepsilon = \varepsilon^* + 1$;
for $q \in V \setminus \phi_0$ **do**
    $\pi_j^{[q]}(0) = 1, \forall j \in \phi_0$;
    $\pi_j^{[q]}(0) = 0, \forall j \in V \setminus \phi_0$;
end for
$\pi_j(0) = 1, \forall j \in \phi_0$;
$\pi_j(0) = 0, \forall j \in V \setminus \phi_0$;
while $\varepsilon \geq \varepsilon^*$ **do**
    for $q \in V \setminus \phi_0$ **do**
        for $j \in V$ **do**
            if $j \in \phi_0$ **then**
                $\pi_j^{[q]}(t+1) = 1$;
                $\pi_j(t+1) = 1$;
            else
                $\pi_j^{[q]}(t+1) = 1 - \prod_{i \in \mathcal{N}_j^{in}} (1 - p_{i,j}^{[q]} \cdot \pi_i^{[q]}(t))$;
                $\pi_j(t+1) = 1 - \prod_{i \in \mathcal{N}_j^{in}} (1 - p_{i,j} \cdot \pi_i^{[j]}(t))$;
            end if
        end for
    end for
    $\varepsilon_1 = \sum_{j \notin \phi_0} |\pi_j(t+1) - \pi_j(t)|$;
    $\varepsilon_2 = \sum_{j \notin \phi_0} |\pi_j^{[q]}(t+1) - \pi_j^{[q]}(t)|$ $(q \in V \setminus \phi_0)$;
    $\varepsilon = max(\varepsilon_1, \varepsilon_2)$;
    $t = t + 1$;
end while
return $\pi_j^n = \pi_j(t-1)$.

Different from *SteadyStateSpread*, for all nodes $j \in V$, *SSS-Noself* not only computes the activation probability of node j at step s, but also the activation probability of node j at step s assuming that any node $q \in V \setminus \phi_0$ remains inactive. Thus the time complexity of *SSS-Noself* is $O(N^2)$, while that of *SteadyStateSpread* is $O(N)$.

3.4.2.4 Fixed-Point Computation of Influence Propagation Along Paths of Bounded Length

In this part, we propose an efficient algorithm, called *SSS-Bounded-Path*, to compute influence propagation along paths of bounded length by applying Equation 3-1. It represents

a family of efficient approaches for influence propagation computation that is parameterized by the value of the bound b_0. The value of activation probability computed by *SSS-Bounded-Path* with bound $b_0 \in \mathbb{N}$ for node $j \in V$ is denoted as $\pi_j^{bp(b_0)}$.

Let us first define the length of the shortest path from the seed set to a node $j \in V \setminus \phi_0$ as sp_j and assume each node $j \in V \setminus \phi_0$ is reachable from the seed set. Kimura et al. [70] proposed SPM and SP1M, where node j can be activated only at step $b = sp_j$ in SPM, or only at step $b = sp_j$ as well as step $b = sp_j + 1$ in SP1M. Nevertheless, they did not discuss how to compute these probabilities without previously determining the corresponding paths, a procedure that may be computationally expensive. Chen et al. [39] and Yang et al. [44] computed the maximum influence paths by the Dijkstra algorithm, which has high complexity.

We propose an approach based on fixed-point computation that does not require preliminarily computing the shortest path. In our procedure we compute sp_j as step t when π_j^{bp} first changes from zero to non-zero. Besides, we set the path bound to compute π_j^{bp} involving not only the shortest paths but also the paths whose length is no greater than $sp_j + b_0$, where b_0 is a constant integer called bound. Obviously, we have SPM when $b_0 = 0$, and SP1M when $b_0 = 1$. Besides, when b_0 is large enough, this algorithm is equivalent to *SteadyStateSpread*.

The procedure of *SSS-Bounded-Path* is shown in Algorithm 7. We denote the upper bound of iteration time for node j as b_j and it is initialized as infinity. Lines (11-12) find the step when π_j^{bp} firstly changes from zero to non-zero and record this step $t+1$ as sp_j. Then b_j is set as $t+1+b_0$. The computation result for the network in Figure 3.2 by *SSS-Bounded-Path* with $b_0 = \{0, 1, 2, 3\}$ is shown in the fifth row of Table 3.2 on page 37.

The *SSS-Bounded-Path* algorithm generalizes SPM [70] and SP1M [70], exploiting the efficient fixed-point computation of *SteadyStateSpread*. The result of *SSS-Bounded-Path* ($b_0 = 0$) can be regarded as a lower-bound for the exact influence propagation. Same with *SteadyStateSpread*, the time complexity of *SSS-Bounded-Path* is $O(N)$. However, in most cases *SSS-Bounded-Path* ($b_0 = 0$) stops the iteration before it converges, thus *SSS-Bounded-*

Algorithm 7 *SSS-Bounded-Path*

Input: An independent cascade network $G_{IC} = (V, E, p)$, seed set $\phi_0 \subset V$, path bound $b_0 \in \mathbb{N}$, stopping criterion $\varepsilon^* > 0$;
Output: Activation probability π_j^{bp} for all nodes $j \in V$;

Initialize $\pi_j^{bp}(0) = 1$, $j \in \phi_0$; $\pi_j^{bp}(0) = 0$, $j \in V \setminus \phi_0$; $t = 0$; $b_j = inf$, $j \in V \setminus \phi_0$;
$stop = 0$;
$\varepsilon = \varepsilon^* + 1$;
while $\varepsilon \geq \varepsilon^*$ **do**
    while $stop = 0$ **do**
        $stop = 1$;
        for $j \in \mathcal{V}$ **do**
            $\pi_j^{bp}(t+1) = \pi_j^{bp}(t)$;
            if $j \notin \phi_0$ and $t \leq b_j$ **then**
                $\pi_j^{bp}(t+1) = 1 - \prod_{i \in \mathcal{N}_j^{in}} (1 - p_{i,j} \cdot \pi_i^{bp}(t))$;
                $stop = 0$;
            end if
            if $\pi_j^{bp}(t+1) \neq 0$ and $\pi_j^{bp}(t) = 0$ **then**
                $b_j = t + 1 + b_0$;
            end if
        end for
        $t = t + 1$;
    end while
    $\varepsilon = \sum_{j \notin \phi_0} |\pi_j^{bp}(t+1) - \pi_j^{bp}(t)|$;
end while
return $\pi_j^{bp} = \pi_j^{bp}(t-1)$.

Path ($b_0 = 0$) is usually less time-consuming than *SteadyStateSpread*.

3.4.3 Comparison of Different Approaches

We show the influence propagation computation results from different approaches for the network in Figure 3.2 in Table 3.2. As discussed above, *Path Method* provides the exact value, while *SteadyStateSpread* gives a larger value. *SSS-Noself* gives a more precise result than *SteadyStateSpread*, while *SSS-Bounded-Path* ($b_0 = 0$) provides a lower-bound. In fact, the influence propagation value computed by these algorithms must satisfy the inequality, shown in Proposition 3.

Proposition 3. Given an Independent Cascade model $G_{IC} = (V, E, p)$ and a set of initial nodes $\phi_0 \subset V$, the activation probability of node j computed by *Path Method* (π_j^p), *SteadyStateSpread* (π_j^s), *SSS-Bounded-Path* ($b_0 = 0$) ($\pi_j^{bp(0)}$) and *SSS-Noself* (π_j^n) satisfy:

TABLE 3.2 Comparison of activation probability value of each node for the network in Figure 3.2

Node	1	2	3	4	5
Path Method	0.08	**0.0616**	0.4	**0.0123**	1
SteadyStateSpread	0.08	**0.0678**	0.4	**0.0132**	1
SSS-Noself	0.08	**0.0630**	0.4	**0.0126**	1
SSS-Bounded-Path ($b_0 = 0$)	0.08	**0.0400**	0.4	**0.0080**	1
SSS-Bounded-Path ($b_0 = 1$)	0.08	**0.0630**	0.4	**0.0126**	1
SSS-Bounded-Path ($b_0 = 2$)	0.08	**0.0660**	0.4	**0.0132**	1
SSS-Bounded-Path ($b_0 = 3$)	0.08	**0.0678**	0.4	**0.0132**	1

$$\pi_j^{bp(0)} \leq \pi_j^p \leq \pi_j^n \leq \pi_j^s \tag{3-7}$$

Proof. First, we prove that $\pi_j^n \leq \pi_j^s$. During the computing process of π_j^s by Equation 3-1, the activation probabilities of N_j^{in} may include the influence of node j. This extra influence erroneously increases the final value of mode j. *SSS-Noself* algorithm disregards this extra influence during the iteration, hence $\pi_j^n \leq \pi_j^s$.

Second, we prove that $\pi_j^p \leq \pi_j^n$. Although *SSS-Noself* avoids the influence of j when computing π_j^n, it has not eliminated the increase caused by dependency relation and other redundant influence in circuits while applying Equation 3-1. Hence for some special network structures, π_j^n is still bigger than π_j^p.

Finally, we prove that $\pi_j^{bp(0)} \leq \pi_j^p$. *SSS-Bounded-Path* ($b_0 = 0$) only consider the influence to node j through the shortest path, i.e., it disregards the influence through the paths from ϕ_0 to node j whose lengths are greater than sp_j. However, this is just a fraction of the influence on node j, hence we have $\pi_j^{bp(0)} \leq \pi_j^p$. □

Considering *SteadyStateSpread* and *SSS-Bounded-Path*, we also have the following remarks:

Remark 1. Given an Independent Cascade model $G_{IC} = (V, E, p)$ and a set of initial nodes

$\phi_0 \subset V$, for $\forall j \in V$, we have $\pi_j^{bp(b)} \leq \pi_j^{bp(b')}$ when $0 \leq b < b'$.

Obviously, $\pi_j^{bp(b')}$ is computed considering more paths than $\pi_j^{bp(b)}$, i.e., the paths whose lengths are between b and b'. Hence, we have $\pi_j^{bp(b)} \leq \pi_j^{bp(b')}$ when $0 \leq b < b'$.

Remark 2. Given an Independent Cascade model $G_{IC} = (V, E, p)$ and a set of initial nodes $\phi_0 \subset V$, for $\forall j \in V$, we have $\lim_{b\to\infty} \pi_j^{bp(b)} = \pi_j^s$ when the same stopping criterion ε^* is used for both *SteadyStateSpread* and *SSS-Bounded-Path*.

SSS-Bounded-Path limits the computation iteration to b steps. When b goes to infinity, only the satisfaction of stopping criterion halts the computation. In that case, *SSS-Bounded-Path* is equivalent to *SteadyStateSpread*, hence we have $\lim_{b\to\infty} \pi_j^{bp(b)} = \pi_j^s$.

3.5 Experimental Evaluation

In this part, we compare the influence propagation with fixed seed set computed by *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, *SSS-Noself* and *SSS-Bounded-Path*.

All approaches are implemented in MATLAB. All experiments are run on a PC with 2.40GHz Intel Core i5 Processor and 8GB memory.

3.5.1 Data Set

We consider two datasets for comparing the different algorithms we have discussed for influence propagation computation.

First, we construct a series of bidirectional grid graphs with a parameter m such that the $m-$th grid graph contains m^2 nodes. Figure 3.5 shows the grid graphs for $m \in \{2, 3, 4\}$. For each edge (i, j), we uniformly at random select $p_{i,j}$ from the set $\{0.1, 0.2, 0.5\}$. We represent this dataset as **Series-Grid**.

The second dataset is a real-world network — **airportsinUS** [71] which is a benchmark

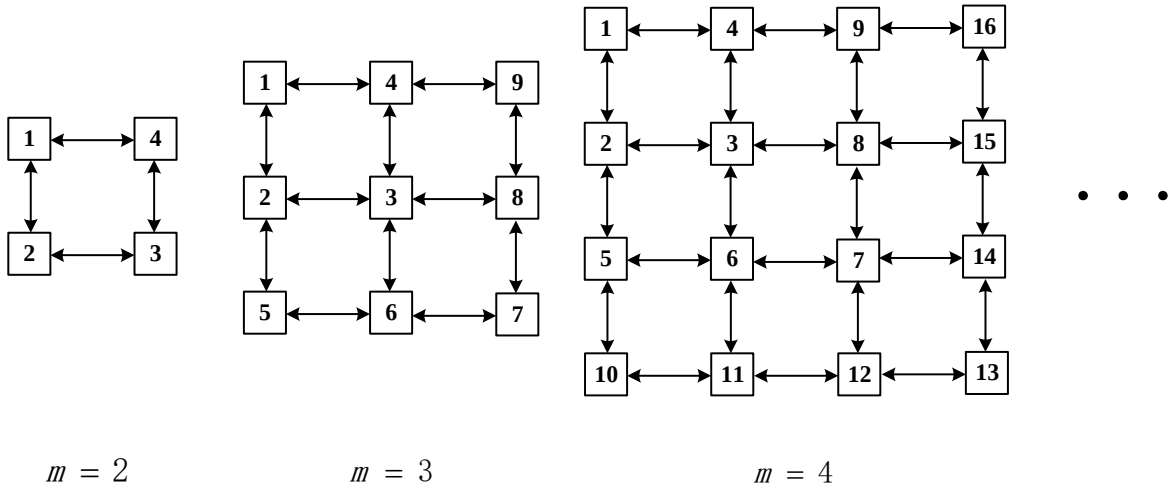

Fig. 3.5 Series of grid graphs

network widely used in social network analysis. It is a weighted network of the 500 airports with the largest amount of traffic from publicly available data in the United States. Nodes represent US airports and edges represent air travel connections among them. There are 5960 edges in total. Based on the weights $w_{i,j}$ of edges, we obtain $p_{i,j}$ by $w_{i,j} \setminus \sum_i w_{i,j}$.

3.5.2 Experimental Setup

We compare *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, *SSS-Noself* and *SSS-Bounded-Path* for influence propagation computation in terms of effectiveness and efficiency. For each network dataset, the seed set is randomly chosen with a certain size. The tested algorithms are briefly described as follows:

Monte Carlo simulation: the average of 10,000 simulation runs. Kempe et al. [11] showed that the quality of approximation after 10,000 iterations is comparable to that after 300,000 or more iterations. The simulation process is described as: assume node i attempts to activate node j at step t, then generate a random number uniformly distributed in the interval $[0, 1]$. The innovation successfully propagates from node i to node j when the random number does not exceed $p_{i,j}$.

- *Path Method*: the exact computation method proposed in Section 3.4.1.

- *SteadyStateSpread*: the heuristic [43] described in Section 3.4.2.
- *SSS-Noself*: the improved algorithm proposed in Section 3.4.2.
- *SSS-Bounded-Path*: the algorithm proposed in Section 3.4.2.

3.5.3 Experimental Results

First, we present the computation results of the influence propagation on Series-Grid using *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, *SSS-Noself* and *SSS-Bounded-Path*. We randomly select one node as seed node for the grid graphs with $m = \{2, 3\}$ and two nodes for the grid graphs with $m = \{4, 5, 6, 7\}$. We set $\varepsilon^* = 10^{-8}$ for the iterations of *SteadyStateSpread*, *SSS-Noself* and *SSS-Bounded-Path*. In order to show the convergence of *SSS-Bounded-Path*, we set $b_0 = \{0, 1, 2, 3, 4, 20, 40, 65, 85, 135\}$. The influence propagation, i.e., the sum of activation probabilities of nodes, on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$ using *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread* and *SSS-Noself* is shown in Table 3.3. The value for $m = \{4, 5, 6, 7\}$ by the *Path Method* is not given since the running time is more than 8 hours, i.e., out of time (o.o.t). The influence propagation on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$ using *SSS-Bounded-Path* is shown in Table 3.4. As a particular case, we list activation probability of each node for the grid graph with $m = 3$ in Table 3.5 and Table 3.6 to show the difference of every node by these five methods.

We can observe that while *SSS-Bounded-Path* ($b_0 = 0$) provides a lower-bound, the result by *SSS-Noself* is always between the exact result by *Path Method* and the result by *SteadyStateSpread* for both activation probability of each node and sum of activation probabilities of all node, i.e., the influence propagation of the network. It verifies the relationship among these four methods in Proposition 3. We can also figure out that within certain paths, the results computed by *SSS-Bounded-Path* converges to the results computed by *SteadyStateSpread*. This is proved in Remark 2. Moreover, it shows that our *SSS-Noself* algorithm provides more precise results than *SteadyStateSpread*.

Second, we compare the running time of these five methods for the influence

TABLE 3.3 Influence propagation computed by *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, and *SSS-Noself* on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$

Method	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
Monte Carlo simulation	1.4343	1.9055	3.2171	4.5072	4.5527	5.1308
Path Method	1.4428	1.9098	o.o.t	o.o.t	o.o.t	o.o.t
SteadyStateSpread	1.4467	2.0936	5.2166	5.8790	7.0053	13.1089
SSS-Noself	1.4428	1.9502	4.0661	5.0710	5.7296	9.1891

TABLE 3.4 Influence propagation computed by *SSS-Bounded-Path* with path bound $b_0 = \{0, 1, 2, 3, 4, 20, 40, 65, 85, 135\}$ on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$

Path bound	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
$b_0 = 0$	1.4396	1.8769	2.8447	3.9560	4.0748	4.5891
$b_0 = 1$	1.4396	1.8769	3.0590	4.1876	4.2843	4.8252
$b_0 = 2$	1.4466	2.0078	3.3066	4.7612	4.8362	5.6117
$b_0 = 3$	1.4466	2.0172	3.5094	4.9387	4.9569	5.7510
$b_0 = 4$	1.4467	2.0566	3.6957	5.2234	5.3128	6.2450
$b_0 = 20$	1.4467	2.0936	5.0722	5.8717	6.8181	10.1654
$b_0 = 40$	1.4467	2.0936	5.2119	5.8790	6.9949	12.5893
$b_0 = 65$	1.4467	2.0936	5.2166	5.8790	7.0050	13.0701
$b_0 = 85$	1.4467	2.0936	5.2166	5.8790	7.0053	13.1045
$b_0 = 135$	1.4467	2.0936	5.2166	5.8790	7.0053	13.1089

TABLE 3.5 Influence propagation computed by *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, and *SSS-Noself* on Series-Grid with $m = 3$

Node	1	2	3	4	5	6	7	8	9
Monte Carlo simulation	1	0.5076	0.0962	0.1099	0.1048	0.0302	0.0151	0.0168	0.0249
Path Method	1	0.5032	0.1009	0.1136	0.1049	0.0302	0.0161	0.0169	0.0238
SteadyStateSpread	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320
SSS-Noself	1	0.5049	0.1119	0.1182	0.1093	0.0345	0.0206	0.0227	0.0280

propagation computation, shown in Table 3.7 and Table 3.8. We can observe that *Path Method* takes exponential time to give exact results as the size of network increases. *SSS-Noself* provides better results than *SteadyStateSpread* with an acceptable increase of computation time for the considered small networks. As b_0 increases, *SSS-Bounded-Path* involves more paths of the network, thus it cost a little more time to compute.

TABLE 3.6 Activation probability of each node computed by *SSS-Bounded-Path* with $b_0 = \{0, 1, 2, 3, 4, 20, 40, 65, 85, 135\}$ on Series-Grid with $m = 3$

Node	1	2	3	4	5	6	7	8	9
$b_0 = 0$	1	0.5000	0.0975	0.1000	0.1000	0.0296	0.0161	0.0137	0.0200
$b_0 = 1$	1	0.5000	0.0975	0.1000	0.1000	0.0296	0.0161	0.0137	0.0200
$b_0 = 2$	1	0.5296	0.1189	0.1264	0.1191	0.0432	0.0236	0.0203	0.0266
$b_0 = 3$	1	0.5296	0.1213	0.1315	0.1191	0.0434	0.0238	0.0208	0.0276
$b_0 = 4$	1	0.5354	0.1279	0.1382	0.1264	0.0491	0.0268	0.0233	0.0296
$b_0 = 20$	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320
$b_0 = 40$	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320
$b_0 = 65$	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320
$b_0 = 85$	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320
$b_0 = 135$	1	0.5392	0.1349	0.1475	0.1317	0.0535	0.0292	0.0255	0.0320

TABLE 3.7 Running time for influence propagation computation by *Monte Carlo simulation*, *Path Method*, *SteadyStateSpread*, and *SSS-Noself* on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$

Running time (s)	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
Monte Carlo simulation	0.48	1.25	0.95	1.40	1.82	1.58
Path Method	0.11	0.31	o.o.t	o.o.t	o.o.t	o.o.t
SteadyStateSpread	0.01	0.02	0.10	0.09	0.34	0.55
SSS-Noself	0.01	0.06	0.66	1.42	4.80	11.41

TABLE 3.8 Running time for influence propagation computation by *SSS-Bounded-Path* with $b_0 = \{0, 1, 2, 3, 4, 20, 40, 65, 85, 135\}$ on Series-Grid with $m = \{2, 3, 4, 5, 6, 7\}$

Running time (s)	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
$b_0 = 0$	0.01	0.01	0.01	0.01	0.01	0.01
$b_0 = 1$	0.01	0.01	0.01	0.01	0.01	0.01
$b_0 = 2$	0.01	0.01	0.01	0.01	0.01	0.01
$b_0 = 3$	0.01	0.01	0.01	0.01	0.01	0.01
$b_0 = 4$	0.01	0.01	0.01	0.01	0.01	0.01
$b_0 = 20$	0.01	0.01	0.01	0.01	0.01	0.02
$b_0 = 40$	0.01	0.01	0.01	0.01	0.02	0.03
$b_0 = 65$	0.01	0.01	0.01	0.02	0.03	0.02
$b_0 = 85$	0.01	0.01	0.01	0.03	0.03	0.04
$b_0 = 135$	0.01	0.02	0.02	0.03	0.05	0.07

Another experiment of influence propagation computation is performed on airportsinUS network data. We evaluate the influence propagation by *Monte Carlo simulation*,

SteadyStateSpread, *SSS-Noself* and *SSS-Bounded-Path* given different sizes of seed sets, shown in Figure 3.6. We have not given the value computed by *Path Method* since it can not be obtained within limited time for this size of network. The seven seed sets are randomly generate with size $|\phi_0| = \{1, 5, 10, 15, 20, 25, 30\}$. The result of *Monte Carlo simulation* is obtained by the average of 10,000 simulation runs proceeded as described in the first experiment. The stopping criterion is fixed as $\varepsilon^* = 0.01$ for *SteadyStateSpread*, *SSS-Noself* and *SSS-Bounded-Path*. The path bound b_0 is chosen from $\{0, 1, 5, 10, 15, 20, 25, 30\}$ for *SSS-Bounded-Path*. We can observe that the results computed by these approaches are consistent with Equation 3-7. According to the proved relationship in Proposition 3, although we have not been able to compute the exact value by *Path Method* because of the net size, we can figure out that *SSS-Noself* is more precise than *SteadyStateSpread*. Moreover, we can see that the results of *SSS-Bounded-Path* increase as the path bound b_0's increase. As a lower-bound, the values computed by *SSS-Bounded-Path* ($b_0 = 0$) are the smallest among all approaches under the same seed set size.

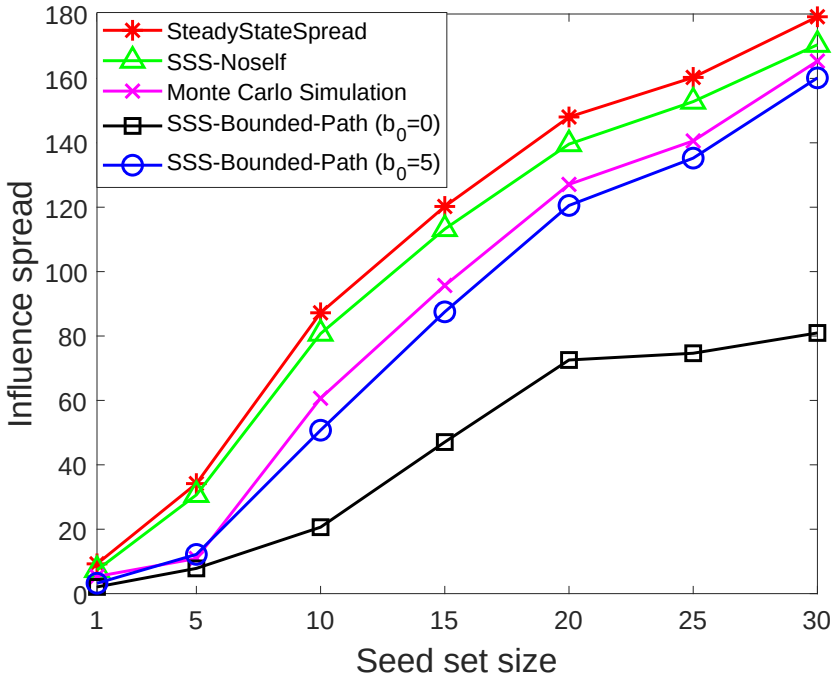

Fig. 3.6 Influence propagation computed by *Monte Carlo simulation*, *SteadyStateSpread*, *SSS-Noself*, *SSS-Bounded-Path* ($b_0 = 0$) and *SSS-Bounded-Path* ($b_0 = 5$) on airportsinUS network data

However, as shown in Table 5.3, the running time of *SSS-Noself* is much longer than *SteadyStateSpread* when the size of the network is large since *SSS-Noself* needs one more

pass of all nodes in a network than *SteadyStateSpread*. For this reason we think that it may be necessary to further improve the efficiency of *SSS-Noself*. Compared with the running time of *SSS-Bounded-Path* in Table 3.10, *SSS-Bounded-Path* is obviously faster than *SteadyStateSpread* when $b_0 = \{0, 1, 5, 10, 15, 20\}$. As b_0 increases, the running time of these two approaches will be similar.

TABLE 3.9 Running time for influence propagation computation by *Monte Carlo simulation*, *SteadyStateSpread* and *SSS-Noself* on airportsinUS network data given different sizes of seed sets $|\phi_0|$

Running time (s)	$\|\phi_0\| = 1$	$\|\phi_0\| = 5$	$\|\phi_0\| = 10$	$\|\phi_0\| = 15$	$\|\phi_0\| = 20$	$\|\phi_0\| = 25$	$\|\phi_0\| = 30$
Monte Carlo simulation	3.86	6.23	36.39	57.70	68.81	98.59	110.99
SteadyStateSpread	5.05	3.20	1.35	1.18	1.10	0.94	0.82
SSS-Noself	1143.62	892.77	719.44	467.74	512.77	383.86	350.49

TABLE 3.10 Running time for influence propagation computation by *SSS-Bounded-Path* on airportsinUS network data given different sizes of seed sets $|\phi_0|$ with path bound $b_0 = \{0, 1, 5, 10, 15, 20, 25, 30\}$

Running time (s)	$\|\phi_0\| = 1$	$\|\phi_0\| = 5$	$\|\phi_0\| = 10$	$\|\phi_0\| = 15$	$\|\phi_0\| = 20$	$\|\phi_0\| = 25$	$\|\phi_0\| = 30$
$b_0 = 0$	0.12	0.12	0.11	0.14	0.12	0.11	0.12
$b_0 = 1$	0.16	0.22	0.16	0.18	0.16	0.15	0.16
$b_0 = 5$	0.26	0.28	0.28	0.30	0.27	0.27	0.28
$b_0 = 10$	0.36	0.38	0.38	0.40	0.38	0.37	0.37
$b_0 = 15$	0.61	0.57	0.56	0.58	0.56	0.56	0.57
$b_0 = 20$	0.57	0.64	0.63	0.65	0.64	0.63	0.62
$b_0 = 25$	0.87	0.87	0.86	0.88	0.86	0.86	0.86
$b_0 = 30$	0.90	0.90	0.91	0.91	0.89	0.88	0.89

Figure 3.7 shows the error between *SSS-Bounded-Path* and *SSS-Noself* which is measured by $|\sum_{j\in\mathcal{V}\setminus\phi_0} \pi_j^{bp} - \sum_{j\in\mathcal{V}\setminus\phi_0} \pi_j^n| \setminus \sum_{j\in\mathcal{V}\setminus\phi_0} \pi_j^n$. We do not present the curve for $|\phi_0| = 25$ due to the limit of space, but point out that it is similar to the curves for $|\phi_0| = \{15, 20, 30\}$. We can find that in the beginning the error decreases as the path bound b_0's increases. At certain path bound the error is the smallest and then increases a bit. Results show that the path bound corresponding to the smallest error varies with different seed set size.

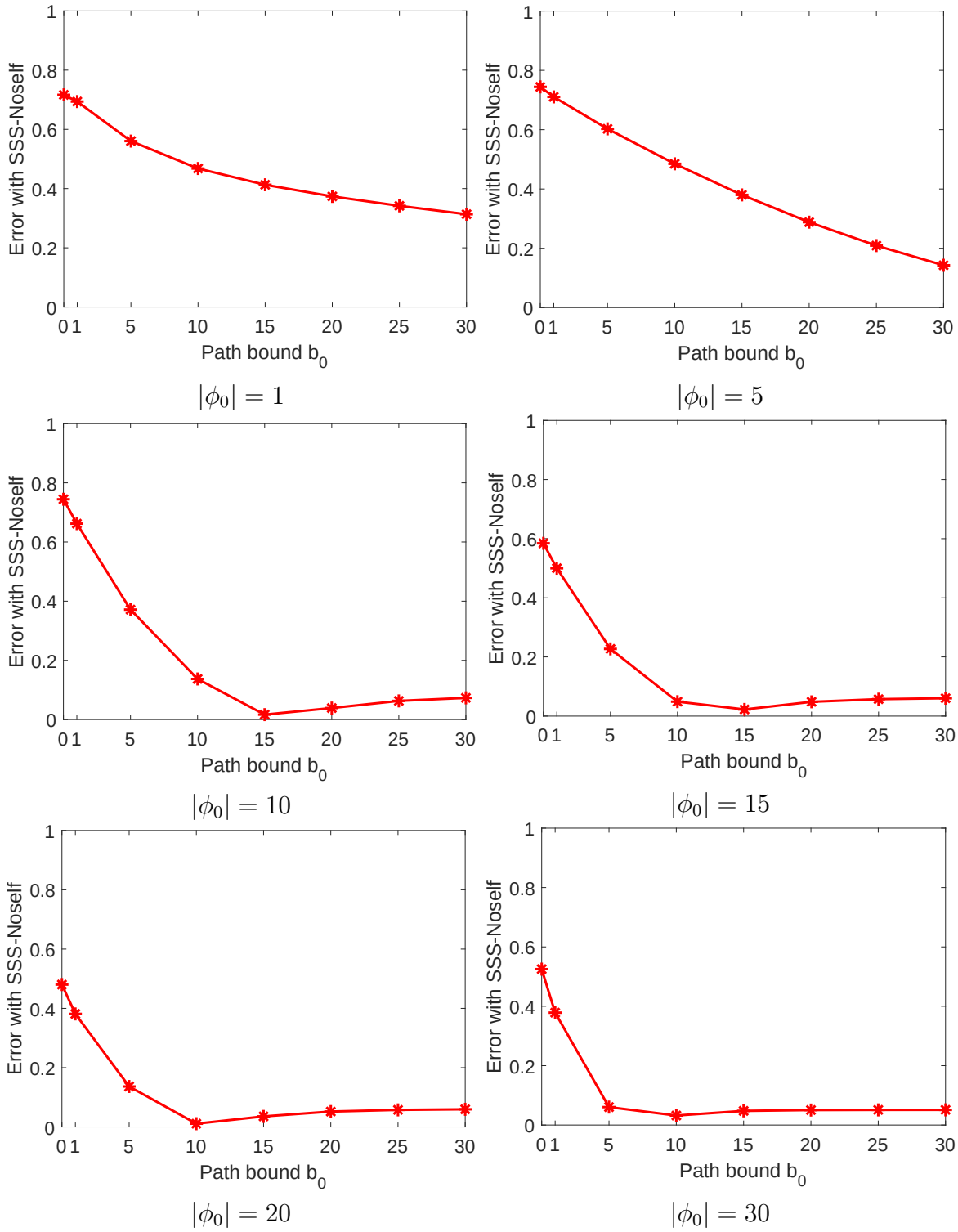

Fig. 3.7 Error between *SSS-Bounded-t* and *SSS-Noself* on airportsinUS network data given different sizes of seed sets $|\phi_0|$ with path bound $b_0 = \{1, 5, 10, 15, 20, 30\}$

3.6 Conclusion and Future Work

In this chapter, we analyze different approaches for influence propagation computation in an Independent Cascade model. We initially propose an approach which can compute the exact value of the influence propagation called *Path Method*. We also discuss the convergence properties of the existing algorithm *SteadyStateSpread*, showing that it converges to a fixed solution by fixed-point theory. We consider the elements resulting in the inaccuracy of *SteadyStateSpread*: the dependency relation between nodes and the existence of circuits. Furthermore, we show how to compute a lower approximation of influence propagation by *SSS-Bounded-Path* and propose an improved algorithm called *SSS-Noself* which partially decreases the error caused by circuits.

The objective of our future work will be to focus on proposing new algorithms to improve efficiency while taking into account the factors that are at the root cause of the inaccuracy of the *SteadyStateSpread* approach. Besides, since we observed that *SSS-Noself* is quite time consuming in large-scale networks, another interesting line of research could be to improve the efficiency of the *SSS-Noself*.

These computational approaches for estimating the influence propagation are applied to solve the influence maximization problem in next chapter.

Chapter 4 Influence Maximization by Seed Selection

4.1 Introduction

As mentioned in Chapter 1, one strong motivation for studying the influence propagation is viral marketing. When one promotes a new product or information through social networks by word-of-mouth effect, a cost-effective way is to target a set of initial adopters in the network, by investing resources. The aim is that these initial users will drive other individuals of the network to adopt the same product or information, thus leading to a potentially large propagation in the network.

The problem of influence maximization by seed selection can be described as follows: given a social network and an influence propagation model, choose a seed set of up to K nodes such that the final influence propagation is maximized.

To solve the problem of influence maximization by seed selection, Kempe et al. [11] firstly formulated the problem as a discrete stochastic optimization problem and proved that the problem in both the Linear Threshold model and Independent Cascade model is NP-hard. Then they proposed a greedy algorithm based on *Monte Carlo simulation* which guarantees that the influence propagation approximates the optimal one within a factor of $(1 - 1/e)$. However, this approach is time-consuming, thus subsequent work concentrated on improving the efficiency of algorithmic approaches.

In this chapter, we discuss how it is possible to use the algorithms for influence propagation computation proposed in Chapter 3- *SteadyStateSpread* and *SSS-Noself* together with *SelectTop*K, *RankedReplace* and greedy algorithm-to select a (possibly optimal) seed set.

4.2 Related Work

The influence maximization as an algorithmic technique for viral marketing was first proposed by Domingos and Richardson [20], within a probabilistic framework based on Markov random fields. Kempe et al. [40] also formulated the issue of choosing influential sets of individuals as a discrete optimization problem. It aims to identify a small subset of initial adopters in a social network to maximize the influence propagation under a given diffusion model. They also proved this influence maximization problem [72–75] is NP-hard and gave a greedy approximation algorithm which guarantees, under certain conditions, that the influence propagation approximates the optimal one within a factor of $(1 - 1/e)$, where e is the base of the natural logarithm. However, this approach requires a long time to run the simulation, thus later much effort was devoted to improve the efficiency of the algorithm. We grouped the subsequent solutions into three categories: approximation algorithms with provable guarantee, heuristic approaches and community-based approaches.

Firstly, the following approximation algorithms with provable guarantee offer significant improvement compared to the initial greedy algorithm proposed by Kempe et al. [40] in terms of efficiency. Leskovec et al. [41] proposed a "Cost-Effective Lazy Forward" (CELF) scheme to reduce the number of evaluations on the influence propagation, nevertheless it still satisfies the approximation guarantee. To further improve the CELF heuristic, Chen et al. [47] presented MixedGreedyIC algorithm for the Independent Cascade model and MixedGreedyWC algorithm for the Weighted Cascade model. Goyal et al. [42] explored the CELF++ approach. Cheng et al. [76] derived StaticGreedy algorithm and a dynamical update strategy to provide both guaranteed accuracy and high scalability. Borgs et al. [48] developed an elegant framework, named Reverse Influence Sampling (RIS), focusing on the reduction of running time. Zhu et al. [77] converted the problem into a quadratic integer programming problem and solved by the concept of semidefinite programming, which improved the approximation guarantee from $1 - 1/e$ to 0.857. Cohen et al. [78] proposed a Sketch-Based Influence Maximization (SKIM) algorithm which also has high scalability. Tang et al. [49, 50] proposed the hop-based algorithm for both the Independent Cascade model and Linear Threshold model. Recently, Nguyen et al. [79] developed Billion-

scale Cost-award Targeted (BCT) algorithm for solving cost-aware targeted viral marketing (CTVM) introduced by them. Their methodology can be adopted for solving the problem of influence maximization by seed selection in both the Linear Threshold model and the Independent Cascade model.

Secondly, the heuristic solutions do not provide an approximation bound but result in faster running time and higher scalability. Chen et al. [39] and Wang et al. [80] proposed Maximum Influence Arborescence (MIA) and Prefix excluding MIA (PMIA) model for influence propagation. Experimental results showed that both MIA and PMIA can achieve high scalability. Jung et al. [81] developed a novel IRIE algorithm based on influence ranking (IR) and influence estimation (IE) in both the Independent Cascade model and its extension IC-N which incorporates negative opinion propagation. Galhotra et al. [82] proposed Opinion Cum Interaction (OCI) model and formulated a problem of Maximizing the Effective Opinion (MEO). They introduced two heuristics, namely Opinion Spread Influence Maximization (OSIM) and EaSyIm to solve the MEO problem. Then Cordasco et al. [83, 84] derived a heuristic which can produce optimal solution for trees, cycles, and complete graphs.

Thirdly, a community is basically a subset of nodes, densely connected among themselves and sparsely connected with the other nodes. There are numerous solutions based on the community-based framework. Wang et al. [85] proposed the community-based greedy algorithm which consist of detecting communities based on information propagation and selecting communities for finding influential nodes. Chen et al. [86, 87] developed a new framework, community-based influence maximization (CIM). Shang et al. [88] proposed CoFIM, a community-based framework in which the influence propagation process is divided into two phases: seeds expansion; and intra-community propagation. Li et al. [89] developed a community-based seeds selection (CSS) algorithm. The CSS algorithm finds seeds efficiently by constructing the PR-tree based indexes offline that precompute users' community based influences, and preferentially computing the marginal influences of those who would be selected as seeds with high probability online.

4.3 Influence Maximization by Seed Selection in the Independent Cascade Model

4.3.1 Problem Formulation

The problem of influence maximization by seed selection aims to maximize the influence propagation through a social network, by targeting a subset of individuals to adopt an innovation initially. We formalize it under the Independent Cascade model as follows.

Problem 1. *Given an Independent Cascade model $G_{IC} = (V, E, p)$ and a constant integer K, find a seed set $\phi_0 \subseteq V$ of cardinality $|\phi_0| = K$, such that the influence propagation $\sigma(\phi_0)$ is maximized, i.e.,*

$$\phi_0 = \underset{\phi_0 \subseteq V}{argmax}\{\sigma(\phi_0) || \phi_0| = K\}.$$

Note that solving this problem requires identifying a set of K nodes with the largest influence which is different from identifying the K nodes with the largest individual influence. For instance, if two nodes both have the largest influence on the same set of individuals, it is sufficient that only one of them is selected as a seed node. In other words, the K nodes with the largest individual influence are not always the best choice for seed nodes.

4.3.2 Hardness of Influence Maximization by Seed Selection

Kempe et al. [11] proved that the problem of influence maximization by seed selection is NP-hard, which is based on the hardness of influence propagation computation. In fact, due to the combinatorial property of finding a set of K seed nodes when K is an input instead of being a constant, the influence under Independent Cascade model also contains NP-complete problems as special cases, making it NP-hard without relying on the counting hardness of influence propagation computation as shown in Theorem 4. The detailed proof for Theorem 4 can be found in [11].

Theorem 4. *[11] The influence maximization problem is NP-hard in the Independent*

Cascade model.

4.3.3 Greedy Approach and Approximation Guarantee

It has been shown that the problem of influence maximization by seed selection is hard to solve exactly. Kempe et al. [11, 40] gave a greedy approach with *Monte Carlo simulation* which can achieve a $(1 - 1/e)$ approximation guarantee. We describe the general greedy approach in Algorithm 8.

Algorithm 8 Greedy Algorithm

Input: a monotone and submodular set function f, an integer $K \in \mathbb{N}^+$;
Output: the selected subset S;

```
Initialize S = ∅;
for q = 1 to K do
    Select i = argmax_{j∈V\S}{σ(S ∪ {j}) − σ(S)};
    S = S ∪ {i};
end for
return set S.
```

Nemhauser et al. [68] proved that when set function f is monotone and submodular, the greedy algorithm can achieve an approximation guarantee shown in Theorem 5.

Theorem 5. *[11] For a non-negative, monotone submodular function f, let ϕ_0 be a set of size K obtained by selecting one node at a time, each time choosing a node that provides the largest marginal increase in the function value. Let ϕ_0^* be a set that maximizes the value of f over all K-node sets. Then $f(\phi_0) \geq (1 - 1/e) \cdot f(\phi_0^*)$; i.e., ϕ_0 provides a $(1 - 1/e)$-approximation, where e is the base of natural logarithm.*

4.4 Methodology

In this section, we present three basic algorithms, named *SelectTop$K$*, *RankedReplace* and greedy algorithm, which will later be combined with the approaches for influence propagation computation in Section 3 to solve the influence maximization problem.

4.4.1 *SelectTopK* Algorithm

In order to select a set of K nodes to maximize the final influence propagation, the basic idea is as follows: let each node $j \in V$ be the single seed node, i.e., $\phi_0 = \{j\}$, then compute the influence propagation by one of algorithms discussed in Section 3. Knowing the influence propagation when each node is set as the single seed node, we select the K nodes with the largest influence propagation as seed set ϕ_0. The detail is described in Algorithm 9.

Algorithm 9 *SelectTopK*

Input: An independent cascade network $G_{IC} = (V, E, p)$, an integer $K \in \mathbb{N}^+$;
Output: Seed set ϕ_0;
1: Compute the influence propagation $\sigma(\{j\})$ for each node $j \in V$;
2: Select K nodes with the highest value of $\sigma(\cdot)$ as seed set ϕ_0;
3: **return** seed set ϕ_0.

Let T be the time complexity required to compute the influence propagation for a given network with N nodes. As we have previously discussed, these values have order $O(N \cdot 3^N)$ for *Path Method*, $O(N)$ for *SteadyStateSpread*, $O(N^2)$ for *SSS-Noself* and $O(N)$ for *SSS-Bounded-Path*. Since *SelectTopK* computes the influence propagation for each node $j \in V$ as a single seed node, we have Proposition 4.

Proposition 4. The time complexity of *SelectTopK* is $O(NT)$, with $N = |V|$.

4.4.2 *RankedReplace* Algorithm

To further improve the SelectTop$K$ algorithm, a number of replacements of seed nodes happen after the selection of initial seed set. As shown in Algorithm 10 [43], the nodes in $V \setminus \phi_0$ are sorted in descending order of the influence propagation value $\sigma(\phi_0)$. Then in each iteration, the nodes in ϕ_0 are sorted in ascending order of influence propagation value $\sigma(\phi_0)$. We pick in order the node in $V \setminus \phi_0$ to replace a node in ϕ_0, if this replacement can increase the influence propagation. Note that in ascending order only the first replacement of a node in ϕ_0 which increases $\sigma(\phi_0)$ is executed.

Let T be the time complexity required to compute the influence propagation for a given network with N nodes. As we have previously discussed, these values have order $O(N \cdot 3^N)$

Algorithm 10 *RankedReplace*

Input: An independent cascade network $G_{IC} = (V, E, p)$, an integer $K \in \mathbb{N}^+$;
Output: Seed set ϕ_0;

```
Compute the influence propagation σ({j}) for each node j ∈ V;
Initialize seed set φ_0 by K nodes with the highest value of σ(·);
Sort nodes j ∈ V \ φ_0 in descending order of σ({j});
for j ∈ V \ φ_0 in descending order of σ(j) do
    Sort nodes i ∈ φ_0 in ascending order of σ({i});
    for i ∈ φ_0 in ascending order of σ({i}) do
        if σ(φ_0 ∪ {j} \ {i}) > σ(φ_0) then
            φ_0 = φ_0 ∪ {j} \ {i};
            break;
        end if
    end for
end for
return seed set φ_0.
```

for *Path Method*, $O(N)$ for *SteadyStateSpread*, $O(N^2)$ for *SSS-Noself* and $O(N)$ for *SSS-Bounded-Path*. Then we have Proposition 5.

Proposition 5. The time complexity of *RankedReplace* is $O(K(N-K)T)$, with $N = |V|$ and $K = |\phi_0|$.

4.4.3 Greedy Algorithm

Algorithm 11 describes the general greedy algorithm for influence maximization which can guarantee that the influence propagation ϕ_0 is within $(1 - 1/e)$ of the optimal value, as pointed by Kempes et al. [11]. In this algorithm, the node which maximizes the incremental influence propagation is selected in each iteration.

Algorithm 11 Greedy Algorithm

Input: An independent cascade network $G_{IC} = (V, E, p)$, an integer $K \in \mathbb{N}^+$;
Output: Seed set ϕ_0;

```
Initialize φ_0 = ∅;
for q = 1 to K do
    Select i = argmax_{j∈V\φ_0} {σ(φ_0 ∪ {j}) − σ(φ_0)};
    φ_0 = φ_0 ∪ {i};
end for
return seed set φ_0.
```

Let T be time complexity required to compute the influence propagation for a given

network with N nodes. As we have previously discussed, these values have order $O(N \cdot 3^N)$ for *Path Method*, $O(N)$ for *SteadyStateSpread*, $O(N^2)$ for *SSS-Noself* and $O(N)$ for *SSS-Bounded-Path*. Then we have Proposition 6.

Proposition 6. The time complexity of greedy algorithm is $O(KNT)$, with $N = |V|$ and $K = |\phi_0|$.

4.5 Experimental Evaluation

4.5.1 Data Set

In this part, the real-world dataset used in seed selection for influence maximization is **HighSchool** [90]. It is a directed network, containing friendship links among 73 boys in a small high-school in Illinois. A node represents a boy and an edge from node i to node j shows that the $i-$th boy chose the $j-$th boy as a friend. The activation probability $p_{i,j}$ is randomly selected from the set $\{0.1, 0.2, 0.5\}$.

4.5.2 Experimental Setup

In this part of experiment, we evaluate the performances of *SteadyStateSpread* and *SSS-Noself* in terms of selecting seed set to maximize the influence propagation. During the process of seed set selection by *SelectTop*K, *RankedReplace* or greedy algorithm, we apply *SteadyStateSpread* or *SSS-Noself* to compute the influence propagation. After selecting the seed set by these different combination of methods, the influence propagation of the selected seed set is evaluated by running *Monte Carlo simulation* for 10,000 times. The tolerance for *SteadyStateSpread* and *SSS-Noself* is fixed as $\varepsilon^* = 0.01$. The tested algorithms are briefly described as following:

Random: Randomly select a set of nodes to be activated.

SelectTop$K$-SSS: Compute the influence propagation by *SteadyStateSpread* and then select K nodes with the largest influence propagation as the seed nodes.

SelectTop$K$-SN: Compute the influence propagation by *SSS-Noself* and then select K nodes with the largest influence as the seed nodes.

Replace-SSS: Compute the influence propagation by *SteadyStateSpread* and then select seed nodes by *RankedReplace*.

Replace-SN: Compute the influence propagation by *SSS-Noself* and then select seed nodes by *RankedReplace*.

Greedy-SSS: Select seed nodes by greedy algorithm, in which computing the influence propagation by *SteadyStateSpread*.

Greedy-SN: Select seed nodes by greedy algorithm, in which computing the influence propagation by *SSS-Noself*.

4.5.3 Experimental Results

We evaluate the algorithms above on the Highschool network under the Independent Cascade model in terms of the influence propagation and the running time. The influence propagation is denoted with the total activation probabilities of all nodes in the network. After selecting a seed set by any one of the above algorithms, the influence propagation is computed by *Monte Carlo simulation*.

The influence propagation with a seed set of size $K = \{1, 5, 10, 15, 20, 25\}$ computed by different algorithms is shown in Figure 4.1. Regardless of which approach computes the influence propagation, either *SteadyStateSpread* or *SSS-Noself*, it is obvious that greedy algorithm performs better than *RankedReplace*, and *RankedReplace* performs better than *SelectTopK* when $K = \{5, 10, 15, 20, 25\}$. Moreover, based on the same algorithm for seed set selection (*SelectTopK*, *RankedReplace* or greedy algorithm), *SSS-Noself* can select a better set of seed nodes, i.e., give a larger influence propagation, compared with *SteadyStateSpread*.

Figure 4.2 shows the running time for selecting nodes by different algorithms above. *Greedy*-SN takes much longer time than other algorithms. Although *SelectTop*K-SN takes

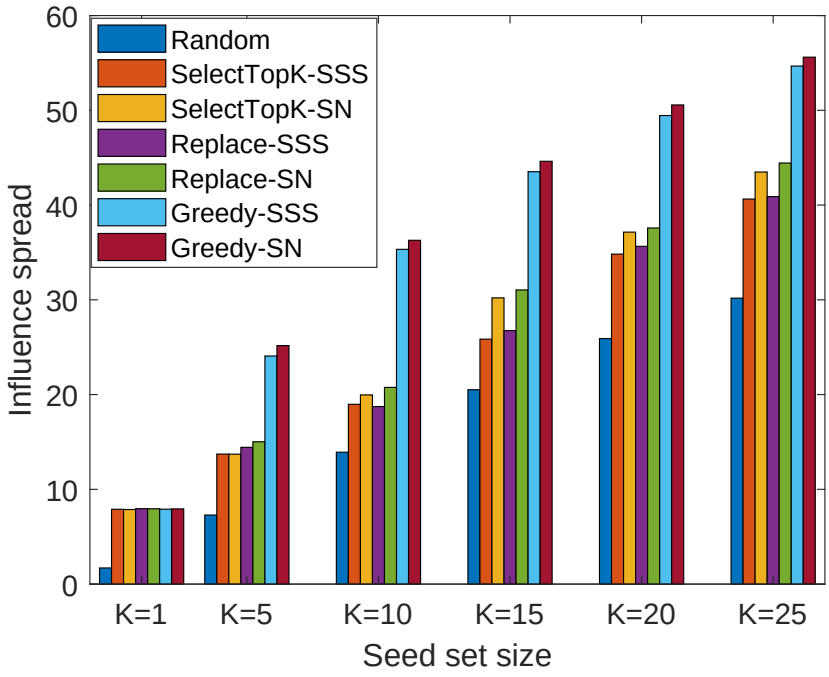

Fig. 4.1 Influence propagation computed by different algorithms on Highschool network data given size of seed set $K = \{1, 5, 10, 15, 20, 25\}$

a bit longer time than *SelectTop*K-SSS and as well *Replace*-SN takes a bit longer time than *Replace*-SSS, these four algorithms are efficient enough.

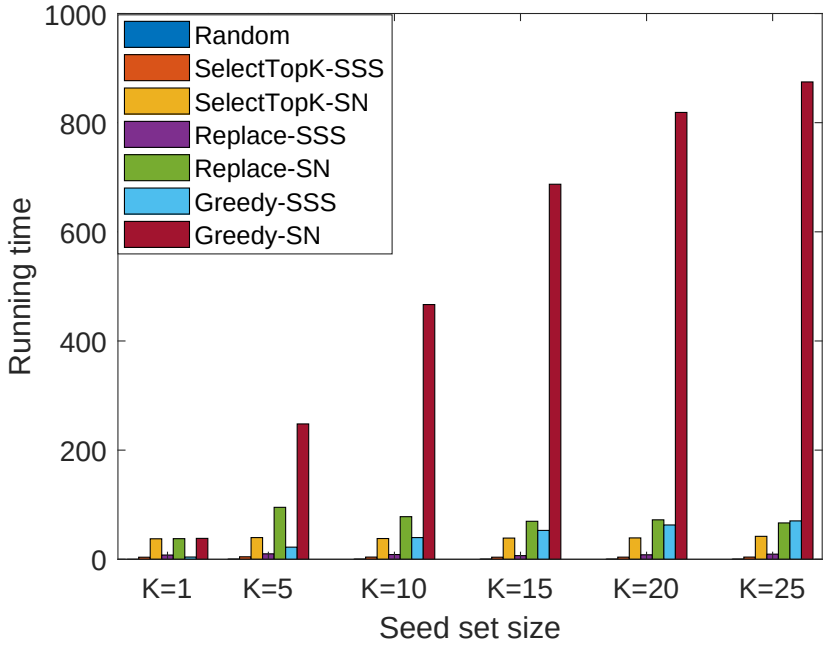

Fig. 4.2 Running time for selecting seed nodes by different algorithms on Highschool network data given size of seed set $K = \{1, 5, 10, 15, 20, 25\}$

4.6 Conclusion and Future Work

Focusing on the influence maximization problem by seed selection, we evaluate the approaches to influence propagation computation: *SteadyStateSpread* and *SSS-Noself* in terms of selecting seed set by combining a selection strategy among *SelectTopK*, *RankedReplace* and greedy algorithm. The experiments performed in real networks show that *Greedy*-SSS and *Greedy*-SN can achieve a larger influence propagation, i.e., select more effective seed set. However, *Greedy*-SN cost much longer time.

Further research directions can be summarized as follows:

- Due to the complexity of computation algorithms, our approaches are not suitable for large-scale networks. Thus it is important to develop more scalable heuristics to handle large network datasets.
- The classic problem of influence maximization by seed selection assumes that the cost for activating a node and the influence ability of each node are the same. However, it is more practical to consider the case where these two parameters differs for different nodes, which is called profit maximization, rather than influence maximization.

Chapter 5 Budgeted Influence Maximization by Link Activation

5.1 Introduction

As mentioned in previous chapters, influence propagation in social networks has been widely studied recently. Previous research mostly focuses on either maximizing the influence by identifying a set of initial adopters, or minimizing the influence by link blocking, node blocking or competitive strategies under a certain diffusion model. In this chapter, we address a budgeted influence maximization problem considering the link activation under the Independent Cascade model.

We assume that the network and the number of initial adopters are given. However, unlike previous approaches, we regard the set of initial adopters as a stochastic variable, while the decision variables that we choose to maximize the influence propagation are the active links in the network. Activating a link has a certain cost and we have a limited budget for that. Our aim is to activate a set of links within a limited budget such as to maximize the expected influence propagation.

We believe that the Independent Cascade model can describe quite well the decision of a company that has a given budget for a publicity campaign. The links of the network could represent different ways in which the influence may propagate and the objective of the campaign is supporting the most successful ones so as to maximize the publicity propagation.

We control the links in order to maximize the influence propagation. Previously, the link control has been used only for influence minimization. Most of the work which aimed to solve the influence minimization problem by link control applied "link blocking" [51–57]. They proposed various approaches to limit the spread of negative innovation such as injection, rumor, virus, etc., in either preventive or reactive way. A preventive strategy

focuses on the network topology modification in order to make the network more resistant to a negative innovation. The algorithm proposed by Tong et al. [55] optimized the leading eigenvalue of the network adjacency matrix to control the influence dissemination process. Unlike the preventive strategy, a reactive strategy takes the initially affected nodes into account to guide the link removal operation. Kimura et al. [53, 58] combined the bond percolation method with the greedy algorithm to approximately solve this problem and compared with the link-removal heuristics based on betweenness and out-degree.

To the best of our knowledge, this is the first study on the influence maximization by means of link activation considering the Independent Cascade model. Unlike other works on influence maximization which focused on selecting a certain number of optimal initial adopters, we aim to activate the most effective links within a limited budget to achieve influence maximization. Unfortunately, the heuristics we consider cannot provide an optimal solution to this problem, but our SimID algorithm is proved to guarantee, under a certain constraint, that the influence propagation is within $\frac{1}{2}(1-\frac{1}{e})$ of the optimal solution. Specifically, our main contribution can be summarized as follows:

- We initially propose the problem of influence maximization by link activation under a limited budget and prove that this problem is NP-hard.

- We analyze the monotonicity and submodularity of our function of expectation of influence propagation (EIP for short) on two variables: seed set and active links. It shows that the function is monotone and submodular with respect to the variable of seed set, but monotone and non-submodular with respect to the variable of active links.

- Aiming at activating a set of links to achieve the maximum influence propagation, we propose two heuristics based on a cost-degree coefficient as well as two heuristics based on an inf-degree coefficient. We prove that our SimID algorithm can activate a set of links to gain the EIP which is within $\frac{1}{2}(1-\frac{1}{e})$ of the optimal one under a certain constraint.

5.2 Related Work

We recall some previous work addressing two related problems: budgeted influence maximization by individual selection and influence minimization by link control.

Firstly, most previous work about budgeted influence maximization focused on the individual selection, i.e., identify a set of influential individuals to gain a maximum influence propagation within a limited budget. Nguyen and Zheng [91] proposed a selection algorithm which guarantees an approximation ratio of $(1 - \frac{1}{\sqrt{e}})$. Han et al. [92] addressed this problem by a balanced seed selection algorithm which combines three different selection mechanisms. Güney [93] reformulated the problem as a mixed integer linear program and proposed a sample average approximation (SAA) scheme. Recently a community-based solution approach is presented by Banerjee et al. [94] to solve the problem.

Secondly, the studies related to link control mostly aimed to solve the influence minimization problem, i.e., minimizing the propagation of negative innovation by blocking a number of links of a network. Kimura et al. [53, 58, 95] considered minimizing both the average contamination degree and the worst contamination degree on basis of a bond percolation mechanism. Nandi et al. [57] proposed mixed-integer programming formulations of four network interdiction models for removing a set of links from a network to minimize the negative influence propagation. Enns et al. [51] evaluated both rank-based and optimization-based approaches for link blocking in various networks.

Different from above two groups of previous studies, our work focuses on solving the influence maximization problem within a limited budget by link activation, rather than by individual selection. It opens another door for network control to achieve a certain goal, e.g., influence maximization. We believe it benefits a lot the management of online network interactive platform, transportation system, etc.

5.3 Budgeted Influence Maximization by Link Activation in the Independent Cascade Model

5.3.1 Problem Formulation

We assume that in a given Independent Cascade network $G_{IC} = (V, E, p)$, edges in E are normally inactive but may activated by an external control agent. Assume we are given a *cost vector* $\mathbf{c} \in V \times V$, where *activation cost* $c_{i,j}$ denotes the cost for activating the link (i, j) between node i and node j. Let us denote $\Phi_K \subseteq 2^V$ the set of all subsets of V of cardinality K: we assume that the seed set is a uniformly distributed random variable, which takes value ϕ_0 with probability $pr(\phi_0) = \frac{1}{|\Phi_K|}$ for all $\phi_0 \in \Phi_K$, where $|\Phi_K|$ is the number of elements in Φ_K. Then we activate a set of links $E_a \subseteq E$ to construct an *active graph* and the total cost should not exceed a budget $\mathcal{B}$. The set of nodes connected by E_a, which are denoted as *potentially active nodes*, is represented by V_a. The active graph is represented by $\hat{G}_{IC} = (V_a, E_a, p)$. The goal is to maximize the expectation of influence propagation (EIP), denoted by $\mathbb{E}[\sigma(E_a, \phi_0)]$ by the activation of the set of links E_a. We formalize this problem as follow:

Problem 2. *Given an Independent Cascade model $G_{IC} = (V, E, p)$ and a constant integer K, let $\boldsymbol{c} \in V \times V$ be a cost vector. $\Phi_K \subseteq 2^V$ denotes the set of all subsets of V of given cardinality K. The seed set ϕ_0, i.e., the initial state of the network, is a random variable taking values in Φ_K with probability $pr(\phi_0) = \frac{1}{|\Phi_K|}$. Activate a set of edges $E_a \subseteq E$ to construct an active graph $\hat{G}_{IC} = (V_a, E_a, p)$, such that the expectation of influence propagation (EIP) $\mathbb{E}[\sigma(E_a, \phi_0)]$ is maximized and the total cost for activating the edges $(i, j) \in E_a$ is no more than a budget $\mathcal{B}$, i.e.,*

$$
\begin{aligned}
max \quad & \mathbb{E}[\sigma(E_a, \phi_0)] \\
s.t. \quad & \sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B} \\
& E_a \subseteq E \\
& \phi_0 \subseteq \Phi_K \subseteq 2^V \\
& \mathcal{B} \in \mathbb{R}_+
\end{aligned}
\tag{5-1}
$$

where

$$\begin{aligned}\mathbb{E}[\sigma(E_a,\phi_0)] &= \sum_{\phi_0\in\Phi_K}\sigma(E_a,\phi_0)\cdot p_r(\phi_0)\\ &= \frac{1}{|\Phi_K|}\cdot\sum_{\phi_0\in\Phi_K}\sigma(E_a,\phi_0),\end{aligned}$$

and

$$\sigma(E_a,\phi_0) = \sum_{j\in V_a\setminus\phi_0}\pi_j. \tag{5-2}$$

Note that this is a stochastic optimization problem where both the input data (the seed set ϕ_0) and the system's performance index (the EIP) are random variables.

5.3.2 Submodularity and Monotonicity

In order to analyse the monotonicity and submodularity of the EIP function with respect to both seed set and active links, we recall a closure property of monotone functions [96] in Proposition 7 and a closure property of submodular functions [97] in Proposition 8.

Proposition 7. If for $i \in \{1,2,...,m\}$, set functions $f_i : 2^V \to \mathbb{R}$ are all monotone and $\alpha_i \geq 0$, then the positive linear combination

$$S = \sum_{i=1}^{n}\alpha_i f_i(S)$$

is monotone.

Proposition 8. If for all $i \in \{1,2,...,m\}$, set functions $f_i : 2^V \to \mathbb{R}$ are all submodular and $\alpha_i \geq 0$, then the positive linear combination

$$S = \sum_{i=1}^{n}\alpha_i f_i(S)$$

is submodular.

First, when the set of active links E_a is fixed, for a given seed set ϕ_0 the influence

propagation function $\sigma(E_a, \phi_0)$ coincides with the usual influence propagation function $\sigma(\phi_0)$ for network (V, E_a, p), as we have defined in chapter 2. This set function was shown in [11, 40, 47] to be monotone and submodular. Then by Proposition 7 and Proposition 8, we have that the EIP function $\mathbb{E}[\sigma(E_a, \phi_0)]$, which is positive linear combination of monotone and submodular functions, is monotone and submodular with respect to the seed set ϕ_0.

Second, when the seed set ϕ_0 is fixed, differently from the influence maximization problem based on node selection discussed in Chapter 4 [11, 40, 47], our EIP function $\mathbb{E}[\sigma(E_a, \phi_0)]$ with respect to the active links E_a is monotone but is not submodular.

The monotonicity of our EIP function $\mathbb{E}[\sigma(E_a, \phi_0)]$ with respect to the set of active links E_a is straightforward: adding an activated link will definitely not lead to a reduction of influence propagation, i.e, $\mathbb{E}[\sigma(E_a \cup \{(i,j)\}, \phi_0)] \geq \mathbb{E}[\sigma(E_a, \phi_0)]$. To show that it is not submodular, we build a counterexample as follows.

Example 5. Consider an Independent Cascade model, where $V = \{1, 2, 3, 4\}$, $E = \{(1, 3), (2, 3), (3, 4)\}$, $p_{1,3} = p_{2,3} = p_{3,4} = 0.5$, and $\mathbf{c} = \mathbf{1}$, shown in Figure 5.1. We set the cardinality of seed set $K = 2$, i.e., $\Phi_2 = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$, and let $S = \{(1, 3)\}$, $T = \{(1, 3), (3, 4)\}$, $u = \{(2, 3)\}$.

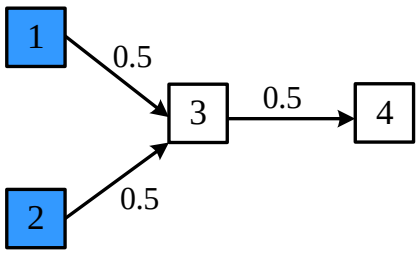

Fig. 5.1 An Independent Cascade model with 4 nodes

Then when fix seed set $\phi_0 = \{1, 2\}$ we have

$$\sigma(S, \phi_0) = p_{1,3} = 0.5,$$

$$\sigma(T, \phi_0) = p_{2,3} + p_{2,3} \cdot p_{3,4} = 0.75,$$

$$\sigma(S \cup \{u\}, \phi_0) = p_{1,3} + (1 - p_{1,3}) \cdot p_{2,3} = 0.75,$$

$$\sigma(T \cup \{u\}, \phi_0) = (p_{1,3} + (1 - p_{1,3}) \cdot p_{2,3})(1 + p_{3,4}) = 1.125,$$

$$\sigma(S \cup \{u\}, \phi_0) - \sigma(\phi_0, S) = 0.25,$$
$$\sigma(T \cup \{u\}, \phi_0) - \sigma(\phi_0, T) = 0.375.$$

i.e., when $\phi_0 = \{1, 2\}$,

$$\mathbb{E}[\sigma(\phi_0, S \cup \{u\})] - \mathbb{E}[\sigma(\phi_0, S)] < \mathbb{E}[\sigma(\phi_0, T \cup \{u\})] - \mathbb{E}[\sigma(\phi_0, T)].$$

Obviously, the EIP function $\mathbb{E}[\sigma(E_a, \phi_0)]$ for this model is not submodular with respect to the set of active links E_a.

Summarizing, for the problem of budgeted influence maximization by link activation, the following result holds.

Theorem 6. *The EIP function $\mathbb{E}[\sigma(E_a, \phi_0)]$ in the Independent Cascade model is monotone and submodular with respect to the seed set ϕ_0; monotone but not submodular with respect to the set of active links E_a.*

5.3.3 Hardness of Budgeted Influence Maximization by Link Activation

We now explore the complexity of the budgeted influence maximization by link activation problem. We will show it is NP-hard by reducing it to the well known *maximal covering*, which is known to be NP-hard.

Theorem 7. *The problem of budgeted influence maximization by link activation is NP-hard under the Independent Cascade model.*

Proof. Consider an instance of the NP-hard *Maximum Coverage* problem, defined by a constant integer k and a collection of sets $\mathcal{S} = \{S_1, S_2, ..., S_m\}$, where each set has a weight $w(S_i)$, $i = 1, 2, ..., m$. The objective is to find a subset $\mathcal{S}' \subseteq \mathcal{S}$ of sets, such that $|\mathcal{S}'| \leq k$ and the total weights of covered elements $\bigcup_{S_i \in \mathcal{S}'} w(S_i)$ is maximized.

Consider the set $S_E = \{S_{(i,j)} \mid (i, j) \in E\}$ and let the EIP obtained by an arbitrary link (i, j) be the weight of set $S_{(i,j)}$. Then the maximum coverage problem can be viewed as a

special case of our problem of the budgeted influence maximization by link activation when the cost $c_{i,j}$ is the same among all links. □

5.4 Cost-Degree Heuristics

Our goal is to maximize EIP on the condition that the total cost for link activation is no more than a given budget. Thus the edges with big propagation probability but small activation cost are considered to be activated firstly. The two approaches proposed in this section are based on the cost-degree coefficient, which takes both propagation probability and activation cost into account.

5.4.1 SimCD Algorithm

Denote $\Theta_{i,j}^{(1)} = \frac{p_{i,j}}{c_{i,j}}$ as the *simplex cost-degree* of link (i,j), then select the link (i,j) with the maximum value of $\Theta_{i,j}^{(1)}$ to be activated in each iteration. The iteration process stops when the total cost for link activation does not satisfy the budget constraint. The SimCD (Simplex Cost-Degree) algorithm shown in Algorithm 12 describes the detailed procedure.

Algorithm 12 SimCD

Input: An Independent Cascade network $G_{IC} = (V, E, p)$, a cost vector **c**, a budget $\mathcal{B}$;
Output: An active edge set E_a;

1: Initialize $E_a = \emptyset$;
2: Compute $\Theta_{i,j}^{(1)} = \frac{p_{i,j}}{c_{i,j}}$ for $\forall (i,j) \in E$;
3: **while** $\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}$ **do**
4: $\quad E_a = E_a \cup \{ \underset{(i,j)\in E\setminus E_a}{argmax} (\Theta_{i,j}^{(1)}) \}$;
5: **end while**
6: **return** E_a.

The time complexity of SimCD is $O(m)$ with $m = |E|$ (the number of edges) since SimCD computes $\Theta_{i,j}^{(1)}$ for each edge $(i,j) \in E$. $\Theta_{i,j}^{(1)}$ is easy to compute and this approach is time-efficient. However, SimCD may not perform well when in a network there is a scenario as follows: a link (i,j) is associated with a big simplex cost-degree $\Theta_{i,j}^{(1)}$, but the out-neighbors of node j are mostly associated with a small $\Theta_{i,j}^{(1)}$. In this case, the link (i,j)

is in fact not a good choice for deriving a large influence propagation under a limited budget. Nevertheless SimCD can not filter out the links described above since it ignores the impact of subsequent links.

5.4.2 MulCD Algorithm

In order to improve SimCD, we further propose the MulCD (Multiple Cost-Degree) algorithm based on the *multiple cost-degree*. The multiple cost-degree is computed by $\Theta^{(2)}_{i,j} = \frac{p_{i,j} \cdot \sum_{s \in N^{out}_j} p_{j,s}}{c_{i,j}}$, which considers the propagation probabilities of link (i,j) and also of its subsequent links. As with SimCD, the time complexity of MulCD shown in Algorithm 13 is $O(m)$ with $m = |E|$ (the number of edges).

Algorithm 13 MulCD

Input: An Independent Cascade network $G_{IC} = (V, E, p)$, a cost vector **c**, a budget $\mathcal{B}$;
Output: An active edge set E_a;

1: Initialize $E_a = \emptyset$;
2: Compute $\Theta^{(2)}_{i,j} = \frac{p_{i,j} \cdot \sum_{s \in N^{out}_j} p_{j,s}}{c_{i,j}}$ for $\forall (i,j) \in E$;
3: **while** $\sum_{(i,j) \in E_a} c_{i,j} \leq \mathcal{B}$ **do**
4: $\quad E_a = E_a \cup \{ \underset{(i,j) \in E \setminus E_a}{argmax} (\Theta^{(2)}_{i,j}) \}$;
5: **end while**
6: **return** E_a.

We give an example in Example 6 to describe the selection procedure of the set of active links by these two cost-degree heuristics.

Example 6. Figure 5.2 shows an Independent Cascade model, and we assume: $c_{1,3} = 10$, $c_{2,3} = 9$, $c_{3,4} = 4$, $c_{4,3} = 8$, $c_{4,5} = 10$, $c_{4,6} = 5$, $c_{4,7} = 9$, the cardinality of seed set $K = 2$ and the budget $\mathcal{B} = 25$.

The simplex cost-degree of each link is:

$$\Theta^{(1)}_{1,3} = \frac{p_{1,3}}{c_{1,3}} = 0.040, \quad \Theta^{(1)}_{2,3} = \frac{p_{2,3}}{c_{2,3}} = 0.033, \quad \Theta^{(1)}_{4,3} = \frac{p_{4,3}}{c_{4,3}} = 0.025,$$
$$\Theta^{(1)}_{4,5} = \frac{p_{4,5}}{c_{4,5}} = 0.020, \quad \Theta^{(1)}_{4,6} = \frac{p_{4,6}}{c_{4,6}} = 0.080, \quad \Theta^{(1)}_{4,7} = \frac{p_{4,7}}{c_{4,7}} = 0.067,$$
$$\Theta^{(1)}_{3,4} = \frac{p_{3,4}}{c_{3,4}} = 0.100.$$

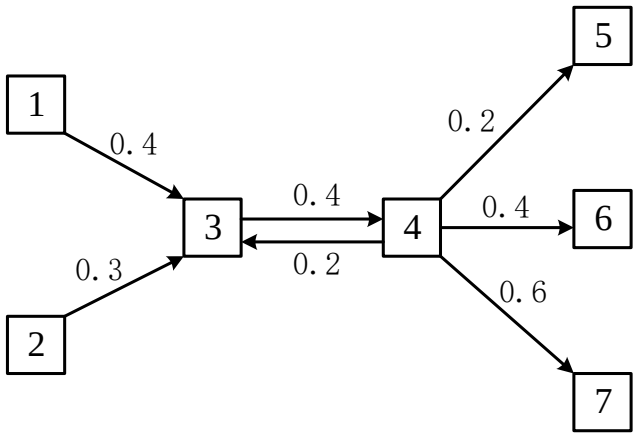

Fig. 5.2 An Independent Cascade model with 7 nodes

Limited by the budget $\mathcal{B} = 25$, the set of active links selected by SimCD is $E_a = \{(3,4),(4,6),(4,7)\}$ and the EIP $\mathbb{E}[\sigma(E_a,\phi_0)] = 0.410$.

The multiple cost-degree of each link is:

$$\Theta^{(2)}_{1,3} = \frac{p_{1,3}\cdot p_{3,4}}{c_{1,3}} = 0.016, \qquad \Theta^{(2)}_{2,3} = \frac{p_{2,3}\cdot p_{3,4}}{c_{2,3}} = 0.013,$$
$$\Theta^{(2)}_{4,3} = \frac{p_{4,3}\cdot p_{3,4}}{c_{4,3}} = 0.010, \qquad \Theta^{(2)}_{4,5} = \frac{p_{4,5}}{c_{4,5}} = 0,$$
$$\Theta^{(2)}_{4,6} = \frac{p_{4,6}}{c_{4,6}} = 0, \qquad \Theta^{(2)}_{4,7} = \frac{p_{4,7}}{c_{4,7}} = 0,$$
$$\Theta^{(2)}_{3,4} = \frac{p_{3,4}\cdot(p_{4,3}+p_{4,5}+p_{4,6}+p_{4,7})}{c_{3,4}} = 0.120.$$

Limited by the budget $\mathcal{B} = 25$, the set of active links selected by MulCD is $E_a = \{(3,4),(1,3),(2,3)\}$ and the EIP $\mathbb{E}[\sigma(E_a,\phi_0)] = 0.307$.

Evidently, these two cost-degree approaches select activated link (i,j) based on the values of propagation probability $p_{i,j}$ and activation cost $c_{i,j}$. Nevertheless in a network there can be some links associated with a big cost-degree value but have few out-neighbors, or are not connected with seed set. These links are not good selections for achieving a large influence propagation but may be selected by SimCD or MulCD. Thus besides the activation cost and the propagation probability, the seed set and the influence propagation of E_a in each iteration are also important parameters for selection of active links. In the

next part, we propose two approaches based on the inf-degree coefficient, which consider both the influence propagation of real-time E_a and the activation cost during the selection of active links.

5.5 Inf-Degree Heuristics

In this section, we propose two approaches based on the inf-degree coefficient for solving this budgeted influence maximization problem.

5.5.1 SimID Algorithm

The SimID (Simplex Inf-Degree) algorithm is based on the *simplex inf-degree* $\zeta_{i,j}^{(1)}$ which involves the influence propagation of E_a at current selection step and the activation cost. In the SimID shown in Algorithm 14, we compute $\zeta_{i,j}^{(1)} = \frac{\mathbb{E}[\sigma(E_a \cup \{(i,j)\})] - \mathbb{E}[\sigma(E_a)]}{c_{i,j}}$ for each link $(i,j) \in E \setminus E_a$ and choose the link (i,j) with the maximum value of $\zeta_{i,j}^{(1)}$ to be activated in each iteration. Let Φ_K denote the set of all subsets of V of cardinality K and assume $\Phi_K = \{\phi_{0,1}, \phi_{0,2}, ..., \phi_{0,r}, ...\}$, $\Delta_r = \sigma(E_a \cup \{(i,j)\}, \phi_{0,r}) - \sigma(E_a, \phi_{0,r})$, then we have

$$\begin{aligned}
\zeta_{i,j}^{(1)} &= \frac{\mathbb{E}[\sigma(E_a \cup \{(i,j)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]}{c_{i,j}} \\
&= \frac{\frac{1}{|\Phi_K|} \cdot [\sum_{\phi_0 \in \Phi_K} \sigma(E_a \cup \{(i,j)\}, \phi_0) - \sum_{\phi_0 \in \Phi_K} \sigma(E_a, \phi_0)]}{c_{i,j}} \\
&= \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_{0,r} \in \Phi_K} \Delta_r}{c_{i,j}}.
\end{aligned}$$

The iteration process stops when the total cost for link activation does not satisfy the budget constraint. Let T be the time required to compute the influence propagation $\sigma(\cdot)$, then the time complexity of SimID is $O(m^2|\Phi_K|T)$ where $m = |E|$ and $|\Phi_K| = \frac{N!}{K!(N-K)!}$ ($N = |V|$).

Unfortunately, the greedy procedure of SimID, which selects a link maximizing the

Algorithm 14 SimID

Input: An Independent Cascade network $G_{IC} = (V, E, p)$, a cost vector $\mathbf{c}$, a budget $\mathcal{B}$, a seed set size K;

Output: An active edge set E_a;

1: Let Φ_K denote the set of all subsets of V of given cardinality K;
2: Initialize $E_a = \emptyset$;
3: **while** $\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}$ **do**
4: Compute $\zeta_{i,j}^{(1)} = \frac{\mathbb{E}[\sigma(E_a \cup \{(i,j)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]}{c_{i,j}}$ for $\forall (i,j) \in E \setminus E_a$;
5: $E_a = E_a \cup \{\underset{(i,j)\in E\setminus E_a}{argmax}(\zeta_{i,j}^{(1)})\}$;
6: **end while**
7: **return** E_a.

simplex inf-degree $\zeta_{i,j}^{(1)}$ at each iteration, has unbounded approximation factor. We explain it in Example 7. However, we observe that if the EIP of E_a selected by SimID is no less than the incremental change of EIP caused by addition of any unselected link, SimID can achieve an approximation ratio of $\frac{1}{2}(1 - \frac{1}{e})$.

Example 7. We consider the network in Figure 5.3 with $V = \{i, j_1, j_2\}$, $E = \{(i, j_1), (i, j_2)\}$, $p_{i,j_1} = 0.1$, $p_{i,j_2} = p$. Let $c_{i,j_1} = 0.1$, $c_{i,j_2} = p + 1$, $\mathcal{B} = p + 1$ and $K = 1$. Then we have

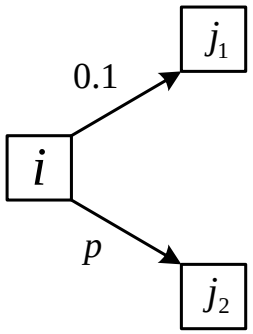

Fig. 5.3 An Independent Cascade model with 3 nodes

$$\mathbb{E}[\sigma(\{(i,j_1)\}, \phi_0)] = \frac{1}{3} \cdot [\sum_{\phi_0 \in \{i,j_1,j_2\}} \sigma(\{(i,j_1)\}, \phi_0) = \frac{1}{30},$$
$$\mathbb{E}[\sigma(\{(i,j_2)\}, \phi_0)] = \frac{1}{3} \cdot [\sum_{\phi_0 \in \{i,j_1,j_2\}} \sigma(\{(i,j_2)\}, \phi_0) = \frac{p}{3},$$
$$\zeta_{i,j_1}^{(1)} = \frac{\mathbb{E}[\sigma(\{(i,j_1)\}, \phi_0)]}{c_{i,j_1}} = \frac{1}{3},$$
$$\zeta_{i,j_2}^{(1)} = \frac{\mathbb{E}[\sigma(\{(i,j_2)\}, \phi_0)]}{c_{i,j_2}} = \frac{p}{3(p+1)}.$$

Thus the solution obtained by the SimID is $E_a = \{(i, j_1)\}$, while the optimal solution in fact depends on the value of p.

In order to analyse the performance guarantee of SimID under a certain constraint, first we propose two lemmas based on the theorems presented by Khuller et al. [98].

Given an Independent Cascade model $G_{IC} = (V, E, p)$, a cost vector $\mathbf{c} \in V \times V$, and a constant integer K such that the cardinality of seed set is $|\phi_0| = K$, there must be an optimal set of links E_a^* maximizing the EIP $\mathbb{E}[\sigma(E_a, \phi_0)]$ while the total cost $\sum_{(i,j)\in E_a} c_{i,j}$ does not exceed a given budget $\mathcal{B}$. Assuming that the SimID is executed for l times before the first link belonging to E_a^* is considered but not added to E_a, obviously the number of links added to E_a during these iterations equals to l. We denote the k-th link added to E_a, $k = 1, 2, ..., l$, as $(i, j)^k$, the first considered link from E_a^* as $(i, j)^{l+1}$ and the cost of the k-th added link as $c_{i,j}^k$, $k = 1, 2, ..., l + 1$. Then the union of selected links in each iteration E_a^k is represented as $E_a^k = \bigcup_{s=1}^{k}(i, j)^s$, $k = 1, 2, ..., l + 1$.

Lemma 1. Given an Independent Cascade model $G_{IC} = (V, E, p)$, a constant integer K such that the cardinality of seed set $|\phi_0| = K$ and a cost vector $\mathbf{c} \in V \times V$, let $E_a^* = \underset{\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}}{argmax} \mathbb{E}[\sigma(E_a, \phi_0)]$ be the optimal set of links maximizing the EIP within a budget $\mathcal{B}$, $E_a^k = \bigcup_{s=1}^{k}(i, j)^s$ be the union of selected links in iteration $k = 1, 2, ..., l + 1$, and $c_{i,j}^k$ be the cost for adding the k-th link $(i, j)^k$. For $k = 1, 2, ..., l + 1$, we have

$$\mathbb{E}[\sigma(E_a^k, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] \geq \frac{c_{i,j}^k}{\mathcal{B}}(\mathbb{E}[\sigma(E_a^*, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)]).$$

Proof. For $\forall(i, j) \in E_a^* \setminus E_a^{k-1}$, we have $\zeta_{i,j}^{(1)} \leq \zeta_{i,j}^{(1)k}$, where $\zeta_{i,j}^{(1)k}$ is the inf-degree of link $(i, j)^k$, since the link $(i, j)^k$ maximizes the inf-degree over all links that have not been selected before iteration k. The total cost of the links in $E_a^* \setminus E_a^{k-1}$ is limited by a budget $\mathcal{B}$, thus we have

$$\mathbb{E}[\sigma(E_a^{k-1} \cup (E_a^* \setminus E_a^{k-1}), \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] \leq \mathcal{B} \cdot \zeta_{i,j}^{(1)k},$$

i.e.,

$$\mathbb{E}[\sigma(E_a^*, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] \leq \mathcal{B} \cdot \zeta_{i,j}^{(1)k}. \tag{5-3}$$

According to the definition of $\zeta_{i,j}^{(1)k}$, we have

$$\zeta_{i,j}^{(1)k} = \frac{\mathbb{E}[\sigma(E_a^k, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)]}{c_{i,j}^k}. \tag{5-4}$$

Substitute Equation 5-4 for $\zeta_{i,j}^{(1)k}$ in Inequality 5-3, then the lemma holds. □

Lemma 2. Given an Independent Cascade model $G_{IC} = (V, E, p)$, a cost vector $\mathbf{c} \in V \times V$ and a constant integer K such that the cardinality of seed set $|\phi_0| = K$, let $E_a^* = \underset{\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}}{argmax} \mathbb{E}[\sigma(E_a, \phi_0)]$ be the optimal set of links maximizing the EIP within a budget $\mathcal{B}$, $E_a^k = \bigcup_{s=1}^{k} (i,j)^s$ be the union of selected links in iteration $k = 1, 2, ..., l+1$, and $c_{i,j}^k$ be the cost for adding the k-th link $(i,j)^k$. For $k = 1, 2, ..., l+1$, we have

$$\mathbb{E}[\sigma(E_a^k, \phi_0)] \geq [1 - \prod_{s=1}^{k}(1 - \frac{c_{i,j}^s}{\mathcal{B}})] \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

Proof. The proof is by induction on k.

For $k = 1$, we have

$$\mathbb{E}[\sigma(E_a^{k=1}, \phi_0)] = \mathbb{E}[\sigma(\{(i,j)^{k=1}\}, \phi_0)] = c_{i,j}^{k=1} \cdot \zeta_{i,j}^{(1)k=1}.$$

From Lemma 1, we obtain

$$\mathbb{E}[\sigma(E_a^{k=1}, \phi_0)] \geq \frac{c_{i,j}^{k=1}}{\mathcal{B}} \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)]. \tag{5-5}$$

Suppose that the statement holds for iterations $1, 2, ..., k-1$, then based on Lemma 1 for iteration k, we write

$$\begin{aligned}
\mathbb{E}[\sigma(E_a^k, \phi_0)] &= \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] + (\mathbb{E}[\sigma(E_a^k, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)]) \\
&\geq \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] + \frac{c_{i,j}^k}{\mathcal{B}} \cdot (\mathbb{E}[\sigma(E_a^k, \phi_0)] - \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)]) \\
&= (1 - \frac{c_{i,j}^k}{\mathcal{B}}) \cdot \mathbb{E}[\sigma(E_a^{k-1}, \phi_0)] + \frac{c_{i,j}^k}{\mathcal{B}} \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)]
\end{aligned}$$

$$
\begin{aligned}
&\geq (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot[(1-\frac{c_{i,j}^{k-1}}{\mathcal{B}})\cdot\mathbb{E}[\sigma(E_a^{k-2},\phi_0)]+1-(1-\frac{c_{i,j}^{k-1}}{\mathcal{B}})\cdot \\
&\quad \mathbb{E}[\sigma(E_a^{*},\phi_0)]]+\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\sigma(E_a^{*}) \\
&\geq (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot\{(1-\frac{c_{i,j}^{k-1}}{\mathcal{B}})\cdot[(1-\frac{c_{i,j}^{k-2}}{\mathcal{B}})\mathbb{E}[\sigma(E_a^{k-3},\phi_0)]+\frac{c_{i,j}^{k-2}}{\mathcal{B}}\cdot \\
&\quad \mathbb{E}[\sigma(E_a^{*},\phi_0)]]+\frac{c_{i,j}^{k-1}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)]\}+\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)] \\
&= (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot[\prod_{s=k-2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}})\cdot\mathbb{E}[\sigma(E_a^{k-3},\phi_0)]+(1-\prod_{s=k-2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}}))\cdot \\
&\quad \mathbb{E}[\sigma(E_a^{*},\phi_0)]]+\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)] \\
&\geq \ldots \\
&\geq (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot[\prod_{s=2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}})\cdot\mathbb{E}[\sigma(E_a^{k=1},\phi_0)]+(1-\prod_{s=2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}}))\cdot \\
&\quad \mathbb{E}[\sigma(E_a^{*},\phi_0)]]+\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)]. \qquad (5\text{-}6)
\end{aligned}
$$

Plugging inequality 5-5 into inequality 5-6, we have

$$
\begin{aligned}
\mathbb{E}[\sigma(E_a^{k},\phi_0)] &\geq (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot(\prod_{s=2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}})\cdot\frac{c_{i,j}^{k=1}}{\mathcal{B}}+1-\prod_{s=2}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}}))\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)] \\
&\quad +\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)] \\
&= (1-\frac{c_{i,j}^{k}}{\mathcal{B}})\cdot(1-\prod_{s=1}^{k-1}(1-\frac{c_{i,j}^{s}}{\mathcal{B}}))\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)]+\frac{c_{i,j}^{k}}{\mathcal{B}}\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)] \\
&= [1-\prod_{s=1}^{k}(1-\frac{c_{i,j}^{s}}{\mathcal{B}})]\cdot\mathbb{E}[\sigma(E_a^{*},\phi_0)].
\end{aligned}
$$

The lemma follows. □

One remark is recalled for the latter analysis of approximation guarantee [98].

Remark 3. For λ_1, λ_2, ..., $\lambda_n \in \mathbb{R}^+$ such that $\sum_{i=1}^{n}\lambda_i = \Lambda$, the function $(1-\prod_{i=1}^{n}(1-\frac{\lambda_i}{\Lambda}))$ achieves the minimum value when $\lambda_1 = \lambda_2 = ... = \lambda_n = \frac{\Lambda}{n}$.

Based on the lemmas and remark above, the theorem for the approximation guarantee of the SimID follows.

Theorem 8. *Given an Independent Cascade model $G_{IC} = (V, E, p)$, a cost vector $\boldsymbol{c} \in V \times V$ and a constant integer K such that the cardinality of seed set is $|\phi_0| = K$, let $E_a^* = \underset{\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}}{argmax} \mathbb{E}[\sigma(E_a, \phi_0)]$ be the optimal set of links maximizing the EIP within a budget $\mathcal{B}$. Compute the set of active links $E_a \subseteq E$ by Algorithm 14 with the same budget and let $(u, v) \in E \setminus E_a$ denote any of the unselected links. If the EIP of E_a is no less than the incremental change of EIP caused by addition of any unselected link, i.e., $\mathbb{E}[\sigma(E_a, \phi_0)] \geq \mathbb{E}[\sigma(E_a \cup \{(u, v)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]$ for $\forall (u, v) \in E \setminus E_a$, we have*

$$\mathbb{E}[\sigma(E_a, \phi_0)] \geq \frac{1}{2}(1 - \frac{1}{e}) \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

Proof. From Lemma 2 we obtain

$$\mathbb{E}[\sigma(E_a^{l+1}, \phi_0)] \geq [1 - \prod_{s=1}^{l+1}(1 - \frac{c_{i,j}^s}{\mathcal{B}})] \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

Since adding $(i, j)^{l+1}$ to E_a^l violates the budget $\mathcal{B}$, i.e., $\sum_{s=1}^{l+1} c_{i,j}^s = \sum_{s=1}^{l} c_{i,j}^s + c_{i,j}^{l+1} > \mathcal{B}$, we have

$$\mathbb{E}[\sigma(E_a^{l+1}, \phi_0)] \geq [1 - \prod_{s=1}^{l+1}(1 - \frac{c_{i,j}^s}{\sum_{s=1}^{l+1} c_{i,j}^s})] \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

According to Remark 3, when $c_{i,j}^s = \frac{\sum_{s=1}^{l+1} c_{i,j}^s}{l+1}$, $s = 1, 2, ..., l+1$, $(1 - \prod_{s=1}^{l+1}(1 - \frac{c_{i,j}^s}{\sum_{s=1}^{l+1} c_{i,j}^s}))$ achieves its minimum value $(1 - (1 - \frac{1}{l+1})^{l+1})$. Then since $\lim_{l\to\infty}(1 - \frac{1}{l+1})^{l+1} = \frac{1}{e}$, we have

$$\mathbb{E}[\sigma(E_a^{l+1}, \phi_0)] \geq (1 - \frac{1}{e}) \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)],$$

i.e.,

$$\mathbb{E}[\sigma(E_a^l, \phi_0)] + \mathbb{E}[\sigma(E_a^{l+1}, \phi_0)] - \mathbb{E}[\sigma(E_a^l, \phi_0)]) \geq (1 - \frac{1}{e}) \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

From the assumption of theorem, we get $\mathbb{E}[\sigma(E_a^l, \phi_0)] \geq \mathbb{E}[\sigma(E_a^{l+1}, \phi_0)] - \mathbb{E}[\sigma(E_a^l, \phi_0)]$.

Thus we have

$$\mathbb{E}[\sigma(E_a^l, \phi_0)] \geq \frac{1}{2}(1 - \frac{1}{e}) \cdot \mathbb{E}[\sigma(E_a^*, \phi_0)].$$

The theorem follows. □

Whether the set of active links E_a selected by SimID can achieve the EIP approximating the optimal one within a factor of $\frac{1}{2}(1 - \frac{1}{e})$ depends on whether E_a satisfies the sufficient condition of Theorem 8. To verify that, after the selection of E_a, we still compute $\mathbb{E}[\sigma(E_a \cup \{(u,v)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]$ for unselected links $(u,v) \in E \backslash E_a$. If $\mathbb{E}[\sigma(E_a, \phi_0)] \geq \max_{(u,v) \in E \backslash E_a} (\mathbb{E}[\sigma(E_a \cup \{(u,v)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)])$, then the sufficient condition holds and the selected set of active links E_a can achieve an approximation guarantee of $\frac{1}{2}(1 - \frac{1}{e})$. Usually the sufficient condition holds for most real-world networks as long as the budget $\mathcal{B}$ is not too small, which means that adequate links have been added to E_a and one more addition will not lead to a big incremental change of the EIP.

5.5.2 MulID Algorithm

The SimID considers the activation cost of the currently investigative link, and it can achieve a constant approximation guarantee under a certain constraint. However, it is more reasonable to consider the total activation cost of all links in the set E_a. In the MulID (Multiple Inf-Degree) algorithm shown in Algorithm 15, we denote the *multiple inf-degree* of link (i,j) as $\zeta_{i,j}^{(2)} = \frac{\mathbb{E}[\sigma(E_a \cup \{(i,j)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]}{\sum_{(u,v) \in E_a \cup \{(i,j)\}} c_{u,v}}$.

Similar to SimID, let T be the time required to compute the influence propagation $\sigma(\cdot)$, then the time complexity of MulID is $O(m^2 |\Phi_K| T)$ where $m = |E|$, $|\Phi_K| = \frac{N!}{K!(N-K)!}$ ($N = |V|$).

Example 8. Consider the Independent Cascade model in Figure 5.2, and the values of the activation cost **c**, the cardinality of seed set K and the budget $\mathcal{B}$ are set as the same as Example 6.

Algorithm 15 MulID

Input: An Independent Cascade network $G_{IC} = (V, E, p)$, a cost vector $\mathbf{c}$, a budget $\mathcal{B}$, a seed set size K;

Output: An active edge set E_a;

1: Let Φ_K denote the set of all subsets of V of given cardinality K;
2: Initialize $E_a = \emptyset$;
3: **while** $\sum_{(i,j)\in E_a} c_{i,j} \leq \mathcal{B}$ **do**
4: Compute $\zeta_{i,j}^{(2)} = \frac{\mathbb{E}[\sigma(E_a \cup \{(i,j)\}, \phi_0)] - \mathbb{E}[\sigma(E_a, \phi_0)]}{\sum_{(u,v)\in E_a \cup \{(i,j)\}} c_{u,v}}$ for $\forall (i,j) \in E \setminus E_a$;
5: $E_a = E_a \cup \{\underset{(i,j)\in E\setminus E_a}{argmax}(\zeta_{i,j}^{(2)})\}$;
6: **end while**
7: **return** E_a.

Applying SimID, at step $t = 1$, we have:

$$\zeta_{1,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(1,3)\}, \phi_0)}{c_{1,3}} = 0.0095, \quad \zeta_{2,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(2,3)\}, \phi_0)}{c_{2,3}} = 0.0079,$$

$$\zeta_{4,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(4,3)\}, \phi_0)}{c_{4,3}} = 0.0060, \quad \zeta_{4,5}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(4,5)\}, \phi_0)}{c_{4,5}} = 0.0048,$$

$$\zeta_{4,6}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(4,6)\}, \phi_0)}{c_{4,6}} = 0.0190, \quad \zeta_{4,7}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(4,7)\}, \phi_0)}{c_{4,7}} = 0.0159,$$

$$\zeta_{3,4}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(3,4)\}, \phi_0)}{c_{3,4}} = 0.0238.$$

After step $t = 1$, $E_a = \{(3,4)\}$, $\mathbb{E}[\sigma(\{(3,4)\}, \phi_0)] = 0.0952$ and $c_{3,4} = 4$.

At step $t = 2$, we have:

$$\zeta_{1,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(1,3)\cup(3,4)\}, \phi_0) - \mathbb{E}[\sigma(\{3,4\}, \phi_0)]}{c_{1,3}} = 0.0126,$$

$$\zeta_{2,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(2,3)\cup(3,4)\}, \phi_0) - \mathbb{E}[\sigma(\{3,4\}, \phi_0)]}{c_{2,3}} = 0.0105,$$

$$\zeta_{4,3}^{(1)} = \frac{\frac{1}{|\Phi_K|} \cdot \sum_{\phi_0\in\Phi_K} \sigma(\{(4,3)\cup(3,4)\}, \phi_0) - \mathbb{E}[\sigma(\{3,4\}, \phi_0)]}{c_{4,3}} = 0.0060,$$

$$\zeta_{4,5}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,5)\cup(3,4)\},\phi_0)-\mathbb{E}[\sigma(\{3,4\},\phi_0)]}{c_{4,5}} = 0.0063,$$
$$\zeta_{4,6}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,6)\cup(3,4)\},\phi_0)-\mathbb{E}[\sigma(\{3,4\},\phi_0)]}{c_{4,6}} = 0.0252,$$
$$\zeta_{4,7}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,7)\cup(3,4)\},\phi_0)-\mathbb{E}[\sigma(\{3,4\},\phi_0)]}{c_{4,7}} = 0.0210.$$

After step $t=2$, $E_a=\{(3,4),(4,6)\}$, $\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]=0.2210$, $c_{3,4}+c_{4,6}=9$.

At step $t=3$, we have:

$$\zeta_{1,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(1,3)\cup(3,4)\cup(4,6)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]}{c_{1,3}} = 0.0135,$$
$$\zeta_{2,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(2,3)\cup(3,4)\cup(4,6)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]}{c_{2,3}} = 0.0133,$$
$$\zeta_{4,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,3)\cup(3,4)\cup(4,6)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]}{c_{4,3}} = 0.0059,$$
$$\zeta_{4,5}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,5)\cup(3,4)\cup(4,6)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]}{c_{4,5}} = 0.0063,$$
$$\zeta_{4,7}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(4,7)\cup(3,4)\cup(4,6)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\},\phi_0)]}{c_{4,7}} = 0.0210.$$

After step $t=3$, $E_a=\{(3,4),(4,6),(4,7)\}$, $\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\cup(4,7)\},\phi_0)] = 0.4100$ and $c_{3,4}+c_{4,6}+c_{4,7}=18$.

At step $t=4$, we have:

$$\zeta_{1,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(1,3)\cup(3,4)\cup(4,6)\cup(4,7)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\cup(4,7)\},\phi_0)]}{c_{1,3}}$$
$$=0.0110,$$
$$\zeta_{2,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum\limits_{\phi_0\in\Phi_K}\sigma(\{(2,3)\cup(3,4)\cup(4,6)\cup(4,7)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\cup(4,7)\},\phi_0)]}{c_{2,3}}$$

$$=0.0013,$$

$$\zeta_{4,3}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum_{\phi_0\in\Phi_K}\sigma(\{(4,3)\cup(3,4)\cup(4,6)\cup(4,7)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\cup(4,7)\},\phi_0)]}{c_{4,3}}$$

$$=0,$$

$$\zeta_{4,5}^{(1)} = \frac{\frac{1}{|\Phi_K|}\cdot\sum_{\phi_0\in\Phi_K}\sigma(\{(4,5)\cup(3,4)\cup(4,6)\cup(4,7)\},\phi_0)-\mathbb{E}[\sigma(\{(3,4)\cup(4,6)\cup(4,7)\},\phi_0)]}{c_{4,5}}$$

$$=0.0062.$$

Since $\underset{(i,j)\in E\setminus E_a}{argmax}(\zeta_{i,j}^{(1)}) = (1,3)$ and $c_{3,4}+c_{4,6}+c_{4,7}+c_{1,3} = 28 > \mathcal{B}$ for step $t = 4$, the set of active links selected by SimID is $E_a = \{(3,4),(4,6),(4,7)\}$ and $\mathbb{E}[\sigma(E_a,\phi_0)] = 0.4100$.

Now we verify that the solution of SimID provides the EIP approximating the optimal one within $\frac{1}{2}(1-\frac{1}{e})$ for this example.

Since

$$\mathbb{E}[\sigma(E_a\cup\{1,3\},\phi_0) = 0.52, \qquad \mathbb{E}[\sigma(E_a\cup\{2,3\},\phi_0) = 0.42,$$

$$\mathbb{E}[\sigma(E_a\cup\{4,3\},\phi_0) = 0.41, \qquad \mathbb{E}[\sigma(E_a\cup\{4,5\},\phi_0) = 0.47,$$

we have $\mathbb{E}[\sigma(E_a,\phi_0)] \geq \mathbb{E}[\sigma(E_a\cup\{(u,v)\},\phi_0)] - \mathbb{E}[\sigma(E_a,\phi_0)]$ for $\forall(u,v) \in E\setminus E_a$. According to Theorem 8, $\mathbb{E}[\sigma(E_a,\phi_0)] \geq \frac{1}{2}(1-\frac{1}{e})\cdot\mathbb{E}[\sigma(E_a^*,\phi_0)]$ holds for this example.

Similarly, when applying MulID, at step $t = 1$, we have $\zeta_{i,j}^{(2)} = \zeta_{i,j}^{(1)}$ for $(i,j) \in E$. Thus $E_a = \{3,4\}$, $\mathbb{E}[\sigma(\{(3,4)\},\phi_0)] = 0.0952$ and $c_{3,4} = 4$ after step $t = 1$.

At step $t = 2$, we have

$$\zeta_{1,3}^{(2)} = 0.0090, \quad \zeta_{2,3}^{(2)} = 0.0073, \quad \zeta_{4,3}^{(2)} = 0.0040, \quad \zeta_{4,5}^{(2)} = 0.0045, \quad \zeta_{4,6}^{(2)} = 0.0140, \quad \zeta_{4,7}^{(2)} = 0.0145.$$

After step $t=2$, $E_a=\{(3,4),(4,7)\}$, $\mathbb{E}[\sigma(\{(3,4)\cup(4,7)\},\phi_0)] = 0.2837$ and $c_{3,4}+c_{4,7}=13$.

At step $t = 3$, we have

$$\zeta_{1,3}^{(2)} = 0.0061, \quad \zeta_{2,3}^{(2)} = 0.0048, \quad \zeta_{4,3}^{(2)} = 0.0023, \quad \zeta_{4,5}^{(2)} = 0.0027, \quad \zeta_{4,6}^{(2)} = 0.0070.$$

After step $t = 3$, $E_a = \{(3,4),(4,7),(4,6)\}$, $\mathbb{E}[\sigma(\{(3,4) \cup (4,7) \cup (1,3)\}, \phi_0)] = 0.4100$ and $c_{3,4} + c_{4,7} + c_{1,3} = 18$.

At step $t = 4$, we have

$$\zeta_{1,3}^{(2)} = 0.0053, \qquad \zeta_{2,3}^{(2)} = 0.0041, \qquad \zeta_{4,3}^{(2)} = 0.0018, \qquad \zeta_{4,5}^{(2)} = 0.0022.$$

Since $\underset{(i,j)\in E\setminus E_a}{argmax}(\zeta_{i,j}^{(2)}) = (1,3)$ and $c_{3,4}+c_{4,7}+c_{4,6}+c_{1,3} = 28 > \mathcal{B}$ for step $t = 4$, the set of active links selected by MulID is $E_a = \{(3,4),(4,7),(4,6)\}$ and $\mathbb{E}[\sigma(E_a, \phi_0)] = 0.4100$.

We list the set of active links E_a and EIP value obtained by SimCD, MulCD, SimID and MulID for the network in Figure 5.2 in Table 5.1. SimID selects the same set of active links with SimCD and MulID, and the solution of SimID provides a EIP value approximating the optimal one within $\frac{1}{2}(1 - \frac{1}{e})$.

TABLE 5.1 Comparison of E_a and EIP computed by SimCD, MulCD, SimID and MulID for the network in Figure 5.2

Heuristics	E_a	$\mathbb{E}[\sigma(E_a, \phi_0)]$
SimCD	{(3,4),(4,6),(4,7)}	0.4100
MulCD	{(3,4),(1,3),(2,3)}	0.3072
SimID	{(3,4),(4,6),(4,7)}	0.4100
MulID	{(3,4),(4,7),(4,6)}	0.4100

Summarized from Example 6 and Example 8, the simplex cost-degree and multiple cost-degree are unique for each link once the network and the activation cost are set. Differently, the simplex inf-degree and multiple inf-degree change in different iteration for the selection of active links.

5.6 Experimental Evaluation

We perform a series of experiments on real-world datasets to evaluate the heuristics proposed above in the aspect of maximizing the EIP by activating a set of links.

All approaches are implemented in Python. All experiments are run on a PC with 2.40GHz Intel Core i5 Processor and 8GB memory.

5.6.1 Data Set

We consider two real-world datasets. The first one, called **Advise-Seek**, is from a study by Robins on one branch of an Australian bank [99]. It presents the advise-seek relation among a staff of 11 people, i.e, a directed edge from i to j denotes that i asks advice from j if an issue arises in work. There are 11 nodes and 30 edges in the Advise-Seek network. The second dataset, called **HighTec**, was collected from the managers of a high-technical company in the United States [100]. It presents the report relation among 21 managers, i.e., a directed edge from i to j denotes that i reports to j. There are 21 nodes and 20 edges in this network.

5.6.2 Experimental Setup

The two considered networks are both directed and binary. To construct from them generally Independent Cascade models, we generate a value of propagation probability $p_{i,j}$ uniformly at random from the interval $[0.1, 0.5)$ for each edge (i, j). The activation cost $c_{i,j}$ for activating the edge (i, j) is uniformly selected at random from the interval $[1, 10)$. The budget for link activation $\mathcal{B}$ varies in the set $\{10, 20, 30, 40, 50\}$. The cardinality of seed set K is selected from $\{1, 3, 5\}$.

We evaluate the performance of SimCD, MulCD, SimID and MulID in terms of influence propagation and running time for link selection. For SimCD and MulCD, we select the set of active links according to the value of either simplex cost-degree or multiple cost-degree. To validate the solution, the influence propagation of each possible seed set

is computed by *SteadyStateSpread*. Then EIP is obtained by averaging these values of influence propagation. For SimID and MulID, the computation of EIP is also based on the influence propagation computation by *SteadyStateSpread*.

5.6.3 Experimental Results

The EIPs computed by SimCD, MulCD, SimID and MulID for a given cardinality of seed set $K = \{1, 3, 5\}$ within budget $\mathcal{B} = \{10, 20, 30, 40, 50\}$ on networks Advise-Seek and HighTec are shown in Figure 5.4 and Figure 5.5. In the network Advise-Seek, we observe that SimID outperforms the other heuristics in most cases, except when $\mathcal{B} = 20$, $K = 3$ and $\mathcal{B} = 20$, $K = 5$, MulID provides larger EIPs than SimID. SimCD and MulCD perform similarly for all given values of K and budget $\mathcal{B}$. In the network HighTec, when $K = 1$ and $\mathcal{B} = 30$, link $(20, 6)$ is selected by SimID and the activation of it contributes a large value of EIP. When $K = 3$ and $K = 5$, SimID and MulID almost perform the same, SimCD and MulCD almost perform the same. On the whole SimID can select the set of active links to achieve a larger value of EIP than the other heuristics in most cases.

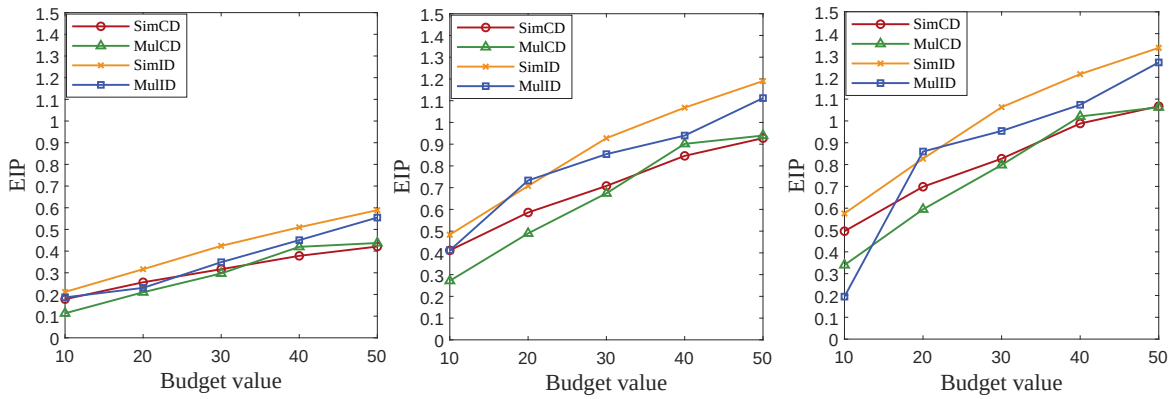

Fig. 5.4 EIP computed by SimCD, MulCD, SimID and MulID on Advise-Seek network data given $K = \{1, 3, 5\}$ within budget $\mathcal{B} = \{10, 20, 30, 40, 50\}$

Note that the running time recorded is the time for selecting active links by different heuristics, not including the time for EIP computation. The link selection by SimCD or MulCD does not depend on the seed set and it only takes very short time in both networks Advise-Seek and High-Tec. The complexity of SimID and MulID is tightly related with the cardinality of seed set K, thus SimID and MulID take much longer time when K increases.

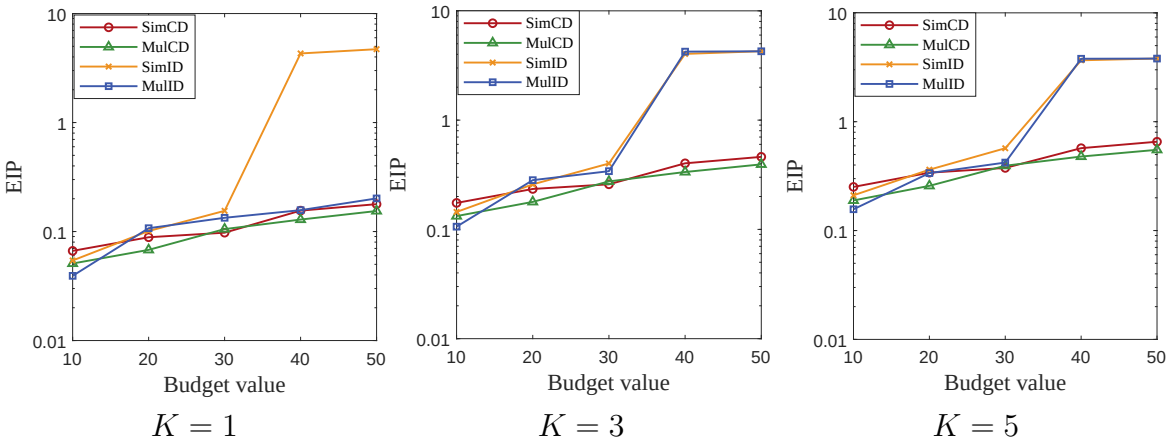

Fig. 5.5 EIP computed by SimCD, MulCD, SimID and MulID on HighTec network data given $K = \{1, 3, 5\}$ within budget $\mathcal{B} = \{10, 20, 30, 40, 50\}$

Besides, a larger budget makes it possible to iterate more to select more active links. Thus the running time of SimID and MulID also increases as the budget increases, nevertheless the growth of the computation time of SimID and MulID still depends more on cardinality of seed set K than budget $\mathcal{B}$.

TABLE 5.2 Running time for link selection by SimCD, MulCD, SimID and MulID on Advise-Seek network data given $K = \{1, 3, 5\}$ within budget $\mathcal{B} = \{10, 20, 30, 40, 50\}$

Seed set size	$K = 1$					$K = 3$					$K = 5$				
Running time(s) \ Budget / Approach	10	20	30	40	50	10	20	30	40	50	10	20	30	40	50
SimCD	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
MulCD	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
SimID	0.9	1.3	1.6	2.0	2.2	12.4	18.0	21.9	28.1	30.9	38.1	54.2	76.3	70.1	76.1
MulID	0.7	0.9	1.2	1.7	2.3	11.2	22.2	19.0	26.6	27.2	28.1	50.0	50.1	65.1	62.3

TABLE 5.3 Running time for link selection by SimCD, MulCD, SimID and MulID on HighTec network data given $K = \{1, 3, 5\}$ within budget $\mathcal{B} = \{10, 20, 30, 40, 50\}$

Seed set size	$K = 1$					$K = 3$					$K = 5$				
Running time(s) \ Budget / Approach	10	20	30	40	50	10	20	30	40	50	10	20	30	40	50
SimCD	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
MulCD	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
SimID	1.2	1.2	2.0	4.1	6.8	52.4	80.0	123.6	193.9	295.9	797.4	1386.6	2006.3	2646.0	3892.6
MulID	0.6	1.2	1.4	2.1	2.0	41.4	80.2	92.3	179.1	240.7	671.0	1029.3	3612.3	3217.9	4175.3

Finally, an experimental study on random networks of different size is carried out to test the scalability of the four algorithms. The networks are generated by Software Gephi with the number of nodes $|V|$ from 10 to 40 where the rewiring probability is selected as 0.05. We set the budget $\mathcal{B}$ to 20 and the cardinality of seed set K to 3. The result is shown

in Figure 5.6, which suggests: SimCD and MulCD are very efficient and the running time of them does not change a lot with the network size; the time for selecting the set of active links by SimID or MulID increases exponentially when the scale of network is large.

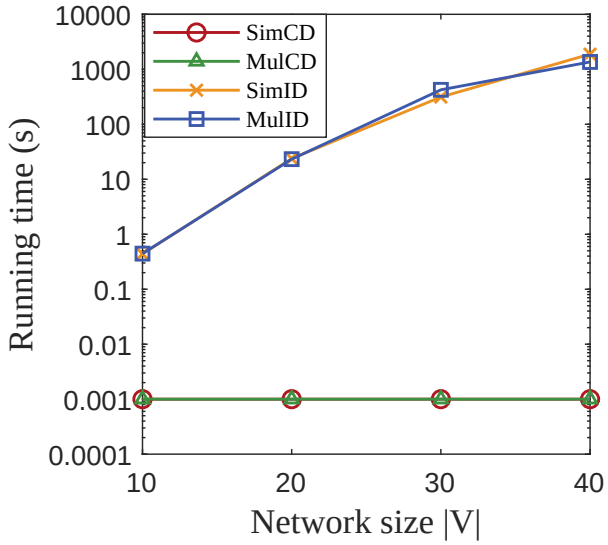

Fig. 5.6 Scalability of the algorithms

Summarily, while Simplex Inf-Degree (SimID) algorithm seems to be the heuristic providing the best EIP, it has a high computational complexity, in terms of execution time. For this reason, as the size of network increases it may be necessary to use different heuristics such as Simplex Cost-Degree (SimCD) algorithm and Multiple Cost-Degree (MulCD) algorithm whose computation time does not grow with the number of nodes $|V|$ and the cardinality of seed set K.

5.7 Conclusion and Future Work

Most previous works focus on the approaches to either influence maximization considering the initial adopters or influence minimization based on link blocking. Differently in this chapter, we formulate an influence maximization problem within a limited budget considering to activate the relatively most effective links. For approximately solving this problem, we propose two types of heuristics associated with either a cost-degree coefficient or an inf-degree coefficient. Emphatically, it is proved that the SimID algorithm can select a

set of active links which under a certain constraint achieves an approximation guarantee of $\frac{1}{2}(1-\frac{1}{e})$.

There are several future directions for this work. First, the propagation probability and cost vector for our current diffusion model are fixed and unrelated. We plan to generalize our model in terms of associating these two parameters together in order to better describe the real-world scenarios. Second, our inf-degree heuristics (SimID and MulID algorithms) consider the real-time innovation diffusion process and thus it is time-consuming to select the set of active links. We aim to improve the efficiency of inf-degree heuristics in the future.

Chapter 6 Conclusion and Future Work

We conclude the thesis by summarizing our main results and discuss possible directions for future research.

6.1 Main Results of the Thesis

Influence maximization in social networks is the central theme of the thesis. Among models for simulating the influence propagation process, we choose the Independent Cascade model to describe and solve our problems. We are mainly interested in the three problems under the Independent Cascade model: the influence propagation estimation from an initial set of adopters; methodology for selecting a suboptimal set of initial adopters to maximize the final influence propagation; formulation and solution to the budgeted influence maximization problem by link activation, i.e., maximizing the expected influence propagation by activating a set of links within a certain budget.

Influence propagation computation is a preliminary step for further network analysis. We propose *Path Method* to compute the exact value of influence propagation for small networks. Two elements leading to the inaccuracy of *SteadyStateSpread* are pointed: dependency relation among nodes and existence of circuits. Then improved algorithms called *SSS-Bounded-Path* and *SSS-Noself* are further proposed to partially decrease the error caused by circuits.

As for influence maximization by seed selection, we evaluate the approaches to influence propagation computation: *SteadyStateSpread* and *SSS-Noself* together with one selection strategy among *SelectTopK*, *RankedReplace* and greedy algorithm.

We initially formulate the problem of influence maximization by link activation under a certain budget. We prove that this problem is a NP-hard and point out that the expectation of influence propagation is monotone and submodular in terms of seed set and monotone but

non-submodular in terms of active links. Heuristics based on a cost-degree coefficient or an inf-degree coefficient are proposed to provide suboptimal solution. We prove that the SimID can select a set of activated links which achieves an approximation guarantee of $\frac{1}{2}(1-\frac{1}{e})$ under a certain constraint.

6.2 Future Work

Specific extended directions for, respectively, influence propagation computation, influence maximization by seed selection and budgeted influence maximization by link activation have been given in Chapter 3, Chapter 4 and Chapter 5. In this section, we discuss three directions of further research addressing the problem of influence maximization in social networks.

- The first is about the extension of classic diffusion models. Most studies about the influence maximization problem are done under either the Linear Threshold model or the Independent Cascade model. Recently, some diffusion models are developed by adding various parameters such like negative opinion, time label and competitive innovations. We will consider extending our solutions for influence maximization problem to these newly developed diffusion models.

- The second is about the efficiency and scalability of the methodologies for maximizing influence propagation. Some of our approaches are not viable for large networks currently. Thus it is important to develop more scalable algorithms to handle large datasets.

- The third is about the model parameter learning. Our current studies assume that the network structure and the model parameter, such as influence weight for Linear Threshold model and activation probability for Independent Cascade model, are known and fixed. The point we focus on is the performance of algorithms in a fixed network. However, it is more practical to learn these parameters from real-world dataset. We try to analyse methodologies about influence propagation in the networks constructed according to real dataset.

References

[1] W. Chen, L. V. Lakshmanan, and C. Castillo, "Information and influence propagation in social networks," *Synthesis Lectures on Data Management*, vol. 5, no. 4, pp. 1–177, 2013.

[2] W. Lu, "Computational social influence: models, algorithms, and applications," Ph.D. dissertation, University of British Columbia, 2016.

[3] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the 3rd ACM international conference on Web search and data mining*. ACM, 2010, pp. 241–250.

[4] R. Durrett, *Lecture notes on particle systems and percolation*. Brooks/Cole Pub Co, 1988.

[5] T. M. Liggett, *Interacting particle systems*. Springer Science & Business Media, 2012, vol. 276.

[6] M. Granovetter, "Threshold models of collective behavior," *American journal of sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.

[7] T. C. Schelling, *Micromotives and macrobehavior*. WW Norton & Company, 2006.

[8] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing letters*, vol. 12, no. 3, pp. 211–223, 2001.

[9] J. Goldenberg, B. Libai, and E. Muller, "Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata," *Academy of Marketing Science Review*, vol. 9, no. 3, pp. 1–18, 2001.

[10] R. M. Anderson, B. Anderson, and R. M. May, *Infectious diseases of humans: dynamics and control*. Oxford university press, 1991.

[11] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 137–146.

[12] D. Centola and M. Macy, "Complex contagions and the weakness of long ties," *American journal of Sociology*, vol. 113, no. 3, pp. 702–734, 2007.

[13] R. F. Bales, "Interaction process analysis; a method for the study of small groups." 1950.

[14] J. Berger, S. J. Rosenholtz, and M. Zelditch Jr, "Status organizing processes," *Annual review of sociology*, vol. 6, no. 1, pp. 479–508, 1980.

[15] J. T. Cacioppo, R. E. Pett, and C. D. Stoltenberg, "Processes of social influence: The elaboration likelihood model of persuasion." 1985.

[16] M. Deutsch and H. B. Gerard, "A study of normative and informational social influences upon

individual judgment." *Journal of abnormal and social psychology*, vol. 51, no. 3, p. 629, 1955.

[17] J. French, B. Raven, and D. Cartwright, "The bases of social power," *Classics of organization theory*, vol. 7, pp. 311–320, 1959.

[18] B. Latané, "Dynamic social impact: The creation of culture by communication," *Journal of communication*, vol. 46, no. 4, pp. 13–25, 1996.

[19] G. Ritzer, *The Blackwell encyclopedia of sociology*. Blackwell Publishing New York, NY, USA, 2007, vol. 1479.

[20] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 57–66.

[21] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[22] R. Pastor-Satorras and A. Vespignani, "Epidemic spreading in scale-free networks," *Physical review letters*, vol. 86, no. 14, p. 3200, 2001.

[23] S. Bhagat, A. Goyal, and L. V. Lakshmanan, "Maximizing product adoption in social networks," in *Proceedings of the 5th ACM international conference on Web search and data mining*. ACM, 2012, pp. 603–612.

[24] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 665–674.

[25] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, and Y. Yuan, "Influence maximization in social networks when negative opinions may emerge and propagate," in *Proceedings of the 2011 siam international conference on data mining*. SIAM, 2011, pp. 379–390.

[26] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model," in *Proceedings of the 2012 siam international conference on data mining*. SIAM, 2012, pp. 463–474.

[27] G. Tong, W. Wu, L. Guo, D. Li, C. Liu, B. Liu, and D.-Z. Du, "An efficient randomized algorithm for rumor blocking in online social networks," *IEEE Transactions on Network Science and Engineering*, 2017.

[28] W. Lu, W. Chen, and L. V. Lakshmanan, "From competition to complementarity: comparative influence diffusion and maximization," *Proceedings of the VLDB Endowment*, vol. 9, no. 2, pp. 60–71, 2015.

[29] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks,"

in *International workshop on web and internet economics*. Springer, 2007, pp. 306–311.

[30] A. Borodin, Y. Filmus, and J. Oren, "Threshold models for competitive influence in social networks," in *International workshop on internet and network economics*. Springer, 2010, pp. 539–550.

[31] A. Borodin, M. Braverman, B. Lucier, and J. Oren, "Strategyproof mechanisms for competitive influence in networks," *Algorithmica*, vol. 78, no. 2, pp. 425–452, 2017.

[32] T. Carnes, C. Nagarajan, S. M. Wild, and A. Van Zuylen, "Maximizing influence in a competitive social network: a follower's perspective," in *Proceedings of the 9th international conference on Electronic commerce*. ACM, 2007, pp. 351–360.

[33] W. Lu, F. Bonchi, A. Goyal, and L. V. Lakshmanan, "The bang for the buck: fair competitive viral marketing from the host perspective," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 928–936.

[34] S. A. Myers and J. Leskovec, "Clash of the contagions: Cooperation and competition in information diffusion," in *2012 IEEE 12th international conference on data mining*. IEEE, 2012, pp. 539–548.

[35] R. Narayanam and A. A. Nanavati, "Viral marketing for product cross-sell through social networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2012, pp. 581–596.

[36] A. Zarezade, A. Khodadadi, M. Farajtabar, H. R. Rabiee, and H. Zha, "Correlated cascades: Compete or cooperate," in *31st AAAI Conference on Artificial Intelligence*, 2017.

[37] B. Liu, G. Cong, D. Xu, and Y. Zeng, "Time constrained influence maximization in social networks," in *2012 IEEE 12th international conference on data mining*. IEEE, 2012, pp. 439–448.

[38] W. Chen, W. Lu, and N. Zhang, "Time-critical influence maximization in social networks with time-delayed diffusion process," in *26th AAAI Conference on Artificial Intelligence*, 2012.

[39] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1029–1038.

[40] D. Kempe, J. Kleinberg, and É. Tardos, "Influential nodes in a diffusion model for social networks," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2005, pp. 1127–1138.

[41] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 420–429.

[42] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++: optimizing the greedy algorithm for influence

maximization in social networks," in *Proceedings of the 20th international conference companion on World wide web*. ACM, 2011, pp. 47–48.

[43] C. C. Aggarwal, A. Khan, and X. Yan, "On flow authority discovery in social networks," in *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM, 2011, pp. 522–533.

[44] Y. Yang, E. Chen, Q. Liu, B. Xiang, T. Xu, and S. A. Shad, "On approximation of real-world influence spread," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2012, pp. 548–564.

[45] P. Domingos, "Mining social networks for viral marketing," *IEEE Intelligent Systems*, vol. 20, no. 1, pp. 80–82, 2005.

[46] C. Zhou, P. Zhang, J. Guo, X. Zhu, and L. Guo, "Ublf: An upper bound based approach to discover influential nodes in social networks," in *Data Mining (ICDM), IEEE 13th International Conference on*. IEEE, 2013, pp. 907–916.

[47] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 199–208.

[48] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the 25th annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 946–957.

[49] J. Tang, X. Tang, and J. Yuan, "Influence maximization meets efficiency and effectiveness: A hop-based approach," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2017, pp. 64–71.

[50] J. Tang, X. Tang, and J. Yuan, "An efficient and effective hop-based approach for influence maximization in social networks," *Social Network Analysis and Mining*, vol. 8, no. 1, p. 10, 2018.

[51] E. A. Enns and M. L. Brandeau, "Link removal for the control of stochastically evolving epidemics over networks: A comparison of approaches," *Journal of theoretical biology*, vol. 371, pp. 154–165, 2015.

[52] J. He, H. Liang, and H. Yuan, "Controlling infection by blocking nodes and links simultaneously," in *International workshop on internet and network economics*. Springer, 2011, pp. 206–217.

[53] M. Kimura, K. Saito, and H. Motoda, "Blocking links to minimize contamination spread in a social network," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 3, no. 2, p. 9, 2009.

[54] J. Marcelino and M. Kaiser, "Critical paths in a metapopulation model of h1n1: Efficiently delaying influenza spreading through flight cancellation," *PLoS currents*, vol. 4, 2012.

[55] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos, "Gelling, and melting,

large graphs by edge manipulation," in *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012, pp. 245–254.

[56] C. J. Kuhlman, G. Tuli, S. Swarup, M. V. Marathe, and S. Ravi, "Blocking simple and complex contagion by edge removal," in *2013 IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 399–408.

[57] A. K. Nandi and H. R. Medal, "Methods for removing links in a network to minimize the spread of infections," *Computers & Operations Research*, vol. 69, pp. 10–24, 2016.

[58] M. Kimura, K. Saito, and H. Motoda, "Minimizing the spread of contamination by blocking links in a network." in *AAAI*, vol. 8, 2008, pp. 1175–1180.

[59] B. Ryan and N. C. Gross, "The diffusion of hybrid seed corn in two iowa communities." *Rural sociology*, vol. 8, no. 1, p. 15, 1943.

[60] J. S. Coleman, E. Katz, H. Menzel *et al.*, *Medical innovation: A diffusion study*. Bobbs-Merrill Indianapolis, 1966.

[61] J. Wortman, "Viral marketing and the diffusion of trends on social networks," 2008.

[62] D. Rosa, "Graph methods in multi agent systems coordination and social network analysis," Ph.D. dissertation, Universita'degli Studi di Cagliari, 2014.

[63] D. Kempe, J. M. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network." *Theory of Computing*, vol. 11, no. 4, pp. 105–147, 2015.

[64] W. Yang, L. Brenner, and A. Giua, "Computation of activation probabilities in the independent cascade model," in *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2018, pp. 791–797.

[65] J. Bus, "Convergence of newton-like methods for solving systems of nonlinear equations," *Numerische Mathematik*, vol. 27, no. 3, pp. 271–281, 1976.

[66] C. Grosan and A. Abraham, "Multiple solutions for a system of nonlinear equations," *International Journal of Innovative Computing, Information and Control*, vol. 4, no. 9, pp. 2161–2170, 2008.

[67] S. Kakutani *et al.*, "A generalization of brouwer's fixed point theorem," *Duke mathematical journal*, vol. 8, no. 3, pp. 457–459, 1941.

[68] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions-i," *Mathematical programming*, vol. 14, no. 1, pp. 265–294, 1978.

[69] E. Schechter, *Handbook of Analysis and its Foundations*. Academic Press, 1996.

[70] M. Kimura and K. Saito, "Tractable models for information diffusion in social networks," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2006, pp. 259–271.

[71] V. Colizza, R. Pastor-Satorras, and A. Vespignani, "Reaction–diffusion processes and

metapopulation models in heterogeneous networks," *Nature Physics*, vol. 3, no. 4, p. 276, 2007.

[72] X. Deng, Y. Dou, T. Lv, and Q. V. H. Nguyen, "A novel centrality cascading based edge parameter evaluation method for robust influence maximization," *IEEE Access*, vol. 5, pp. 22 119–22 131, 2017.

[73] H. Wu, J. Shang, S. Zhou, Y. Feng, B. Qiang, and W. Xie, "Laim: A linear time iterative approach for efficient influence maximization in large-scale networks," *IEEE Access*, vol. 6, pp. 44 221–44 234, 2018.

[74] F. Ye, J. Liu, C. Chen, G. Ling, Z. Zheng, and Y. Zhou, "Identifying influential individuals on large-scale social networks: A community based approach," *IEEE Access*, vol. 6, pp. 47 240–47 257, 2018.

[75] Q. Yu, H. Li, Y. Liao, and S. Cui, "Fast budgeted influence maximization over multi-action event logs," *IEEE Access*, vol. 6, pp. 14 367–14 378, 2018.

[76] S. Cheng, H. Shen, J. Huang, G. Zhang, and X. Cheng, "Staticgreedy: solving the scalability-accuracy dilemma in influence maximization," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 2013, pp. 509–518.

[77] Y. Zhu, W. Wu, Y. Bi, L. Wu, Y. Jiang, and W. Xu, "Better approximation algorithms for influence maximization in online social networks," *Journal of Combinatorial Optimization*, vol. 30, no. 1, pp. 97–108, 2015.

[78] E. Cohen, D. Delling, T. Pajor, and R. F. Werneck, "Sketch-based influence maximization and computation: Scaling up with guarantees," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 629–638.

[79] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "A billion-scale approximation algorithm for maximizing benefit in viral marketing," *IEEE/ACM Transactions on Networking (TON)*, vol. 25, no. 4, pp. 2419–2429, 2017.

[80] C. Wang, W. Chen, and Y. Wang, "Scalable influence maximization for independent cascade model in large-scale social networks," *Data Mining and Knowledge Discovery*, vol. 25, no. 3, pp. 545–576, 2012.

[81] K. Jung, W. Heo, and W. Chen, "Irie: Scalable and robust influence maximization in social networks," in *2012 IEEE 12th International Conference on Data Mining*. IEEE, 2012, pp. 918–923.

[82] S. Galhotra, A. Arora, and S. Roy, "Holistic influence maximization: Combining scalability and efficiency with opinion-aware models," in *Proceedings of the 2016 International Conference on Management of Data*. ACM, 2016, pp. 743–758.

[83] G. Cordasco, L. Gargano, and A. A. Rescigno, "Influence propagation over large scale social networks," in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2015, pp. 1531–1538.

[84] G. Cordasco, L. Gargano, M. Mecchia, A. A. Rescigno, and U. Vaccaro, "A fast and effective heuristic for discovering small target sets in social networks," in *Combinatorial Optimization and Applications*. Springer, 2015, pp. 193–208.

[85] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1039–1048.

[86] Y. Chen, S. Chang, C. Chou, W. Peng, and S. Lee, "Exploring community structures for influence maximization in social networks," in *the 6th SNA-KDD Workshop on Social Network Mining and Analysis Held in Conjunction with KDD*, vol. 12, 2012, pp. 1–6.

[87] Y. C. Chen, W. Y. Zhu, W. C. Peng, W. C. Lee, and S. Y. Lee, "Cim: Community-based influence maximization in social networks," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 2, p. 25, 2014.

[88] J. Shang, S. Zhou, X. Li, L. Liu, and H. Wu, "Cofim: A community-based framework for influence maximization on large-scale networks," *Knowledge-Based Systems*, vol. 117, pp. 88–100, 2017.

[89] X. Li, X. Cheng, S. Su, and C. Sun, "Community-based seeds selection algorithm for location aware influence maximization," *Neurocomputing*, vol. 275, pp. 1601–1613, 2018.

[90] J. S. Coleman *et al.*, "Introduction to mathematical sociology." *Introduction to mathematical sociology.*, 1964.

[91] 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.

[92] S. Han, F. Zhuang, Q. He, and Z. Shi, "Balanced seed selection for budgeted influence maximization in social networks," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2014, pp. 65–77.

[93] E. Güney, "On the optimal solution of budgeted influence maximization problem in social networks," *Operational Research*, pp. 1–15, 2017.

[94] S. Banerjee, M. Jenamani, and D. K. Pratihar, "Combim: A community-based solution approach for the budgeted influence maximization problem," *Expert Systems with Applications*, vol. 125, pp. 1–13, 2019.

[95] M. Kimura, K. Saito, and H. Motoda, "Solving the contamination minimization problem on networks for the linear threshold model," in *Pacific Rim International Conference on Artificial Intelligence*. Springer, 2008, pp. 977–984.

[96] G. Fasshauer, "Math 590: Meshfree methods," *Department of Applied Mathematics, Illinois Institute of Technology*, 2010.

[97] S. Fujishige, *Submodular functions and optimization*. Elsevier, 2005, vol. 58.

[98] S. Khuller, A. Moss, and J. S. Naor, "The budgeted maximum coverage problem," *Information processing letters*, vol. 70, no. 1, pp. 39–45, 1999.

[99] P. Pattison, S. Wasserman, G. Robins, and A. M. Kanfer, "Statistical evaluation of algebraic constraints for social networks," *Journal of mathematical psychology*, vol. 44, no. 4, pp. 536–568, 2000.

[100] D. Krackhardt, "Cognitive social structures," *Social networks*, vol. 9, no. 2, pp. 109–134, 1987.

[101] 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.

[102] B. Xiang, Q. Liu, E. Chen, H. Xiong, Y. Zheng, and Y. Yang, "Pagerank with priors: an influence propagation perspective." in *IJCAI*. Citeseer, 2013, pp. 2740–2746.

[103] L. G. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, 1979.

[104] C. Hadjicostis, "Personal communication," 2016.

[105] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[106] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1, pp. 107–117, 1998.

[107] A. Giua, "Mathematical models for the diffusion of innovation in social networks," in *Notes for the course "Control of Network Systems", Polytech Marseille, Aix-Marseille University*.

[108] J. Tang, J. Sun, C. Wang, and Z. Yang, "Social influence analysis in large-scale networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 807–816.

[109] J. Tang, S. Wu, B. Gao, and Y. Wan, "Topic-level social network search," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 769–772.

Acknowledgements

I wish to express my deepest gratitude to my supervisors Prof. Alessandro GIUA and Dr. Leonardo BRENNER. This dissertation would not have been possible without their supervision, continuous guidance and encouragement. Throughout the years I have learned so much from them, and their unparalleled passion for research will always motivate me to pursue excellence.

I sincerely thank my supervisory committee members Prof. Dimitri LEFEBVRE, Prof. Mariagazia DOTOLI and Prof. Paolo FRASCA for their generous time and valuable feedback on the dissertation. I am very thankful for their questions and comments.

I am grateful to Prof. Zhiwu LI for being a great mentor in my Master and Ph.D. studies. His sincere suggestions play a significant role at some key points of my life.

I gratefully acknowledge the funding sources that made my academic work possible. I was funded by Archimede LabEx during my Master 2 study and by Chinese Scholarship Council during my doctoral study.

The acknowledgements would be barely complete without thanking all of my friends for making this journey joyful. Special thanks also go to all the colleagues in LIS, specially the members of MoFED group.

Most importantly, I am greatly indebted to my mother, my families and my boyfriend. Their unconditional love and support means everything to me. This dissertation is devoted to them.