Doctoral Thesis

Information Spreading with Network Augmentation

PhD Program in Computer Science: XXIX cycle

Author:
Yllka Velaj

Supervisors:
Prof. Pierluigi Crescenzi
Dr. Gianlorenzo D'Angelo

GSSI GRAN SASSO SCIENCE INSTITUTE
SCHOOL OF ADVANCED STUDIES
Scuola Universitaria Superiore

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

Declaration of Authorship

I, Yllka Velaj, declare that this thesis, titled “Information Spreading with Network Augmentation”, and the work presented in it are my own under the guidance of my supervisors Prof. Pierluigi Crescenzi and Dr. Gianlorenzo D’Angelo. I confirm that:

- Chapter 4 and Chapter 5 are based on papers "Recommending links through influence maximization" [27], submitted to *Theoretical Computer Science Journal*, and "Influence maximization in the independent cascade model" [25].
- Chapter 6 reports results published in "Selecting nodes and buying links to maximize the influence diffusion in a network" [28, 29].
- The experimental results in Chapter 4 and Chapter 5 are novel contributions, not appeared or submitted elsewhere.

Other results obtained during my PhD are published in [24, 23, 26, 8] but they are not included in this thesis since they focus on different topics.
Signed:

Date:

Abstract

When a new idea or innovation arises in a network of individuals, it can either quickly propagate to a large part of the network and be adopted by many individuals or immediately expire. Understanding the dynamics that regulate these behaviours has been one of the main goals in the field of complex network analysis and has been studied under the name of influence spreading or information diffusion problem. This problem is motivated by many applications in different fields from marketing to epidemiology, going through the study of adoption of innovations and the analysis of social networks.

Several studies have been conducted with the aim of shaping a given diffusion process so as to maximize or minimize the number of nodes that spread the information, called active nodes, at the end of the process by taking intervention actions. One of the most studied problems has been formalized by Kempe et al. [40] and consists in finding a set of initial nodes, called seeds, that maximizes the expected number of active nodes under a budget constraint. Beside it, other intervention actions may be used to facilitate or limit the diffusion processes such as inserting or deleting edges and adding or deleting nodes in the network.

In this work we present two possible intervention actions. First, we study the problem of maximizing the information spread in a network by creating a limited amount of new edges incident to a given initial set of active nodes. We show that the problem cannot be approximated within a constant factor greater than $1 - \frac{1}{2e}$, unless $P = NP$. We then propose an algorithm that guarantees an approximation factor of $1 - \frac{1}{e} - \epsilon$, where ϵ is any positive real number. We experimentally show that, with a small number of added edges our algorithm increases by far the expected number of active nodes with respect to an initial set of seeds. Moreover, our algorithm outperforms several other baselines. The experiments have been conducted both on artificial and real-world instances.

Then, we focus on a generalization of the problem of Kempe et al. in which we are allowed to spend the budget to buy seeds or new edges incident to the seeds according to a cost function. The problem does not admit a $PTAS$, unless $P = NP$. We propose two approximation algorithms: the former one gives an approximation ratio that depends on the edge costs and increases when such costs are high, the latter algorithm gives a constant approximation guarantee which is greater than that of the first algorithm when the edge costs can be small.

Acknowledgements

Firstly, I would like to express my gratitude to my advisor Prof. Pierluigi Crescenzi for providing me insightful research questions which stimulated me to widen my research from various perspectives.

My sincere thanks goes also to Dr. Gianlorenzo D'Angelo for the continuous and precious support, for his patience and encouragement. He is always available for any help or discussion. I could not have imagined having a better advisor for my PhD.

Besides my advisors, I would like to thank my thesis' reviewers: Prof. Francesco Bonchi and Prof. Christos Zaroliagis for their comments which helped me improve the quality of this thesis.

Secondly, I would like to thank all of my friends and my colleagues at GSSI, the best part of my PhD was the time spent with them. With a special mention to Lorenzo. It has been a lot of fun not only working with him, but also travelling together. I am glad I found a friend not only a coauthor.

Last but not the least, I would like to thank my family: my mother and my brother for supporting me throughout my PhD and in my life in general.

Contents

Declaration of Authorship **iii**

Abstract **v**

Acknowledgements **vii**

1 Introduction **1**
1.1 Main contributions 5
1.2 Thesis outline 7

2 Preliminaries **9**
2.1 Definitions 9
2.2 Influence diffusion models 12
2.2.1 Linear Threshold Model 12
2.2.2 Independent Cascade Model 13
2.3 Influence Maximization problem 14
2.3.1 Diffusion process approximation 16
2.4 Influence Maximization with link addition problems 17

3 A survey of related works **19**
3.1 Seed selection problem 19
3.2 Graph modification problem 21

4 Link addition problem **25**
4.1 Problem definition 25
4.2 Hardness result 26
4.3 Approximation algorithm 27
4.4 Improving the running time 31
4.5 Limited seed selection problem 33
4.6 Sketch-based algorithm 35
4.7 Experimental results 36

5 Link addition problem with costs **47**
5.1 Problem definition 47
5.2 Approximation algorithm 47
5.3 Experimental results 52

6 Budgeted seed selection and link addition problem **59**
6.1 Problem definition 59
6.2 Constant budget 60
6.3 Algorithm for lower bounded edge costs 61
6.4 Algorithm for general costs 68

7 Conclusion **77**

Bibliography **81**

Chapter 1

Introduction

In recent years, computers powered by internet have increased the connections between people and revolutionized the way they communicate with each other. The process of interacting and sharing knowledge or ideas leads to the development of *complex networks*.

The study of complex networks is a young and active area of research inspired by the empirical study of real-world networks such as computer networks and social networks.

By complex network we mean a directed graph. Each graph consists of nodes that represent users, companies and so on. The nodes are linked through connections which represent the relationships between them such as friendship, collaborations, interactions, etc. Examples of complex networks are: WWW, social networks as Facebook or Twitter, collaboration networks, financial and trade networks, neural networks, gene and protein interactions networks, metabolic networks, food webs, distribution networks, citation networks, and many others.

The word "complex" to refer to these networks is due to the non trivial topological features that often occur when modelling real case systems. Some of the typical attributes of many real-world complex networks [62] are described in the following.

They exhibit the *small-world effect*, namely, the distance between two randomly chosen nodes grows proportionally to the logarithm of the number of nodes in the network. This hypothesis was first described by Frigyes Karinthy and then tested experimentally by Stanley Milgram in 1967 [58, 72]. The main idea is that two arbitrary persons are connected by only six degrees of separation. The small-world effect has implication for the processes taking place in a network. If we consider the spreading of information it will be faster in the networks showing this phenomenon.

Complex networks have also a high clustering coefficient, usually larger than random graphs. The *clustering coefficient* represents the density of triangles in the network and it measures the degree to which the nodes tends to cluster together. In social networks, for example, this means that a friend of your friend is likely to be your friend too. Moreover, complex networks are *scale-free* [6] meaning that the probability that a node, selected uniformly at random, has a certain degree, follows a power low. This kind of degree distribution has been observed in citation networks, the WWW, the Internet, metabolic networks and so on.

Other attributes are *assortativity* and *disassortativity*. In assortative networks, well-connected nodes tend to join to other well-connected nodes, as in many social networks. In disassortative networks, by contrast, well-connected nodes join to a much larger number of less-well-connected nodes, this is typical of biological networks.

Analysis of complex networks is a very relevant field and continues to develop: it has been applied not only to social contexts, but also to analyse metabolic and genetic regulatory networks, to model and design scalable communication networks, to generate complex wireless networks, to develop vaccination strategies for the control of diseases and to a broad range of other practical issues.

The field has its origins in the work of psychologists, sociologists, economists, anthropologists and statisticians dating back to the late 1940's and early 1950's, but exploded in the late 1990's due to several breakthroughs.

The study of the structure and the dynamics in complex networks pervaded all of science from sociology to economics and computer science. In particular, the dynamic process by which information, ideas or influence propagates in a network has received a lot of attention in both academic and business contexts. This problem has been studied under the name of *influence spreading* or *information diffusion* [30, 40].

Nowadays, every decision an individual makes, whether it is watching a movie, reading a book or buying a new phone, is influenced by his or her personal network. As a result, understanding how a new idea or innovation propagates to a large part of the network and is adopted by many individuals, or why it expires immediately, has been one of the main goals in the field of complex networks analysis.

The motivating application span several fields: from marketing with the aim of evaluating the success of a new product or maximizing its adoption [7, 13, 33, 56, 66], to epidemiology in order to limit the diffusion of a virus or disease [57, 61], from the study of adoption of innovations [21, 67, 73], to the analysis of social networks to find influential users, from the study of how information flows through the network [5], to the analysis of cascading failures in power networks [3].

One of the fundamental problems in the study of influence spreading is the problem of *Influence Maximization* (IM). It has been proposed by Domingos and Richardson in the field of viral marketing and asks to find an initial set of users to be the early adopters of new technologies in order to activate a large cascade of further adoptions in the network [30, 66].

The problem has been formalized by Kempe et al. as follows: if we are allowed to choose at most k users, which ones should be selected so as to maximize the number of active nodes, i.e. those that spread the information, resulting from the diffusion process [41].

Different models of information diffusion process have been introduced in the literature. Two widely studied models are the *Linear Threshold Model* (LTM) [37, 41, 68] and the *Independent Cascade Model* (ICM) [33, 34, 40, 41]. In both models, we can distinguish between nodes which spread the information, called *active*, or *affected*, and *inactive* ones. At the beginning, in order to let the information diffusion process start,

a small percentage of nodes of the graph is set to active. These nodes are called *seeds*. Recursively, currently affected nodes can infect their neighbours with some probability. After a certain number of these cascading cycles, a large number of nodes might become affected in the network. Once a node is infected, it will remain active until the end of the process.

In LTM the idea is that a node becomes active as more of its neighbours become active. Formally, each node u has a threshold t_u chosen uniformly at random in the interval $[0, 1]$. The threshold represents the weighted fraction of neighbours of u that must be influenced in order for u to become active. During the process, a node u is activated if the total weight of its active neighbours is greater than t_u.

Also in ICM an active node u tries to influence its inactive neighbours but, differently from the previous model, the success of node u in activating node v only depends on the propagation probability of the edge from u to v (each edge has its own value). Regardless of its success, the same node will never get another chance to influence the same inactive neighbour. The process terminates when no further node gets activated.

An interesting question, in the analysis of the information diffusion through a network, is how to shape a given diffusion process so as to maximize or minimize the number of activated nodes at the end of the process by taking intervention actions. Besides the seeds selection proposed by Kempe et al. [40], other intervention actions may be used to facilitate the diffusion processes, such as inserting or deleting edges and adding or deleting nodes in the network.

Nowadays, billions of people are using Facebook and other social networks and those users are highly engaged with the content posted on the network: millions of people like something, or follow some posts, every single day.

These social networks are used also for advertising, but if the audience is too large it can actually be a drawback. Users only want to pay to expose their brand to people who might be interested in doing business with them. Our problem model this scenario: a user can spend his/her budget asking the social network to send advertisement to exactly the right audience for his/her business to maximize its profit and to provide him/her new connections.

Since in social networks, and in other complex networks, users can add edges incident to themselves, in this thesis we consider the possibility to create a limited number of new edges incident to the seed nodes.

In detail, we study three variations of the problem of Kempe et al. [41]: the *Influence Maximization with Link addition* problem (IM-L), the *Budgeted Influence Maximization with Link addition* problem (BIM-L) and the *Budgeted Influence Maximization with Seeds selection and Link addition* problem (BIM-SL).

In BIM-L, we consider the problem of recommending links to a given set of seeds in a social network, without exceeding a given budget, in such a way that the number of users reached by the contents generated by these seeds is maximized. In the IM-L problem, instead, we restrict to the case in which all edges to be added have unitary cost. Our main objective is to improve the capability of the given users to diffuse their

own contents, this in turn drives the network evolution towards an increment of the spreading capability of the whole network.

In the latter, we relax the hypothesis of knowing the initial set of seeds. In particular we want to find a set of seed nodes A and a set of edges S incident to the nodes in A such that the expected number of active nodes at the end of the diffusion process is maximized and the overall cost of A and S does not exceed a given budget k. We assume that all the seeds have the same cost, i.e. $\forall a \in A$, $c_a = 1$ and each edge e has cost $c_e \in [0, 1]$.

The problems we analyse differs from the problem proposed in the literature since we make the reasonable restriction that the edges to be added can only be incident to the seed nodes and that to add such edges there is a cost to be paid. To our knowledge, similar problems have never been studied for the Independent Cascade Model.

Besides the relevance of the problem itself as a generalization of the famous model of Kempe et al., we find motivations and applications of our problem in the link recommendation field.

In the previous works, influence was considered to flow over explicit social links only, such as in social networks like Facebook or Twitter. However, recently, recommender systems have emerged and they have become extremely popular and successful in a very short time. Some examples of recommender systems are Amazon, a product recommendation engine, Youtube, a video recommendation engine, and Netflix, a movie recommendation engine.

Link recommendation is one of the most important scientific issue in network analysis. Recommending suitable connections to social network users has a twofold impact: it improves the user's experience by enlarging users' social circles and personal network, and it increases social network revenue by enhancing the user involvement and retention rate.

Most of the existing link recommendation methods attempt to estimate the likelihood of the existence of a link between two users [4, 53, 54, 55]. In social networks the likelihood is estimated by considering the similarity of user profiles and some structural network properties.

The most used similarity metrics in friendship social networks, like Facebook, are based on the number of common neighbours. In general, similarity based algorithms assign a score s_{uv} for each pair of nodes u and v, which is directly defined as the similarity between these node. All new possible links are ranked according to their scores and the links connecting more similar nodes are supposed to be of higher existence likelihood. In spite of the simplicity of the idea, the definition of node similarity is a nontrivial challenge. Similarity index can be very simple or very complicated and it may work well for some networks while fails for someothers. For example, the Friend-of-Friend (FoF) algorithm [54] recommends the users that have the highest number of common friends with the receiver of the recommendation. Other examples of such similarity metrics are the Adamic-Adar [1] index, the Jaccard's coefficient [38], and the preferential attachment

index [10, 60]. In content-centric social networks such as Twitter and Google+ the link recommender systems take also into account the similarity of the users' interests.

In this thesis, we attack the link recommendation problem from a different point of view: we consider the impact of the new links on the capability of the network to spread information while most of the known methods aim at having a high accuracy in the prediction of the suggested links. In fact, this approach speeds up the network growth but is only able to infer links that will likely occur in the near future.

Another drawback of the existing methods is that, in most of the cases, they suggest links to a short range of users and this does not necessarily lead to a network growth and an improved user engagement [76].

On the other hand, the capability of a social network to spread information is directly correlated with both the engagement of a single user and the revenue of an online social network [12]. From a user's perspective, being able to quickly and effectively disseminating information is highly desirable as it helps the user to share contents so to reach a large number of other users. This in turn helps the user to build its own social reputation, to express and diffuse its own opinion, and to discover novel contents and information. From the social network point of view, the effectiveness of the information spreading capabilities helps in improving the user engagement, which increases the retention rate and the number of new subscriptions. Moreover, being able to quickly deliver diverse contents to a large portion of the users, increases the opportunities of making revenue from advertisement.

Most of the existing link recommendation systems overlook these aspects, which are crucial for both users and social networks. With this work we aim at overcoming the limitations of these link recommendation systems.

1.1 Main contributions

The problem addressed in this research is to maximize the diffusion of information in complex networks using intervention action such as seed selection and link addition.

For both problems, our goal is to study their computational complexity and to design algorithms to solve them approximately. In order to assess the performance of our algorithms, we evaluate them on both synthetic and real-world graphs. Moreover, our aim is to implement some heuristics in order to analyse large networks.

Based on the goal mentioned above, in this thesis we give the following contributions.

- We show that the IM-L problem is NP-hard to be approximated within a constant factor greater than $1 - \frac{1}{2e}$. We then provide an approximation algorithm that almost matches such upper bound by guaranteeing an approximation factor of $1 - \frac{1}{e} - \epsilon$, where ϵ is any positive real number. The algorithm is based on a greedy technique and the approximation factor is proven by showing that the expected number of activated nodes is monotonically increasing and submodular with respect to the possible set of edges incident to the seeds.

- We study the more general BIM-L problem where we are given a budget k and the cost of edges is in $[0, 1]$. We propose an algorithm that combines greedy and enumeration techniques and that achieves an approximation guarantee of $1-\frac{1}{e}-\epsilon$.
- We assess the quality of the solutions performing several experiments. We apply the greedy approach to real-world networks and we observe that by adding few edges the number of influenced nodes increases significantly.
- We compare our algorithms against many baselines, applied in the problems of link recommendation, and we show that ours outperform all of them. We run the experiments on both artificial and real-world networks.
- We prove that the BIM-SL problem cannot be approximated within a factor greater than $1-\frac{1}{e}$, unless $P = NP$. We then focus on approximation algorithms. We first assume that the edge costs are all greater that a given constant c_{min} and we propose two approximation algorithms that guarantee approximation factors of $1-\frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}$ and $\left(1-\frac{1}{e}\right)c_{\min}$, respectively. Then, we focus on the general case of our problem that consists in finding a good approximation even when the cost function includes small values. We propose an algorithm that guarantees a constant approximation ratio of about 0.0878.

The main results are summarized in Table 1.1.

Problem	**Approximation Upper bound**	**Approximation algorithms**
IM (Chapter 2)	$1-\frac{1}{e}$	$1-\frac{1}{e}-\epsilon$
IM-L (Chapter 4)	$1-\frac{1}{2e}$	$1-\frac{1}{e}-\epsilon$
BIM-L (Chapter 5)	$1-\frac{1}{2e}$	$1-\frac{1}{e}-\epsilon$
BIM-SL (Chapter 6)	$1-\frac{1}{e}$	$1-\frac{1}{e^{\frac{c_{min}}{1+c_{min}}}}$ $\left(1-\frac{1}{e}\right)c_{min}$ 0.0878

Table 1.1: Summary of the main results.

1.2 Thesis outline

The rest of this thesis is organized as follows. In Chapter 2 and Chapter 3, we introduce the main definitions related to influence maximization in complex networks, we provide an overview of the different models used to capture the dynamics of influence spreading in networks and we give a review of the literature in this area.

In Chapter 4 and Chapter 5 we focus on the IM-L and BIM-L problem for the Independent Cascade Model. We study the computational complexity of these problems. Moreover, we experimentally evaluate the quality of the solution and the performance of the greedy algorithm on both synthetic and real network.

In Chapter 6 we analyse the BIM-SL problem and we present our theoretical results. Finally, in Chapter 7, we conclude and present several future research directions.

Chapter 2

Preliminaries

In this chapter, we give the main definitions that will be used in the rest of this work and we describe the most widely studied models of information diffusion. Then, we formally define the three problems we focus on: *Influence Maximization with Link addition* problem (IM-L), *Budgeted Influence Maximization with Link addition* (BIM-L) problem and *Budgeted Influence Maximization with Seeds selection and Link addition* (BIM-SL) problem.

2.1 Definitions

Networks, and in particular social networks, are represented by graphs where nodes are users and edges are interactions or relationships between users.

In particular, we model a network as a weighted directed graph $G = (V, E, p, c)$, where V represents the set of nodes and E represents the set of relationships between nodes such as friendship, partnership, information exchange, etc.

Moreover, $p : V \times V \rightarrow [0, 1]$ is the propagation probability of an edge, that is the probability that the information is propagated from u to v. In other words, $\forall e = (u, v) \in V \times V$, p_e denotes the amount of influence exerted by u on v. We define also the cost function $c : V \times V \rightarrow [0, 1]$ for adding an edge to E.

In considering the model of influence spreading, we refer to each node as being either *active* or *inactive*. If a node is active (or adopter of the innovation), then it is already aware of the information under diffusion. If a node is inactive, then it is not informed or not influenced. The process runs in discrete steps. At the beginning of the process, few nodes known as *seed nodes* are given the information. Upon receiving the information these nodes become active. In each discrete step, an active node tries to influence its inactive neighbours. The success of node u in activating the node v depends on the propagation probability of the edge (u, v). The process terminates when no further nodes become activated from the inactive state.

We denote the set of seed nodes as A and we define the influence of a set $A \subseteq V$ in the graph G, denoted by $\sigma(A)$, to be the expected number of active nodes in G at the end of the process.

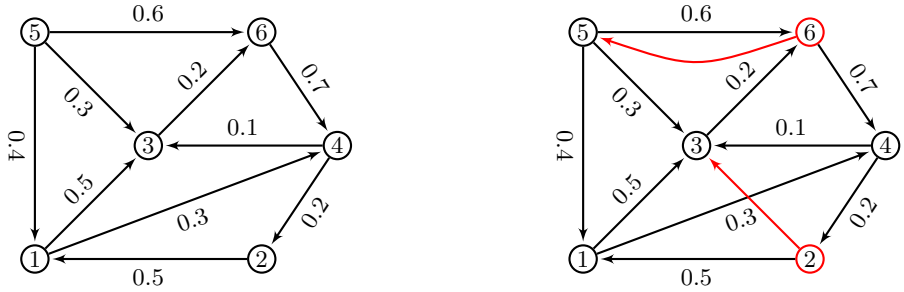

FIGURE 2.1: Graph G, seeds set $A = \{2, 6\}$ and edges set S.

As stated in the introductory chapter, we are interested in taking intervention action to facilitate the information diffusion process. In particular we focus our attention on graph augmentation through edge addition.

Given a set S of edges not in E, we denote by $G(S)$ the graph augmented by adding the edges in S to G, i.e. $G(S) = (V, E \cup S)$. We denote by $\sigma(A, S)$ the influence of A in the augmented graph $G(S)$. We assume that each edge $e \in (V \times V) \setminus E$ can be selected with cost $c_e \in [0, 1]$.

In Figure 2.1 we give an example of a weighted directed graph G. We show in red the set of seed nodes A and the set $S = \{(2, 3), (6, 5)\}$ of new added edges.

We give now some definitions useful to prove our results. We will use the definition of *live-edge graph* $X = (V, E_X)$ which is a directed graph where the set of nodes is equal to V and the set of edges is a subset of E. More specifically, the edge set E_X is given by an edge selection process in which each edge in E is either *live* or *blocked* according to its propagation probability. We can assume that, for each edge $e = (u, v)$ in the graph, a coin of bias p_e is flipped and the edges for which the coin indicated an activation are live, the remaining are blocked. It is easy to show that the information diffusion process is equivalent to a reachability problem in live-edge graphs: given any seed set A, the distribution of active node sets after the diffusion process ends is the same as the distribution of node sets reachable from A in live-edge graphs. In Figure 2.2 we show two possible live-edge graphs of the graph G in Figure 2.1 where the dashed edges are the blocked ones.

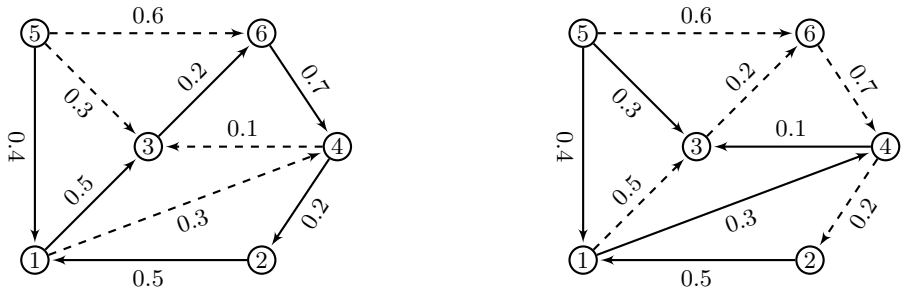

FIGURE 2.2: Example of *live-edge* graphs.

We denote by χ the probability space in which each sample point specifies one possible set of outcomes for all the coin flips on the edges, that is the set of all possible

live-edge graphs of G. For a set of edges $S \subseteq (V \times V) \setminus E$, the set of all possible live edge graphs of $G(S)$ is denoted by $\chi(S)$.

Given two set of edges S, T, such that $S \subseteq T$, for each live-edge graph X in $\chi(S)$ we denote by $\chi(T, X)$ the set of live-edge graphs in $\chi(T)$ that have X as a subgraph and possibly contain other edges in $T \setminus S$. In other words, a live-edge graph in $\chi(T, X)$ has been generated with the same outcomes as X on the coin flips in the edges of $E \cup S$ and it has other outcomes for edges in $T \setminus S$. See Figure 2.3 for a visualization. On the top we show a live-edge graph X in $\chi(S)$ where $S = \{(2, 3)\}$ and on the bottom we show $\chi(T, X)$ for $T = \{(6, 5), (2, 3)\}$.

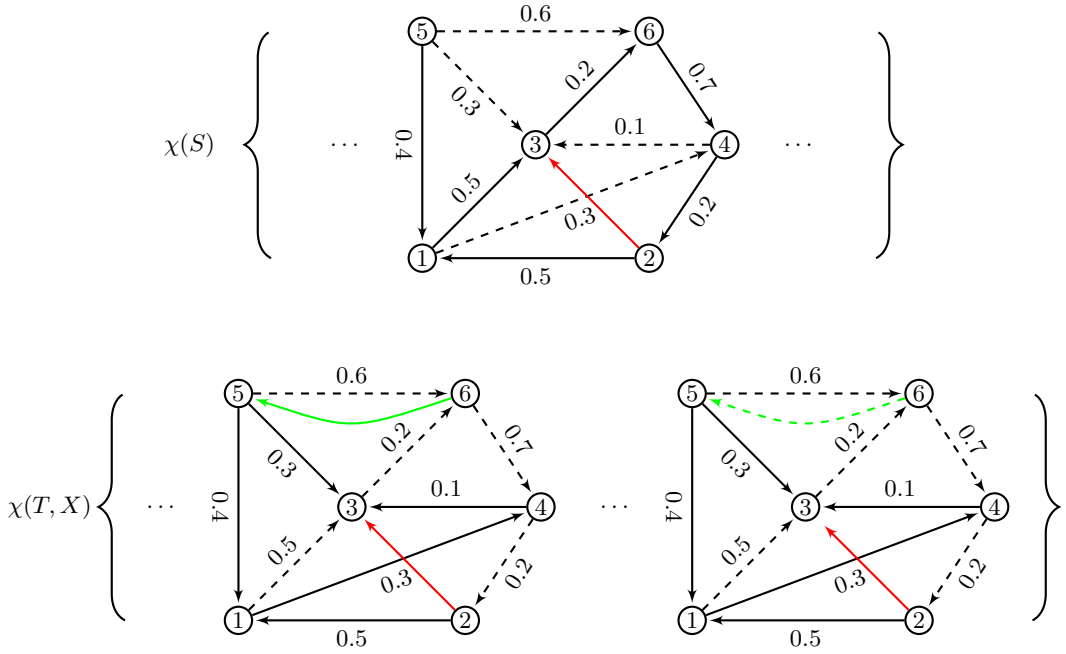

FIGURE 2.3: Examples of probability spaces $\chi(S)$ ans $\chi(T, X)$.

The following holds: $|\chi(T, X)| = 2^{|T \setminus S|}$, for each $Y \in \chi(T, X)$ $\mathbb{P}[Y] = \mathbb{P}[Y|X]\mathbb{P}[X]$, $\mathbb{P}[X] = \sum_{Y \in \chi(T,X)} \mathbb{P}[Y]$, and $\sum_{Y \in \chi(T,X)} \mathbb{P}[Y|X] = 1$.

For a node $a \in V$ and a live-edge graph X in $\chi(S)$, let $R(a, X)$ denote the set of all nodes that can be reached from a in graph X, that is for each node $v \in R(a, X)$, there exists a path from a to v consisting entirely of live edges with respect to the outcome of the coin flips that generates X.

Let $R(A, X) = \bigcup_{a \in A} R(a, X)$, then $\sigma(A, S)$ can be computed as

$$\sigma(A, S) = \sum_{X \in \chi(S)} \mathbb{P}[X] \cdot |R(A, X)|.$$

Given a set of edges S, for each graph $X \in \chi(S)$ and subset of edges $T \subseteq S$, we denote by X^T the graph obtained by removing edges in T from X.

Given two feasible solutions (A_1, S_1) and (A_2, S_2), such that $A_2 \subseteq A_1$ and $S_2 \subseteq S_1$, we denote with $\delta(A_1, S_1, A_2, S_2)$ the expected number of nodes affected by (A_1, S_1) and

not affected by (A_2, S_2), formally:

$$\delta(A_1, S_1, A_2, S_2) = \sum_{X \in \chi(S_1)} \mathbb{P}[X] \cdot \left(|R(A_1, X)| - |R(A_2, X^T)| \right),$$

where $T = S_1 \setminus S_2$. Before proving the following proposition, we give the definition of edges outgoing a set A. Given a set of nodes $A \subseteq V$, we refer to $S \subseteq V \times V$ as an outgoing set of edges if it is such that $S = \{(u, v) | u \in A\}$.

Proposition 1. *For each $A_2 \subseteq A_1 \subseteq V$ and $S_2 \subseteq S_1 \subseteq V \times V$, such that the edges in S_1 and S_2 are outgoing A_1 and A_2, respectively, then*

$$\delta(A_1, S_1, A_2, S_2) = \sigma(A_1, S_1) - \sigma(A_2, S_2).$$

Proof.

$$\begin{aligned}
\delta(A_1, S_1, A_2, S_2) &= \sum_{X \in \chi(S_1)} \mathbb{P}[X] \cdot \left(|R(A_1, X)| - |R(A_2, X^T)| \right) \\
&= \sigma(A_1, S_1) - \sum_{X \in \chi(S_1)} \mathbb{P}[X] \cdot |R(A_2, X^T)| \\
&= \sigma(A_1, S_1) - \sum_{X \in \chi(S_2)} \mathbb{P}[X] \sum_{Y \in \chi(S_1)} \mathbb{P}[Y|X] \cdot |R(A_2, X^T)| \\
&= \sigma(A_1, S_1) - \sigma(A_2, S_2).
\end{aligned}$$

□

Another important notion we use to prove our results is that of *submodular function*. A function z is submodular if it satisfies the following property: the marginal gain from adding an element to a set S is at least as high as the marginal gain from adding the same element to a superset of S. Formally, for a ground set N, a function $z : 2^N \to \mathbb{R}$ is submodular if for any pair of sets $S \subseteq T \subseteq N$ and for any element $e \in N \setminus T$, $z(S \cup \{e\}) - z(S) \geq z(T \cup \{e\}) - z(T)$.

2.2 Influence diffusion models

Several models of information diffusion had been introduced in the literature. Two widely studied models are *Linear Threshold Model* (LTM) [37, 41, 68] and *Independent Cascade Model* (ICM) [33, 34, 40, 41].

2.2.1 Linear Threshold Model

The *Linear Threshold Model* was proposed as an approach to capture influence diffusion dynamics by Granovetter [37] and Schelling [68].

In LTM the idea is that a node becomes active as more of its neighbours become active. Formally, each node u has a threshold t_u chosen uniformly at random in the interval $[0, 1]$. The threshold represents the weighted fraction of neighbours of u that must become active in order for u to become active. The diffusion starts by setting to

active a small percentage of nodes in the graph. Recursively, the affected nodes can influence their neighbours with some probability. During the process, a node u becomes active if the total weight of its active neighbours is greater than t_u:

$$\sum_{(v,u):v \text{ is active}} p_{(v,u)} \geq t_u$$

Thus, these thresholds intuitively represent the different latent tendencies of nodes to adopt the innovation or to be aware of information when their neighbours do.

It has been proved that, even with random node thresholds, finding the best seeds set A to maximize influence spreading is NP-hard [41].

Figure 2.4 shows an example of the diffusion process in the LTM. The red nodes are seeds and the red edges represents the diffusion process. We notice that node 3 becomes active because $0.3 + 0.5 \geq 0.7$ while node 4 remains inactive since $0.3 \leq 0.5$.

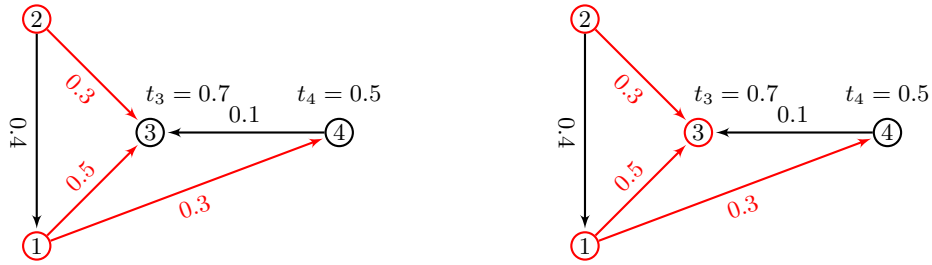

FIGURE 2.4: Diffusion process in LTM.

A generalization of the Linear Threshold Model has been proposed in the literature by lifting the assumption that influences from individual nodes can only be aggregated linearly. In the *General Threshold Model* [41] a node u become active according to an arbitrary monotone function of the set of neighbours of u that are already active.

Since the Influence Maximization problem under the General Threshold Model is highly intractable, a natural restriction is to require the influence functions of nodes to have diminishing influence as the size of the influencing set increases. The property that models diminishing influence of individual additions is submodularity and the model is called *Submodular Threshold Model* [41].

2.2.2 Independent Cascade Model

The *Independent Cascade Model* (ICM) was first introduced by Goldenberg et al. [33, 34]. As in the previous model the diffusion process occurs in discrete steps starting from an initial set of seed nodes. An active node u tries to influence its inactive neighbours but the success of node u in activating the node v only depends on the propagation probability of the edge from u to v (each edge has its own value). Regardless of its success, the same node will never get another chance to activate the same inactive neighbour. The process terminates when no further node gets activated.

The propagation probability can be associated with edges according to different models. In the literature, the most used ways to assign probabilities in the ICM are the:

- uniform scheme that assigns the same probability to each directed edge;
- weighted model that, for each edge (u, v), assigns $p_{(u,v)} = 1/d_v$, where d_v is the in-degree of v;
- trivalency model that assigns a random probability in the set $\{0.1, 0.01, 0.001\}$ to each edge.

As in the LTM, the Influence Maximization problem has been proven to be NP-hard [41] in the ICM and different generalizations of it has been proposed.

The Independent Cascade Model can be naturally generalized, by considering the probability $p_{(u,v)}$, that u succeeds in activating a neighbour v, dependent on the set of v's neighbours that have already tried. In the *General Cascade Model* [41] the information diffusion works in the same way as in the independent case but when a node u tries to activate its neighbours v, it succeeds with probability $p_v(u, S)$, where S is the set of neighbours that have already tried, and failed, to activate v. It is easy to see that in the Independent Cascade model $p_v(u, S) = p_{(u,v)}$, and it is independent of S.

Another generalization is the *Decreasing Cascade Model* [41] which takes into account the effect of previous attempts to activate a node on its chances of being activated in the future, hence it is no longer independent. In this model the probability of a node u to influence v is non-increasing as a function of the set of nodes that have previously tried to influence v. The decreasing aspect is the assumption that the more neighbours try to influence v, the less likely it will become active. This reflects the quality usually refer to as market-saturation.

2.3 Influence Maximization problem

One of the fundamental problems in the study of influence spreading is the problem of *Influence Maximization*. It was first proposed by Domingos and Richardson [30, 66] in the field of viral marketing and asks to find a small set of "influential" seeds in a network in order to activate a large part of the network.

The problem has been formalized by Kempe et al. [41] as follows: given an integer k and a social network represented by a directed graph with users as nodes, edges corresponding to social ties and edge weights capturing influence probabilities, the goal is to find a seed set of k users such that by targeting these, the expected influence spread, defined as the expected number of influenced users, is maximized.

Influence Maximization (IM)	
Given:	A weighted directed graph $G = (V, E, p, c)$; and an integer $k \in \mathbb{N}$
Solution:	A set of seeds $A \subseteq V$ such that $c(A) \leq k$, where $c(A) = \lvert A \rvert$
Goal:	Maximize $\sigma(A)$

In [41], the IM problem has been proven to be NP-hard to approximate to within a factor of $n^{1-\epsilon}$, for any $\epsilon > 0$. The authors, then, focus on the ICM and the LTM and provide a polynomial time approximation algorithm that approximates the optimal solution for Influence Maximization to within a factor of $(1 - \frac{1}{e} - \epsilon)$, where e is the base of the natural algorithm and $\epsilon > 0$ any positive real number.

Theorem 2 ([41]). *In the Linear Threshold and Independent Cascade models there is a polynomial-time algorithm approximating the Maximum Influence to within a factor of* $(1 - \frac{1}{e} - \epsilon)$*, where* ϵ *is any positive constant.*

The algorithm that achieves this performance guarantee uses a greedy strategy. It starts from the empty set $A = \emptyset$ and repeatedly adds to A the node $\hat{a}$ maximizing the marginal gain $\sigma(A \cup \{\hat{a}\}) - \sigma(A)$.

Algorithm 1: Greedy algorithm for IM.

Input : A directed graph $G = (V, E, p, c)$; and an integer $k \in \mathbb{N}$
Output: A set of vertices $A \subseteq V$; such that $c(A) \leq k$, where $c(A) = |A|$

```
A := ∅;
for i = 1, 2, ..., k do
    â = arg max{σ(A ∪ {a}) | a ∈ (V \ A)};
    A := A ∪ {â};
return A;
```

The algorithm, whose pseudocode is reported in Algorithm 1, exploits two observations.

First, the fact that $\sigma(\cdot)$ is a submodular function of the initial set of active nodes A, proved in [41]. The key point of the proof is the exhibition of a distribution over the graph with the following property: the expected activation $\sigma(A)$ equals the expected number of nodes reachable from A if a graph G is chosen according to the distribution.

Second, the results of Nemhauser et al. [59] on the approximation of monotone submodular objective functions. Let us consider the following optimization problem: given a finite set N, an integer k', and a real-valued function z defined on the set of subsets of N, find a set $S \subseteq N$ such that $|S| \leq k'$ and $z(S)$ is maximum. If z is *monotone and submodular*, then the following greedy algorithm exhibits an approximation of $1 - \frac{1}{e}$ [59]: start with the empty set, and repeatedly add an element that gives the maximal marginal gain, that is if S is a partial solution, choose the element $j \in N \setminus S$ that maximizes $z(S \cup \{j\})$.

Theorem 3 ([59]). *For a non-negative, monotone submodular function* z*, let* S *be a set of size* k *obtained by selecting elements one at a time, each time choosing an element that provides the largest marginal increase in the value of* z*. Then* S *provides a* $\left(1 - \frac{1}{e}\right)$*-approximation.*

2.3.1 Diffusion process approximation

We now focus on the estimation $\sigma(\cdot)$ and highlight that, the influence function $\sigma(A)$ cannot be evaluated exactly in polynomial time since it has been proved that it is generally $\#P$-complete for the Independent Cascade Model [13] and for the Linear Threshold Model [15]. However, by simulating the diffusion process sufficiently many times and sampling the resulting active sets, it is possible to obtain arbitrarily good approximations to $\sigma(A)$. The next proposition bounds the number of times that the diffusion process must be simulated to obtain a good approximation of $\sigma(A)$.

Theorem 4 ([41])**.** *If the diffusion process starting with A on graph G is simulated at least $\Omega(\frac{n^2}{\lambda^2}\ln\frac{1}{\delta})$ times, then the average number of activated nodes over these simulations is a $(1\pm\lambda)$-approximation to $\sigma(A)$, with probability at least $1-\delta$.*

It is a easy to see that by using $(1\pm\lambda)$-approximate values for the function to be optimized, the greedy algorithm still guarantees a $(1-\frac{1}{e}-\epsilon)$-approximation, where ϵ depends on λ and goes to 0 as $\lambda\to 0$.

Algorithm 2: Greedy algorithm for IM.

Input : A directed graph $G=(V,E,p,c)$; and an integer $k\in\mathbb{N}$
Output: A set of vertices $A\subseteq V$; such that $c(A)\leq k$, where $c(A)=|A|$

```
A := ∅;
for i = 1, 2, ..., k do
    foreach v ∈ V \ A do
        Use repeated sampling to approximate σ(A ∪ v) to within (1 ± λ) with
        probability 1 − δ;
        Let σ̃(A ∪ {v}) be this estimation.;
    â = arg max{σ̃(A ∪ {a}) | a ∈ (V \ A)};
    A := A ∪ {â};
return A;
```

Algorithm 2 is the modified version of Algorithm 1 which takes into account the simulation of the diffusion process.

In Figure 2.5 we show that the quality of the approximation for $\sigma(A)$ after 500 simulations of the diffusion process is comparable to that after 100000 iterations. We report the results for the *Twitter* network ($n=465017$ nodes and $m=834797$ edges), the results for the other networks are similar and therefore omitted. We run the diffusion process assigning random probabilities to each edge and selecting a random set of seed nodes A such that $|A|=1\%\cdot n$. We plot the ratio between the number of active nodes using $100, 500, 1000, 5000, 10000, 25000, 50000, 100000$ samples and the number of active nodes using 100000 samples, we notice that the ratio is almost 1. We choose 500 as the number of samples to use for our experiments in the following chapters.

Starting from these results on the Influence Maximization problem, we give the definitions of three problems which are at the core of this thesis.

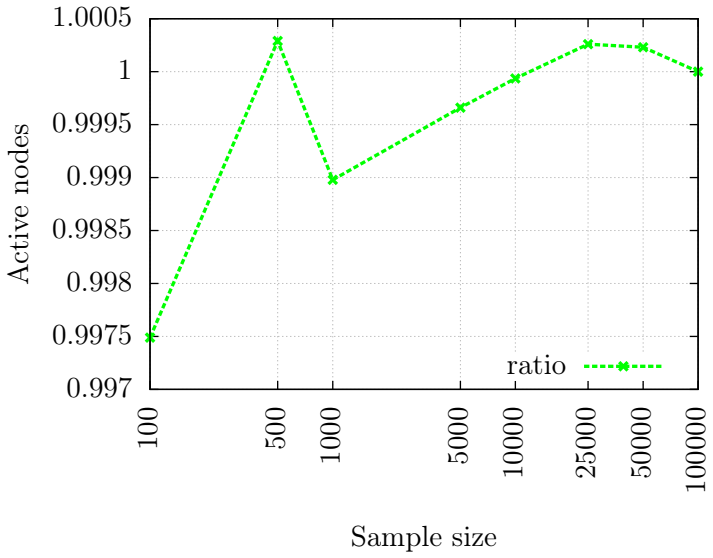

FIGURE 2.5: Approximation of $\sigma(A)$ using repeated sampling.

2.4 Influence Maximization with link addition problems

In this section we define the problems we investigate in Chapter 4, Chapter 5 and Chapter 6.

The first problem we present is the *Influence Maximization with Link addition* (IM-L) problem. In detail, the IM-L problem is defined as follows: given a graph G, an integer k and a set A of seeds, find a set S of edges such that $S \subseteq (A \times V) \setminus E$, $|S| \leq k$, and $\sigma(A, S)$ is maximum.

Influence Maximization-Link addition (IM-L)	
Given:	A weighted directed graph $G = (V, E, p, c)$; a set of seeds $A \subseteq V$; and an integer $k \in \mathbb{N}$
Solution:	A set $S \subseteq (A \times V) \setminus E$ of edges incident to nodes in A, $S = \{(a, v) \mid a \in A\}$, such that $c(S) \leq k$, where $c(S) = \lvert S \rvert$
Goal:	Maximize $\sigma(A, S)$

Moreover, we consider also the general cost version of our problem, we refer to it as the *Budgeted Influence Maximization with Link addition* (BIM-L) problem. Differently from the previous problem, we assume that each edge $e \in (V \times V) \setminus E$ can be selected with cost $c_e \in [0, 1]$. In detail: given a graph G, a budget k and a set A of seeds, find a set S of edges such that $S \subseteq (A \times V) \setminus E$, $c(S) \leq k$, and $\sigma(A, S)$ is maximum, where $c(S) = \sum_{e \in S} c_e$.

Budgeted Influence Maximization-Link addition (BIM-L)	
Given:	A weighted directed graph $G = (V, E, p, c)$; a set of seeds $A \subseteq V$; and a budget $k \in \mathbb{R}^+$
Solution:	A set $S \subseteq (A \times V) \setminus E$ of edges incident to nodes in A, $S = \{(a, v) \mid a \in A\}$, such that $c(S) \leq k$, where $c(S) = \sum_{e \in S} c_e$
Goal:	Maximize $\sigma(A, S)$

Finally we present the *Budgeted Influence Maximization with Seeds selection and Link addition* problem (BIM-SL). This is the most general problem since it includes not only edges addition, but also seeds selection. We look for a set of seeds A and a set of edges S, to be added to G, incident to these seeds that maximize $\sigma(A, S)$. W.l.o.g. we assume that each seed node can be selected with cost 1, while each edge $e \in (V \times V) \setminus E$ can be selected with cost $c_e \in [0, 1]$. More formally the BIM-SL problem is defined as follows: given a graph G and a budget k, find a set A of seeds and a set S of edges such that $S \subseteq (A \times V) \setminus E$, $c(A, S) \leq k$, and $\sigma(A, S)$ is maximum, where $c(A, S) = |A| + \sum_{e \in S} c_e$.

Budgeted Influence Maximization-Seeds selection Link addition (BIM-SL)	
Given:	A weighted directed graph $G = (V, E, p, c)$; and a budget $k \in \mathbb{R}^+$
Solution:	A set on seeds $A \subseteq V$, a set $S \subseteq (A \times V) \setminus E$ of edges incident to nodes in A, $S = \{(a, v) \mid a \in A\}$, such that $c(A, S) \leq k$, where $c(A, S) = \lvert A \rvert + \sum_{e \in S} c_e$
Goal:	Maximize $\sigma(A, S)$

Chapter 3

A survey of related works

The seeds selection, in the Influence Maximization problem, has been widely studied and lot of effort has been made in order to reduce the running time. We can distinguish between heuristics which maintain the $(1 - \frac{1}{e})$-approximation guarantee of the greedy algorithm while speeding up the computation in practice and heuristics which guarantee a faster running time, but do not provide guarantees or give significantly weaker ones.

Beside seed selection, other actions may be used to modify a graph in order to maximize or minimize the number of influenced nodes. These actions include edges and nodes addition or deletion.

In the following we focus on both seeds selection and graph modification techniques.

3.1 Seed selection problem

Several performance improvements have been proposed to the seed selection problem. Leskovec et al. [51] proposed the CELF heuristic, based on a lazy evaluation of the marginal contribution of the objective function. The main idea exploits the submodularity definition and it is the following. If a node's marginal contribution in the previous iteration of the greedy algorithm was already smaller than the current best node's, then it need not be evaluated again since its contribution in the current iteration can only be lower. They run experiments on the detection of contaminations in a water distribution network and on the selection of informative blogs in a network. Few years after, Goyal et al. [36] proposed CELF++ which is an improvement to CELF trough several heuristics which avoid unnecessary re-computations of marginal gains incurred by CELF. The same authors propose an alternative algorithm called SIMPATH [35]. SIMPATH exploits two optimization techniques: the first cuts down the spread estimation calls in the first iteration, while the second improves the efficiency in subsequent iterations. Instead of using expensive Monte-Carlo simulations to estimate the spread, the authors show that under the LT model, the spread can be computed by enumerating the simple paths starting from the seed nodes since they notice that the majority of the influence flows within a small neighbourhood, because probabilities of paths diminish rapidly as they get longer.

To support massive graphs, several studies focus is on speeding up the computation of the objective function via approximations.

Chen et al. [14] propose a discounted high-degree heuristic. The core idea is that, while selecting seed nodes according to their degree, when considering selecting v, we should not count the edge (u, v) such that node u has been selected as a seed, towards its degree. Thus we should discount v's degree by one due to the presence of u in the seed set. Thanks to this refinement of the highest degree node selection, they were able to analyse large-scale social networks.

Jung et al. [39] design IRIE, a heuristic approximation of greedy addition of seed nodes. They propose a novel global influence ranking method IR derived from a belief propagation approach, which uses a small number of iterations to generate a global influence ranking of the nodes and then select the highest ranked node as the seed instead of estimating influence spread for each node at each round. However, this approach works for selecting a single seed, if used for selecting more than one, then their influence spread may overlap with one another and not result in the best overall influence spread. To overcome this issue, the authors integrate IR with a simple influence estimation method, called IE, such that after one seed is selected, it estimates the additional influence impact of this seed to each node in the network, which is much faster than estimating marginal influence for many seed candidates, and then use the results to adjust next round computation of influence ranking.

An algorithm with theoretical guarantees, was proposed by Borgs et al. [11]. In order to speed up the repeated computation of the influence of nodes, they propose a preprocessing step which generates a random hypergraph sampled according to reverse reachability probabilities in the original graph. They explain that when selecting a node v uniformly at random, and determining the set of nodes that would have influenced v, a certain node u will appear often as an "influencer" then it is likely a good candidate for a seed node.

Recently Tang et al. [71] developed TIM. The Two-phase Influence Maximization algorithm consists in two steps: the parameter estimation phase computes a lower-bound of the maximum expected spread among all node sets of size k, and then uses the lower-bound to derive a parameter θ; the node selection phase samples θ reverse reachable sets from the graph, and then derives a size-k node set that covers a large number of reverse reachable sets and returns it as the final result. The main drawback is that the $(1 - \frac{1}{e} - \epsilon)$ approximation is guaranteed only for a given seed set of size k.

The limit of TIM was overcome by IMM (Influence Maximization with Martingales) proposed by the same authors in [70]. This new algorithm provides the same worst-case guarantees as the state of the art, but offers an improved empirical efficiency. The core of the approach is based on martingales, a statistical tool.

Finally, Cohen et al. [20] develop a novel sketch-based design for influence computation. The greedy Sketch-based Influence Maximization (SKIM) algorithm scales to graphs with billions of edges and still has a guaranteed approximation ratio. In practice its quality nearly matches that of exact greedy. The core of this approach is a structure called combined reachability sketch. For a node, a sketch represent the influence coverage across the sampled instances. We give a detailed description of this technique

in Chapter 4 since, to the best of our knowledge, this is the fastest algorithm to solve the Influence Maximization problem and we explain how we make use of it to solve our Influence Maximization with link addition problems.

In [63] Nguyen et al. define the Budgeted Influence Maximization (BIM) problem as follows: given a fixed budget and an arbitrary cost for selecting each node, select the seed nodes to maximize the total number of nodes influenced in social networks at a total cost no more than the budget. The proposed algorithm for the BIM problem guarantees an approximation ratio of $(1-1/\sqrt{e})$. They identify the linkage between the computation of marginal probabilities in Bayesian networks and the influence spread, which allows them to devise efficient heuristic algorithms for the latter problem.

3.2 Graph modification problem

The problem of recommending links by taking into account the information spreading capability, instead, has received little attention in the literature. In the following we focus on such problem and on the problems of modifying a graph in order to maximize or minimize the spread of information through a network under the LTM and the ICM.

Yu et al. [76] in their work consider both the recommendation success rate and content spread in the network. They propose an Algebraic Connectivity Regularized Friends-of-Friends recommendation algorithm, called ACR-FoF, that uses the algebraic connectivity of a connected network to estimate its capability for spreading contents. The algebraic connectivity of G is the second-smallest eigenvalue λ_2 of the Laplacian matrix of G and the authors show that, when adding edges, maximizing λ_2 is equivalent to minimizing the upper bound of the diameter and the mean distance. This is why optimizing algebraic connectivity will improve all the content spread according to the three metrics described in the paper: expected content spreading (ECS), used in [12]; network diameter; and enhanced pairs (EnPair), the number of node pairs whose content sharing probability is improved by adding new edges. The authors give experimental evidence that ACR-FoF achieves significant improvement in content spread at a very tiny loss on recommendation accuracy, but do not prove any approximation guarantees.

Chaoji et al. [12] study the problem of maximizing the expected number of nodes influenced by some contents by adding a set of edges under the constraint that each node has at most k incident new edges. The authors consider a model in which each node is associated with an independent probability to share a content with all its neighbours and with a set of contents that are generated by the node itself. They show that the problem is NP-hard and propose an information diffusion model called Restricted Maximum Probability Path Model in which a content is propagated between two users along the path with maximum probability among those containing a recommended edge. They show that under this model the objective function is submodular and hence the problem can be approximated within a constant bound. Moreover, they run simulations on realistic graphs and the results demonstrate the superiority of their approach in comparison with commonly used heuristics.

Li et al. [52] introduce the notion of user diffusion degree which is a measure of the influence that a user has and is computed by combining community detection algorithms with information diffusion models. They propose an algorithm that suggests links by combining the diffusion degree with the FoF algorithm. The main idea is: for a user i, if he has a large number of neighbour pairs who are not directly connected and the users connected to user i belong to many different groups, then user i gets a high diffusion degree. By means of experiments on two networks, they show that their algorithm outperforms some known baseline in terms of number of affected nodes under the ICM and LTM models.

In what follows, we outline the graph modification problems that have been studied for LTM. Khalil et al. [42] consider two types of graph modification, adding edges to or deleting edges from the existing network to minimize the information diffusion and they show that this network structure modification problem has a supermodular objective and therefore can be solved by algorithms with provable approximation guarantees. Following the work of Kempe et al. [41], they exploit the correspondence between LTM and the live-edge graph construction, and estimate the objective function sampling over random live-edge graphs. In order to support approximate evaluation of the objective function, they propose an approach that allows to scale to networks with millions of nodes by designing two data structures: the descendant-count trees for the edge deletion problem and the neighbour-counting graphs for the edge addition problem. Therefore, they evaluate the algorithms on both synthetic and real-world diffusion networks.

Zhang et al. [77] formulate Group Immunization problems that involve both edge and node removal, modelling quarantining and vaccination. They develop algorithms with rigorous performance guarantees and good empirical performance.

Kimura et al. [45] use a greedy approach to delete edges under the LTM but do not provide any rigorous approximation guarantees. They have experimentally demonstrated that the proposed method works and significantly outperforms the conventional link-removal heuristics based on the betweenness and out-degree.

Kuhlman et al. [46] propose heuristic algorithms, called edge-covering heuristic, for edge removal under a simpler deterministic variant of LTM which is not only hard, but also has no approximation guarantee. In detail: in the first part of the heuristic, the dynamics are simulated on the network, up to time T and times at which nodes become affected are recorded. Then, it proceeds from step 1 to T determining whether the total cost of all edges used to transport contagion to affected nodes is less than the budget. If so, then these edges constitute the solution. Otherwise, the necessary number of least cost edges required to save each affected node is computed.

Papagelis [64] and Crescenzi et al. [24] study the problem of augmenting the graph in order to increase the connectivity or the centrality of a node, respectively and experimentally show that this increases the expected number of eventual active nodes. In particular, Crescenzi et al. consider the optimization problem of determining how much a vertex can increase its harmonic centrality by creating a limited amount of new edges incident to it. Formally the *harmonic centrality* of a node u is equal to the sum

of the reciprocal of the distances to u from all other vertices. This measure somehow evaluates the efficiency of a vertex while spreading information to all other vertices in its connected component. They consider both the undirected and the directed graph cases. In both cases, they first prove that the optimization problem does not admit a polynomial-time approximation scheme, unless $P = NP$, and then propose a greedy approximation algorithm, whose performance is then tested on synthetic graphs and real-world networks.

Furthermore, the authors show, by means of experiments, how adding a limited number of edges incident to some randomly-chosen seeds highly increases the number of nodes that become active in the threshold model. In the experiments, they choose a small percentage of nodes as seeds and they run different experiments where the threshold t_u is uniform, i.e. all the nodes have the same threshold, or distributed uniformly at random. Their results clearly show that the number of active nodes highly increases even with few edges addition and that the percentage of active nodes tends to 100%.

They compare their results to the solution obtained by alternative baselines such as connecting seeds to random nodes and adding edges to nodes with highest degree or to nodes with highest harmonic centrality. They observe that such solutions are competitive with that obtained by the proposed greedy algorithm if the number k of added edges is small (i.e. $k \leq 3$). If $k > 3$, the percentage of informed nodes is smaller than that obtained by using the greedy algorithm and the gap between the baselines and the greedy algorithm increases with k.

Parotsidis et al. [65] study the problem of recommending links with the objective of improving the centrality of a node within a network. They show that the problem is NP-hard, but the objective function is monotone and submodular and as a result the greedy algorithm gives a solution with constant approximation factor.

Under ICM, Wu et al. [75] consider graph modifications other than edge addition, edge deletion and source selection, such as increasing the probability that a node infects its neighbours. They proved that optimizing the selection of such modifications with a limited budget is NP-hard and is neither submodular nor supermodular. They designed algorithms for directed rooted trees and for general directed graph and the results showed that they are faster than existing algorithms while producing near optimal solutions.

Sheldon et al. [69] study the problem of node addition to maximize the spread of information, and provide a counterexample showing that the objective function is not submodular. They propose a mixed integer programming formulation and their approach results in solutions with stochastic optimality guarantees.

Kimura et al. [44] propose methods for efficiently finding good approximate solutions on the basis of a greedy strategy for the edge deletion problem under the ICM, but do not provide any approximation guarantees.

D'Angelo et al. [25] introduce a preliminary version of the edge addition problem where new edges are incident to a given initial set of active nodes. They focus on the

case in which the initial set of seeds is a singleton and they investigate the existence of a constant approximation algorithm.

Other works consider the problem of activating all the network by adding links such as Cordasco et al. [22]. The authors study the Minimum Links Problem: the aim is to find the minimum number of links that a set of external influencers should form to people in the network, in order to activate the entire social network. They prove that the Minimum Links problem cannot be approximated to within a ratio of $O(2^{log^{1-\epsilon}n})$ or any fixed $\epsilon \geq 0$ unless $NP \subseteq DTIME(n^{polylog(n)})$, where n is the number of nodes in the network. They give linear time algorithms to solve the problem for trees, cycles, and cliques. For general graphs, instead, they design a polynomial time algorithm to compute size-efficient link sets that can activate the entire graph.

Chapter 4

Link addition problem

In this chapter, we study the problem of *Influence Maximization with Link addition*. We analyse the unit-cost version of the problem providing hardness of approximation results on directed networks. Then we propose a greedy approximation algorithm with an almost tight approximation ratio and we evaluate the performance of our algorithm on both synthetic and real networks. Most of the results presented in this chapter are included in [27, 25].

4.1 Problem definition

Given a weighted directed graph $G = (V, E, p, c)$, where $p : V \times V \to [0, 1]$ is the propagation probability of an edge and $c : V \times V \to [0, 1]$ is the cost of adding an edge, we investigate the IM-L problem.

The IM-L problem is the unit-cost version of the more general BIM-L problem. Differently from the general case, we assume that each edge $e \in (V \times V) \setminus E$ can be selected with cost $c_e = 1$. In detail: given a graph $G = (V, E, p, c)$, a budget k and a set A of seeds, find a set S of edges such that $S \subseteq (A \times V) \setminus E$, $c(S) \leq k$, and $\sigma(A, S)$ is maximum, where $c(S) = |S|$.

Even if IM-L problem represents a limited number of real scenarios, it can be approximated using a greedy algorithm that guarantees an approximation factor of $1 - \frac{1}{e}$ exploiting the properties of submodular functions.

In the following we give our results on the IM-L problem and then, in the next chapter, we extend them to the BIM-L problem.

In the following proofs we use the definitions given in Chapter 2. In particular, we slightly change the definition of the expected number of nodes affected by a solution (A_1, S_1) and not affected by (A_2, S_2), i.e. $\delta(A_1, S_1, A_2, S_2)$, in $\delta(S_1, S_2)$ since the set of seed nodes is given and A_1 is equal to A_2. Formally:

$$\delta(S_1, S_2) = \sum_{X \in \chi(S_1)} \mathbb{P}[X] \cdot \big(|R(A, X)| - |R(A, X^T)|\big)$$

where $T = S_1 \setminus S_2$.

4.2 Hardness result

In this section we show that IM-L problem does not admit a PTAS, unless $P = NP$.

A preliminary result is given in [25], where it is shown that the statement holds even when $|A| = 1$.

Theorem 5. *It is NP-hard to approximate* IM-L *within a factor greater than $1 - \frac{1}{2e}$ for any A such that $|A| \geq 1$.*

The proof is based on a reduction from the *Maximum Set Coverage* problem (MSC) which has been shown to be NP-hard to approximate within a factor greater than $1 - \frac{1}{e}$ [32]. In the MSC problem, we are given collection of sets defined over a domain of elements and an integer k', the goal is to select at most k' of these sets such that we cover the maximum number of elements. In detail:

Maximum Set Coverage (MSC)	
Given:	A finite set X, a finite family $\mathcal{F}$ of subsets of X, and an integer k',
Solution:	A family $\mathcal{F}' \subseteq \mathcal{F}$ such that $\lvert\mathcal{F}'\rvert \leq k'$
Goal:	Maximize $s(\mathcal{F}') = \lvert \cup_{S_i \in \mathcal{F}'} S_i \rvert$

Proof. We follow the scheme of L-reductions, since it has been shown that if there is an L-reduction with parameters a and b from maximization problem Π to maximization problem Π', and there is an α-approximation algorithm for Π', then there is an $(1 - ab(1-\alpha))$-approximation algorithm for Π [74, Chapter 16].

Applying this result to our problem, we can show that if there exists an α-approximation algorithm $A_{\text{IM-L}}$ for IM-L and the following two conditions are satisfied for some values a and b:

$$OPT(I_{\text{IM-L}}) \leq aOPT(I_{\text{MSC}}) \tag{4.1}$$

$$OPT(I_{\text{MSC}}) - s(S_{\text{MSC}}) \leq b\left(OPT(I_{\text{IM-L}}) - \sigma(A, S_{\text{IM-L}})\right), \tag{4.2}$$

where OPT denotes the optimal value of an instance of an optimization problem, then there exists an approximation algorithm A_{MSC} for MSC with an approximation factor of $1 - ab(1-\alpha)$. Since it is NP-hard to approximate MSC within a factor greater than $1 - \frac{1}{e}$ [32], then the approximation factor of A_{MSC} must be smaller than $1 - \frac{1}{e}$, unless $P = NP$. This implies that $1 - ab(1-\alpha) < 1 - \frac{1}{e}$ that is, the approximation factor α of $A_{\text{IM-L}}$ must satisfy $\alpha < 1 - \frac{1}{abe}$, unless $P = NP$.

Therefore, in what follows we give two polynomial-time algorithms: the first one transforms any instance $I_{\text{MSC}} = (X, \mathcal{F}, k')$ of MSC into an instance $I_{\text{IM-L}} = (G, A, k)$ of IM-L and the second algorithm transforms any solution $S_{\text{IM-L}}$ for $I_{\text{IM-L}}$ into a solution S_{MSC} for I_{MSC} such that the above two conditions are satisfied.

Given $I_{\text{MSC}} = (X, \mathcal{F}, k')$, we define $I_{\text{IM-L}} = (G, A, k)$ where $G = (V, E, p)$ is a directed graph containing a node v_{x_i} for each element $x_i \in X$, a node v_{S_j} for each set $S_j \in \mathcal{F}$, a seed node a, and a directed edge (v_{S_j}, v_{x_i}), whenever $x_i \in S_j$. We define $k = k'$ and $A = \{a\}$. We denote by $V_{\mathcal{F}}$ the set of nodes corresponding to sets in

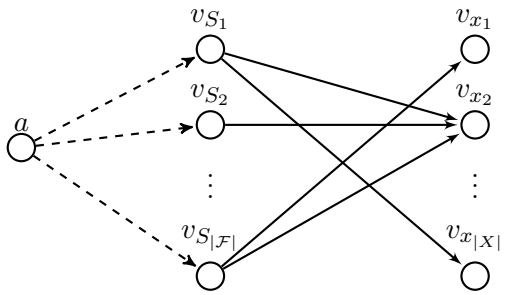

FIGURE 4.1: Reduction used in Theorem 5. The dashed arcs denote those added in a solution.

$\mathcal{F}$. The propagation probability p_e is equal to 1 if $e \in E \cup A \times V_{\mathcal{F}}$ and 0 otherwise. See Figure 4.1 for a visualization. Note that in $I_{\text{IM-L}}$, the information diffusion is a deterministic process, as all probabilities are 0 or 1. Moreover, any solution S for $I_{\text{IM-L}}$ contains only edges (a, v_{S_j}), for some $S_j \in \mathcal{F}$, since any other edge would have probability 0 of being activated. Given a solution $S_{\text{IM-L}} = \{(a, v_{S_j}) \mid S_j \in \mathcal{F}\}$ to $I_{\text{IM-L}}$, we construct the solution $S_{\text{MSC}} = \{S_j \mid (a, v_{S_j}) \in S_{\text{IM-L}}\}$ to I_{MSC}. W.l.o.g., we can assume that $|S_{\text{IM-L}}| = k$ and since by construction $|S_{\text{MSC}}| = |S_{\text{IM-L}}|$, then $|S_{\text{MSC}}| = k = k'$.

The nodes influenced by node a in $S_{\text{IM-L}}$ are all nodes v_{S_j} such that $(a, v_{S_j}) \in S_{\text{IM-L}}$ and all the nodes v_{x_i} such that $x_i \in S_j$, for some S_j such that $(a, v_{S_j}) \in S_{\text{IM-L}}$. The nodes of the former type are k, while the nodes of the latter type are all those nodes v_{x_i} such that $x_i \in \cup_{(a,v_{S_j}) \in S_{\text{IM-L}}} S_j$. Therefore,

$$\sigma(A, S_{\text{IM-L}}) = k + |\cup_{(a,v_{S_j}) \in S_{\text{IM-L}}} S_j| = k + s(S_{\text{MSC}}).$$

It follows that Conditions (4.1) and (4.2) are satisfied for $a = 2$, $b = 1$ since:

$$OPT(I_{\text{IM-L}}) = OPT(I_{\text{MSC}}) + k \leq 2OPT(I_{\text{MSC}}), \quad \text{and}$$

$$OPT(I_{\text{MSC}}) - s(S_{\text{MSC}}) = OPT(I_{\text{IM-L}}) - \sigma(A, S_{\text{IM-L}}),$$

where the first inequality is due to the fact that $OPT(I_{\text{MSC}}) \geq k$, as otherwise the greedy algorithm finds an optimal solution for MSC. The statement follows by plugging the values of a and b into $\alpha < 1 - \frac{1}{abe}$. This concludes the proof for $|A| = 1$. We can extend this result to cases in which $|A| > 1$ by adding nodes in A with the limitation that they can only add edges with propagation probability equal to 0. □

4.3 Approximation algorithm

In this section, we provide an algorithm that guarantees a constant approximation ratio for the IM-L problem.

Our greedy algorithm, whose code is reported in Algorithm 3, iterates k times and, at each iteration, it adds to an initially empty solution S an edge $\hat{e} = (\hat{a}, \hat{v})$

s.t. $(\hat{a}, \hat{v}) \in (A \times V) \setminus E$ that, when added to S, gives the largest marginal increase in the value of $\sigma(A, S)$, that is $\sigma(A, S \cup \{\hat{e}\})$ is maximum among all the possible edges in $(A \times V) \setminus (E \cup S)$ to be added to S.

Algorithm 3: Greedy algorithm for IM-L.

Input : A weighted directed graph $G = (V, E, p, c)$; a set of vertices $A \subseteq V$; and an integer $k \in \mathbb{N}$

Output: Set of edges $S \subseteq (A \times V) \setminus E$ such that $c(S) \leq k$, where $c(S) = |S|$

```
S := ∅;
for i = 1, 2, ..., k do
    ê = arg max{σ(A, S ∪ {e}) | e = (a, v) ∈ (A × V) \ (E ∪ S)};
    S := S ∪ {ê};
return S;
```

The algorithm exploits the results of Nemhauser et al. on the approximation of monotone submodular objective functions [59]. Therefore we show that $\sigma(A, \cdot)$ is monotone and submodular with respect to the possible set of edges incident to nodes in A. In fact, assuming that for a set $S \subseteq (A \times V) \setminus E$ we are able to compute $\sigma(A, S)$, Algorithm 3 provides a $\left(1 - \frac{1}{e}\right)$-approximation.

Theorem 6. *Given a weighted directed graph $G = (V, E, p, c)$, $\sigma(A, S)$ is a monotonically increasing submodular function of sets $S \subseteq (A \times V) \setminus E$.*

In order to prove Theorem 6, we first show, in the next lemma, that function $R(A, \cdot)$ is submodular with respect to the insertion, in the live-edge graphs, of outgoing edges from nodes in A. Note that this is a deterministic property.

Lemma 7. *Given a set of nodes nodes $A \subseteq V$, two live-edge graphs X and Y such that $E_X \subseteq E_Y$ and $E_Y \setminus E_X \subseteq A \times V$, and an edge $e \in (A \times V) \setminus E$, let X^+ and Y^+ denote the live-edge graphs obtained by adding e to X and Y, respectively, that is $X^+ = (V, E_X \cup \{e\})$ and $Y^+ = (V, E_Y \cup \{e\})$. Then,*

$$|R(A, Y^+)| - |R(A, Y)| \leq |R(A, X^+)| - |R(A, X)|.$$

Proof. Let $r(X, e)$ be the set of nodes that are reachable from A in X^+ by means of edge e and that are not reachable in X, formally: $r(X, e) = R(A, X^+) \setminus R(A, X)$. Similarly, let $r(Y, e) = R(A, Y^+) \setminus R(A, Y)$.

Since $E_X \subseteq E_Y$ and $E_Y \setminus E_X \subseteq A \times V$, then $R(A, X) \subseteq R(A, Y)$ and the set of nodes that are reachable from A *only* by means of edge e is smaller in Y^+ than in X^+. It follows that $|r(X, e)| \geq |r(Y, e)|$. Therefore,

$$|R(A, Y^+)| - |R(A, Y)| = |r(X, e)| \geq |r(Y, e)| = |R(A, X^+)| - |R(A, X)|.$$

□

We can now prove Theorem 6.

Proof. (Theorem 6) We divide the proof in two parts. First we prove that $\sigma(A, \cdot)$ a is monotonically increasing function, then we prove that it is a submodular function. To prove the first property, we show that for each $S \subseteq (A \times V) \setminus E$ and $e = (a, v) \in (A \times V) \setminus (E \cup S)$, $\sigma(A, S \cup e) - \sigma(A, S) \geq 0$, that is

$$\sum_{X \in \chi(S \cup \{e\})} \mathbb{P}[X] \cdot |R(A, X)| - \sum_{X \in \chi(S)} \mathbb{P}[X] \cdot |R(A, X)| \geq 0. \tag{4.3}$$

For each live-edge graph X in $\chi(S)$, there are two different corresponding live-edge graph X^+ and X^- in $\chi(S \cup \{e\})$, whose edge sets depend on the outcome of the coin flipped for e, that is $E_{X^+} = E_X \cup \{e\}$ and $E_{X^-} = E_X$, respectively. The probabilities for the live-edge graph X^+ and X^- to occur, are:

- $\mathbb{P}[X^+] = \mathbb{P}[X] \cdot p_e$
- $\mathbb{P}[X^-] = \mathbb{P}[X] \cdot (1 - p_e)$,

while the set of reachable nodes are such that

- $R(A, X) \subseteq R(A, X^+)$
- $R(A, X) = R(A, X^-)$, because in X^+ there is one more edge e and $X^- = X$.

Therefore, $|R(a, X^+)| \geq |R(a, X)|$ and we can write Inequality (4.3) as:

$$\sum_{X \in \chi(S)} \big(\mathbb{P}[X^+] \cdot |R(A, X^+)| + \mathbb{P}[X^-] \cdot |R(A, X^-)|\big) - \sum_{X \in \chi(S)} \mathbb{P}[X] \cdot |R(A, X)| =$$
$$\sum_{X \in \chi(S)} \big(\mathbb{P}[X] \cdot p_e \cdot |R(A, X^+)| + \mathbb{P}[X] \cdot (1 - p_e) \cdot |R(A, X^-)| - \mathbb{P}[X] \cdot |R(A, X)|\big) \geq$$
$$\sum_{X \in \chi(S)} (\mathbb{P}[X] \cdot p_e \cdot |R(A, X)| + \mathbb{P}[X] \cdot (1 - p_e) \cdot |R(A, X)| - \mathbb{P}[X] \cdot |R(A, X)|) = 0.$$

this shows that σ is a monotonically increasing.

To prove the submodularity, we show that for each pair of sets S, T such that $S \subseteq T \subset (A \times V) \setminus E$ and for each $e = (a, v) \in (A \times V) \setminus T$, the increment in expected number of influenced nodes that edge e causes in $S \cup \{e\}$ is larger than the increment it produces in $T \cup \{e\}$, that is $\sigma(A, S \cup \{e\}) - \sigma(A, S) \geq \sigma(A, T \cup \{e\}) - \sigma(A, T)$, or

$$\sum_{X \in \chi(S \cup \{e\})} \mathbb{P}[X] \cdot |R(A, X)| - \sum_{X \in \chi(S)} \mathbb{P}[X] \cdot |R(A, X)| \geq \tag{4.4}$$
$$\sum_{X \in \chi(T \cup \{e\})} \mathbb{P}[X] \cdot |R(A, X)| - \sum_{X \in \chi(T)} \mathbb{P}[X] \cdot |R(A, X)|. \tag{4.5}$$

As in the proof for monotonicity, for each live-edge graph X in $\chi(T)$, let X^+ and X^- be the live-edge graphs in $\chi(T \cup \{e\})$ such that $E_{X^+} = E_X \cup \{e\}$ and $E_{X^-} = E_X$, respectively.

Again, $\mathbb{P}[X^+] = \mathbb{P}[X] \cdot p_e$, $\mathbb{P}[X^-] = \mathbb{P}[X] \cdot (1 - p_e)$, $R(A, X) \subseteq R(A, X^+)$, and $R(A, X) = R(A, X^-)$.

Then, Formula (4.4) is equal to

$$\sum_{X\in\chi(S)} \left(\mathbb{P}[X^+]\cdot|R(A,X^+)| + \mathbb{P}[X^-]\cdot|R(A,X^-)| - \mathbb{P}[X]\cdot|R(A,X)|\right) =$$
$$\sum_{X\in\chi(S)} \left(\mathbb{P}[X]\cdot p_e\cdot|R(A,X^+)| + \mathbb{P}[X]\cdot(1-p_e)\cdot|R(A,X^-)| - \mathbb{P}[X]\cdot|R(A,X)|\right)$$
$$\sum_{X\in\chi(S)} \mathbb{P}[X]\cdot p_e\cdot(|R(A,X^+)| - |R(A,X)|).$$

While, Formula (4.5) is equal to

$$\sum_{X\in\chi(S)}\sum_{Y\in\chi(T,X)} \left(\mathbb{P}[Y^+]\cdot|R(A,Y^+)| + \mathbb{P}[Y^-]\cdot|R(A,Y^-)| - \mathbb{P}[Y]\cdot|R(A,Y)|\right) =$$
$$\sum_{X\in\chi(S)}\sum_{Y\in\chi(T,X)} \left(\mathbb{P}[Y]\cdot p_e\cdot(|R(A,Y^+)| - |R(A,Y)|)\right).$$

By Lemma 7, $|R(A,Y^+)| - |R(A,Y)| \leq |R(A,X^+)| - |R(A,X)|$.
Moreover, $\sum_{Y\in\chi(T,X)}\mathbb{P}[Y] = \mathbb{P}[X]$ and then

$$\sum_{X\in\chi(S)}\sum_{Y\in\chi(T,X)} \mathbb{P}[Y]\cdot p_e\cdot(|R(A,Y^+)| - |R(A,Y)|) \leq$$
$$\sum_{X\in\chi(S)} \mathbb{P}[X]\cdot p_e\cdot(|R(A,X^+)| - |R(A,X)|),$$

which concludes the proof. □

Since we proved that $\sigma(\cdot)$ is a monotone submodular function and we assumed to be able to compute $\sigma(A,S)$, we can apply the theorem by Nemhauser et al.[59] and we can state that:

Theorem 8. *Algorithm 3 guarantees an approximation factor of* $\left(1-\frac{1}{e}\right)$ *for the* IM-L *problem.*

We showed is Chapter 2 how to find an arbitrarily good approximation of $\sigma(A,S)$ in polynomial time exploiting Theorem 4. We can generalize the theorem by Nemhauser et al.[59] to the case in which each step of the greedy algorithm selects an element whose marginal increment is within a factor $(1\pm\lambda)$ to the maximal one. In this case, the greedy algorithm guarantees a $\left(1-\frac{1}{e}-\epsilon\right)$-approximation, where ϵ depends on λ and goes to 0 as $\lambda\to 0$. By combining these results, we can formally define Algorithm 4 that differs from Algorithm 3 on how it computes $\sigma(A,S)$.

Theorem 9. *Algorithm 4 guarantees an approximation factor of* $\left(1-\frac{1}{e}-\epsilon\right)$ *for the* IM-L *problem, where* ϵ *is any positive real number.*

Note that the computational complexity of Algorithm 4 is $O(k\cdot n\cdot g(n,m+k))$, where $g(n,m+k)$ is the complexity of computing an approximation $\tilde{\sigma}(A,S)$ of $\sigma(A,S)$ in a graph with n nodes and $m+k$ edges. Specifically, it runs in k iterations, each of which

Algorithm 4: Greedy algorithm for IM-L with approximate estimation of marginal increment.

Input : A directed graph $G = (V, E, p, c)$; a set of vertices $A \subseteq V$; and an integer $k \in \mathbb{N}$
Output: Set of edges $S \subseteq (A \times V) \setminus E$ such that $c(S) \leq k$, where $c(S) = |S|$

1 $S := \emptyset$;
2 **for** $i = 1, 2, \ldots, k$ **do**
3 **foreach** $e \in (A \times V) \setminus (E \cup S)$ **do**
4 Use repeated sampling to estimate a $(1 + \lambda)$-approximation of $\sigma(A, S \cup \{e\})$ with prob. $1 - \delta$;
5 Let $\tilde{\sigma}(A, S \cup \{e\})$ be the estimation;
6 $\hat{e} = \arg\max\{\tilde{\sigma}(A, S \cup \{e\}) \mid e = (a, v) \in (A \times V) \setminus (E \cup S)\}$;
7 $S := S \cup \{\hat{e}\}$;
8 **return** S;

requires estimating the expected spread of $O(n)$ node sets. Since $g(n, m+k) = O(m \cdot R)$ where R is the number of simulations, then the complexity of the greedy algorithm is $O(k \cdot n \cdot m \cdot R)$ which is clearly too much in terms of computational time for networks with millions of vertices and edges, the "normal" size of many interesting real-world networks.

4.4 Improving the running time

In what follows we propose some techniques to heuristically improve the running time of the greedy algorithm. In Section 4.7 we evaluate an implementation of the algorithm that exploits a combination of all such heuristics.

Exploiting submodularity. Let $\delta(S \cup \{(u, v)\}, S)$ be the increment to the expected number of active nodes after adding the arc (u, v) to graph $G(S)$. Since $\sigma(A, S)$ is submodular, then $\delta(S \cup \{(u, v)\}, S)$ is monotonic non-increasing. It follows that $\delta(S \cup \{(u, v)\}, S)$ is upper bounded by $\delta(S' \cup \{(u, v)\}, S)$, where $S' \subseteq S$. Let us consider the loop at line 2 of Algorithm 4. If at iteration $i \geq 2$, for some edge (u, v), we have that the maximum improvement found so far is greater than $\delta(S' \cup \{(u, v)\}, S)$, where S' is the solution found at iteration $i - 1$, then (u, v) cannot increase the value of $\sigma(A, S)$ more than the maximum found so far. Therefore, in this case we prune the search.

Live-edge graph reduction. At the end of each iteration of the loop at line 2 of Algorithm 4, we reduce the size of all the live-edge graphs by removing the nodes that become influenced when the edge $\hat{e} = (\hat{a}, \hat{u})$ found in the iteration is added to the solution. In this way the size of the residual live-edge graph is reduced after each iteration and hence the time required to compute $\tilde{\sigma}(A, S \cup \{e\})$ for each $e = (a, u) \in (A \times V) \setminus (E \cup S)$ is also decreased.

Low probability candidate edge pruning. The greedy algorithm needs to compute $\tilde{\sigma}(A, S \cup \{e\})$ for each $e = (a, u) \in (A \times V) \setminus (E \cup S)$ in order to find the edge that maximizes this quantity (see the loop at line 3 of Algorithm 4). However,

if for some nodes $a_1, a_2 \in A$ and $u \in V \setminus A$ we have that $p_{(a_1,u)} > p_{(a_2,u)}$, then $\sigma(A, S \cup \{a_1, u\}) > \sigma(A, S \cup \{a_2, u\})$. Therefore, for each node $u \in V \setminus A$ we can exclude from the loop of line 3, of Algorithm 4, edges incident to u with small probability. In particular, since it is unlikely that more than two edges towards the same node are added in the solution, we only evaluate the two incident candidate edges with the highest probability.

By means of experiments, we can show that even if we do not allow addition of edges from two, or more, different seeds $a_1, \ldots, a_i$ to the same node v_i, we do not affect the approximation guarantee.

We run the greedy algorithm on five randomly generated set of seeds choosing $|A| = 1\% \cdot |V|$, adding $|S| = 2 \times |A|$ edges and assigning probabilities according to the weighted model. In particular, we plot the results of the relative error for the network Wiki-Vote in Figure 4.2. In this graph, obtained from SNAP [50], nodes in the network represent Wikipedia users and a directed edge from node i to node j represents that user i voted on user j. It is easy to see that the two functions representing the number of influenced nodes coincide on most of the points.

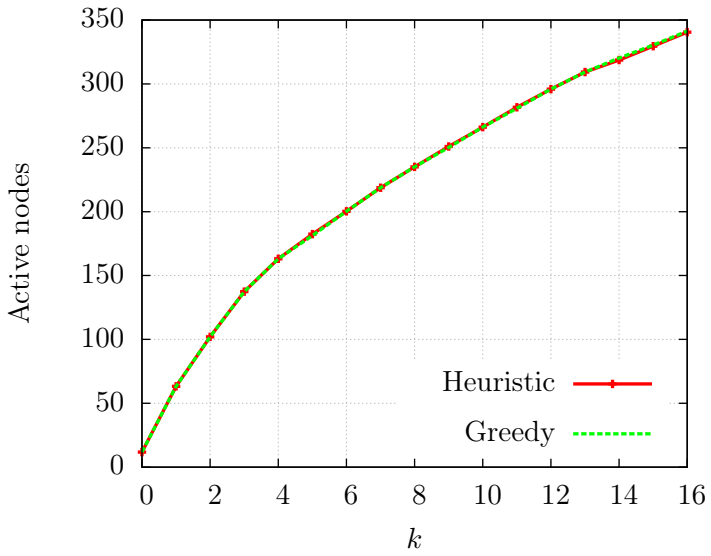

FIGURE 4.2: Approximation error for Wiki-Vote network.

The results for many other networks, taken from the ArnetMiner [2] repository, are similar and reported in Table 4.1: the first three columns report the name of the network, the number of nodes and the number of edges. The last column of the table reports the maximum error e_{max} among all the edge insertions and it is computed as

$$e_{max} = \left| \frac{\sigma(A, S_{all}) - \sigma(A, S)}{\sigma(A, S_{all})} \right|$$

where $\sigma(A, S_{all})$ is the expected number of active nodes computed using the greedy algorithm and allowing multiple edges to the same node while $\sigma(A, S)$ is that obtained adding only one outgoing edge per seed to the same node v_i.

Name	$n = \|V\|$	$m = \|E\|$	e_{max}
Software Engineering (SE)	3141	14787	0.045
Theoretical CS (TCS)	4172	14272	0.035
High-Performance Comp. (HPC)	4869	35036	0.033
Wiki-Vote (Wiki)	7115	103689	0.033
Computer Graphic (CGM)	8336	41925	0.045
Computer Networks (CN)	9420	53003	0.035

TABLE 4.1: Real-world networks and relative approximation error.

Hence, we can conclude that also the previous heuristic gives good approximations to the values computed by the greedy algorithm.

4.5 Limited seed selection problem

Even if the techniques described in the previous section improve the running time, it is still not enough for analysing real-world networks with millions of nodes and edges. Therefore, we show how to formulate the IM-L problem in terms of seeds selection problem in order to be able to adapt the efficient algorithms defined for the latter problem to solve ours.

We define the *Limited Seed Selection problem* (LSS) as follows: given a weighted directed graph with edge weights capturing influence probabilities, an integer k and a set of nodes $L \subseteq V$, the goal is to find a seed set of k users $A \subseteq L$ such that by targeting them the expected number of influenced users is maximized.

Limited Seed Selection problem (LSS)	
Given:	A weighted directed graph $G = (V, E, p, c)$; an integer $k \in \mathbb{N}$, a set of nodes $L \subseteq V$
Solution:	A set of seed nodes $A \subseteq L$ such that $c(A) \leq k$, where $c(A) = \|A\|$
Goal:	Maximize $\sigma(A)$

In what follows we describe how to modify a given graph $G = (V, E, p, c)$ with the purpose of showing that any instance $I_{\text{IM-L}}$ of the IM-L problem can be transformed into an instance I_{LLS} of the LSS problem and any solution S_{LSS} for I_{LLS} can be transformed into a solution $S_{\text{IM-L}}$ for $I_{\text{IM-L}}$.

Given $I_{\text{IM-L}} = (G, A, k)$, we define $I_{LSS} = (G', L, k)$ where $G' = (V', E')$ is the graph obtained from G by adding $|L| = |V \setminus A| \cdot |A|$ nodes and edges. More formally:

- $V' = V \cup L$,
- $E' = E \cup \bigcup_{\substack{v_i \in L \\ u_i \in V \setminus A}} \{(v_i, u_i)\}$.

In other words, for each node $u_i \in V \setminus A$ we add $|A|$ new nodes v_i and, for each of these nodes, a new incident edge (v_i, u_i). See Figure 4.3 for a visualization. The red nodes denote the seed nodes in A and the blue elements are the new nodes and edges, the dots stand for the rest of the graph G. The probability p_e is chosen accordingly to

the propagation model we are considering. We refer to the previous graph modification technique as $\mathcal{T}$.

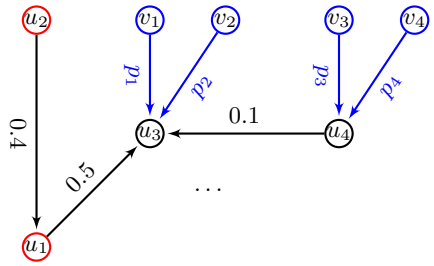

FIGURE 4.3: Grafo G' obtained using transformation $\mathcal{T}$.

Any solution for the instance of I_{LSS} given using the greedy algorithm is made of nodes $v_i \in L$. It is easy to see that each of these nodes corresponds to an edge (a_i, u_i) in the instance of $I_{\text{IM-L}}$ where a_i is the seed node such that $p_{(a_i,u_i)} = p_{(v_i,u_i)}$.

Theorem 10. *Given an instance $I_{\text{IM-L}} = (G, A, k)$ of the* IM-L *problem and an instance $I_{LLS} = (G', L, k)$ of the LSS problem, obtained modifying graph G in G' according to $\mathcal{T}$, then the value of the solution S_{LSS} of I_{LLS} is equal to the value of the solution $S_{\text{IM-L}}$ of $I_{\text{IM-L}}$.*

Proof. We focus on the solution $A' = A \cup \{v_i\}$ of I_{LSS}, by definition:

$$\sigma(A', E') = \sum_{X \in \chi(E')} \mathbb{P}[X]|R(A', X)|$$

and on the solution $S' = S \cup \{(a_i, u_i)\}$ for the instance $I_{\text{IM-L}}$:

$$\sigma(A, S') = \sum_{X \in \chi(S')} \mathbb{P}[X]|R(A, X)|$$

Is easy to see that for any live-edge graph generated from G there exist a live-edge graph generated from G' with the same probability and vice versa since both sets S' and E' have been generated with the same outcomes on the coin flips in the edges of E and same outcomes for edges in $E' \setminus E$ and in $S' \setminus E$.

Moreover, both graphs have can reach the same set of nodes $R(A', X)$ and $R(A, X)$, in particular: $R(A', X) = R(A, X)$ if the edge (v_i, u_i) is blocked and $R(A', X) = R(A, X) \cup R(u_i, X)$ if (v_i, u_i) is live, the same holds for $R(A, X)$ when we add (a_i, u_i). □

Thanks to Theorem 10, we can use efficient algorithms designed for IM to solve IM-L. In particular we adapt the algorithm presented in [20] since it allows us to find a set of seed nodes $A \subseteq L$ given a limited set of nodes L.

To further improve the running time, we do not generate $G' = (V', E')$ from G by adding $|L| = |V \setminus A| \cdot |A|$ nodes and edges, but we consider the heuristic where we add only $|L| = |V \setminus A|$ new elements. As we showed in the previous section, this does not affect the approximation guaranteed in practice.

In the following section we focus on the description of the algorithm designed in [20].

4.6 Sketch-based algorithm

The core of the Sketch-based influence maximization algorithm (SKIM) are the *combined reachability sketches* which are structures X_u associated to each node of the graph $G = (V, E, p, c)$.

Before defining these structures, we need some additional definitions. Let s be the number of live-edge graph samples used to approximate the value of $\sigma(A)$ where A is the initial set of seeds, i.e. $s = |\gamma|$ such that $\gamma \subseteq \chi$. The *reachability set* $R(u, X)$ of a node u in the instance $X \in \gamma$ is the set of nodes v reachable from u in X: $R(a, X) = \{v \mid a \rightsquigarrow v\}$, by $\rightsquigarrow$ we mean the existence of a path in X from u to v made entirely of live edges. The *combined reachability set* R_u is a set of node-instance pairs that combines the nodes reachable from u for all $X \in \gamma$, formally $R_u = \{(v, i) \mid u \rightsquigarrow_X v\}$.

According to these definitions, we can redefine $\sigma(A)$ as follows:

$$\sigma(A) = \frac{1}{s}\Big|\bigcup_{a \in A} R_a\Big|.$$

SKIM approximates $\sigma(A)$ using *combined reachability sketches* which are bottom-k min-hash sketches [18, 19] for estimating the size of combined reachability sets. Each node-instance pair (v, i) is associated with a random rank $r_v^i \sim U[0, 1]$ where $U[0, 1]$ is the uniform distribution over $[0, 1]$. Now we can define a combined reachability sketch of a node u, denoted by X_u, as the set of the k smallest rank values among $\{r_v^i \mid (v, i) \in R_u\}$.

Let the threshold rank τ_u of a node u be the k-th rank of X_u. Then, we can estimate the size of combined reachability set of a node u, namely $|R_u|$, as $\frac{k-1}{\tau_u}$ if $|X_u| = k$ and as $|X_u|$ if $|X_u| < k$.

The main idea behind SKIM it to compute combined reachability sketches, but only until the node with the maximum estimated influence is computed. This node is then added to the seed set. In detail, a random rank is assigned to all the node-instance pairs, then the algorithm processes them by increasing rank performing a Breadth First Search from the node u in the reverse graph instance X. For each scanned node, the reachability sketch is increased by one. For a given value k, the first node v whose sketch X_v reaches value k is selected as seed node and inserted in A.

Then the sketches are updated to be with respect to a residual problem in which the node that is selected into the seed set and the node it influences are no longer present. This process is performed using a BFS visit for each instance graph X. SKIM then resumes the sketch computation, starting with the residual sketches, but stopping again when a node with maximum estimated influence in the current, residual, instance is found. A new residual problem is then computed. This process is iterated until the seed set reaches the desired size.

We underline that, to estimate the influence of a node, SKIM samples a number of random nodes and utilizes the reverse graph to search which nodes can influence the sampled nodes. The main idea is that, in this process, influential nodes will likely

appear frequently in the reachability sets and so have a higher reachability sketch. An illustration is given in Figure 4.4. On the left we show the graph G, on the right we plot three live-edge graphs in $\gamma = \{X_1, X_2, X_3\}$ and the blue nodes are the sampled nodes. It is easy to see that 1 is the node with highest reachability sketch equal to three since it is reachable picking 2, 4 and 3; 3 has reachability sketch equal to one while 2 and 4 zero. Thus, 1 is the new seed node.

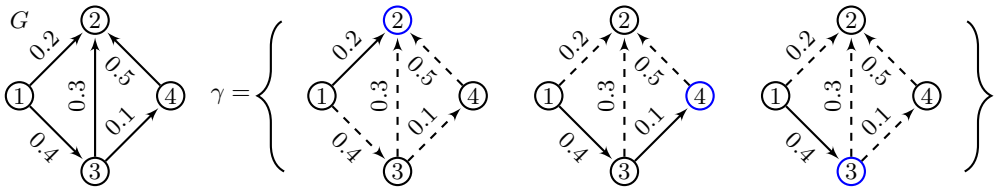

FIGURE 4.4: Example of reachability sketches.

The authors in [20] state that SKIM, given a set γ of s live-edge graphs, can be run to exhaustion, producing a full permutation of nodes in $O(m\epsilon^{-2}log^2(n) + \sum_{i=1}^{s} |E_{X_i}|)$ time, where m is the sum over nodes of the maximum indegree over all instances and $|E_{X_i}|$ is the number of live edges in the graph X_i. Moreover, SKIM computes a sequence of nodes $s \geq 1$ which influence is at least $1-(1-1/s)^s-\epsilon$ times the maximum influence of a seed set of the same size computed with the greedy algorithm, with a probability of at least $1 - \frac{1}{n^c}$, for a constant c.

4.7 Experimental results

In this section, we compare the performance of our algorithms by means of experiments. We focus on the greedy algorithm improved with the heuristics described in the previous section and on the SKIM algorithm applied to solve the LSS problem.

In detail, we compare the number of activated nodes in a graph augmented by using the greedy solution with the number of activated nodes in the original graph and in the graph augmented by using several alternative baselines. We show that the number of activated nodes highly increases with respect to the original graph and that the greedy solution outperforms all the other baselines.

All our experiments have been performed on a computer equipped with an AMD Opteron 6376 CPU with 16 cores clocked at 2.30GHz and 64GB of main memory. The programs have been coded in `C++` (gcc compiler v4.8.2 with optimization level O3).

We evaluate the performance of the algorithm on four types of randomly generated directed networks which exhibit many of the structural features of complex networks, namely directed Preferential Attachment (in short, `PA`) [10], Erdős-Rényi (`ER`) [31], Copying (`COPY`) [47], Compressible Web (`COMP`) [17] and Forest Fire (`FF`) [49]. For each combination (n, m), we generated five random directed graphs. We choose also different real-world graphs that are suitable for our problem, taken from the SNAP [50], KONECT [48], and ArnetMiner [2] repositories. The size of the graphs is reported in Table 4.2 and Table 4.3. For both synthetic and real-world networks, we choose 0.1%

of the nodes in V as seeds and we add up to $k = 2 \cdot |A|$ edges. The seed nodes are chosen uniformly at random, while the probability to edges is assigned according to the weighted model.

Name	$n = \|V\|$	$m = \|E\|$
Software Engineering (SE)	3,141	14,787
Theoretical CS (TCS)	4,172	14,272
High-Performance Comp. (HPC)	4,869	35,036
Wiki-Vote (Wiki)	7,115	103,689
Computer Graphic (CGM)	8,336	41,925
Computer Networks (CN)	9,420	53,003
Artificial Intelligence (AI)	27,617	268,460
Slashdot (Sl)	51,083	130,370
Epinions (Epi)	75,879	508,837
Slashdot-Zoo (Sl-z)	79,116	515,397
Digg	279,630	1,731,653
Citeseer	384,413	1,751,463
Twitter	465,017	834,797
Pokec	1,632,803	30,622,564

TABLE 4.2: Real-world networks.

Name	$n = \|V\|$	$m = \|E\|$
PA5	5000	6500
PA10	10000	13000
PA15	15000	20000
FF5	5000	10000
FF10	10000	20000
FF15	15000	30000
COPY5	5000	5000, 10000, 25000
COPY10	10000	20000, 50000, 100000
COPY15	15000	45000, 100000
COMP5	5000	5000, 10000, 25000
COMP10	10000	20000,50000, 100000
COMP15	15000	45000, 100000
ER5	5000	10000, 25000, 50000
ER10	10000	40000, 100000, 200000
ER15	150000	90000, 225000, 450000

TABLE 4.3: Random networks.

As a measure of the quality of the solution, we adopt the expected number of active nodes $\sigma(A, S)$. As discussed in Chapter 2, it has been proven that evaluating this function is generally $\#P$-complete for ICM [13]. However, by simulating the diffusion process sufficiently many times and sampling the resulting active sets, it is possible to obtain arbitrarily good approximations to $\sigma(A, S)$. We choose 500 as the number of samples to use for our experiments. Moreover, for the evaluation of all algorithms and baselines we run 500 trial to estimate the value of σ in the final solution.

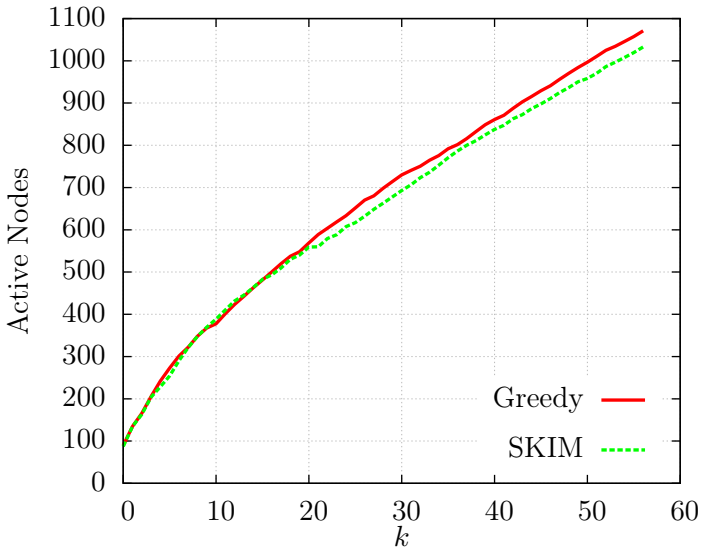

FIGURE 4.5: Comparison of the Greedy algorithm and Skim on AI network.

In Figure 4.5 we compare the result obtained applying the greedy algorithm and the Skim algorithm to the Artificial Intelligence network given a seed set A. The expected number of active nodes is similar for both algorithms and a small difference is due to the sampling technique used to estimate $\tilde{\sigma}(A, S)$. The results for the other real-world and random networks are reported in Tables 4.4 and Table 4.5, respectively. We use $***$ when the greedy algorithm did not compute the final solution for a given graph in the time limit that we set to four hours. In both tables, columns $\sigma(A, G)$ and $\sigma(A, G)\%$ report the absolute and relative initial number of infected nodes in the original graph G, respectively. Columns Greedy and SKIM report the results for the algorithms and they are divided into $\sigma(A, S)$, $\sigma(A, S)\%$, $I\%$ and time that represent the absolute number of infected nodes and relative number of infected nodes after the edge addition, the relative increment computed as $I = \frac{\sigma(A,S)-\sigma(A,G)}{\sigma(A,G)} \times 100$ and the computational time in seconds, respectively.

G	$\sigma(A,G)$	$\sigma(A,G)\%$	Greedy				SKIM			
			$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	time (sec.)	$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	time (sec.)
SE	13.38	0.43	103.09	3.28	670.56	5.74	103.45	3.29	670.95	0.04
TCS	9.49	0.23	97.54	2.34	928.26	10.44	97.63	2.34	928.76	0.07
HPC	9.36	0.19	164.92	3.39	1662.23	12.18	165.75	3.40	1670.83	0.07
Wiki	10.07	0.14	338.93	4.76	3266.84	35.31	333.37	4.69	3257.87	0.12
CGM	20.34	0.24	253.84	3.05	1148.13	35.88	257.62	3.09	1166.71	0.12
CN	22.21	0.24	408.69	4.34	1740.10	49.33	397.95	4.22	1765.86	0.10
AI	68.94	0.25	1055.76	3.82	1431.44	374.66	1017.82	3.69	1362.71	0.47
Sl	126.11	0.25	621.75	1.22	393.01	2050.65	612.52	1.20	408.63	1.72
Epi	352.17	0.46	1236.71	1.63	251.17	3884.27	1230.23	1.62	249.32	3.25
Sl-z	570.84	0.72	3485.02	4.41	510.50	3730.36	3425.87	4.40	505.14	2.63
Digg	3848.57	1.38	***	***	***	***	14835.40	5.31	285.48	14.32
Citeseer	683.16	0.18	***	***	***	***	13072.36	3.40	1813.52	12.20
Twitter	2861.20	0.62	***	***	***	***	207061.00	44.53	7136.87	10.23
Pokec	23472.20	1.44	***	***	***	***	83726.90	5.13	256.71	78.91

TABLE 4.4: Results for real-world networks.

G	$\sigma(A,G)$	$\sigma(A,G)\%$	Greedy				SKIM			
			$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	time (sec.)	$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	time (sec.)
PA5	7.30	0.17	333.37	7.67	4467.18	10.04	332.84	7.66	4448.94	0.03
PA10	15.05	0.17	688.57	7.84	4474.88	38.02	685.10	7.80	4474.17	0.07
PA15	20.51	0.16	1094.54	8.35	5237.70	86.93	1092.27	8.33	5230.44	0.12
FF5	10.07	0.20	137.94	2.76	1269.51	14.04	136.06	2.72	1246.09	0.07
FF10	19.08	0.19	276.40	2.76	1348.45	55.82	273.96	2.74	1343.34	0.15
FF15	28.34	0.19	432.76	2.89	1426.96	115.94	429.65	2.86	1420.92	0.25
COPY5-5	7.39	0.15	48.06	0.96	549.97	11.21	47.85	0.96	547.33	0.14
COPY5-10	9.12	0.18	81.71	1.63	795.56	13.61	80.43	1.61	783.16	0.09
COPY5-25	11.97	0.24	157.78	3.16	1218.38	12.45	154.85	3.10	1208.11	0.06
COPY10-20	17.99	0.18	167.47	1.67	830.95	51.26	164.63	1.65	809.46	0.21
COPY10-50	26.51	0.27	335.79	3.36	1166.53	48.53	329.07	3.29	1150.25	0.13
COPY10-100	31.98	0.32	559.43	5.59	1649.14	39.45	547.73	5.48	1608.47	0.11
COPY15-45	30.41	0.20	329.38	2.20	983.29	113.47	323.41	2.16	962.07	0.27
COPY15-100	40.40	0.27	630.23	4.20	1459.87	97.38	619.77	4.13	1446.42	0.20
COPY15-225	52.00	0.35	1128.13	7.52	2069.52	73.86	1102.94	7.35	2006.04	0.18
COMP5-5	9.39	0.19	62.43	1.25	564.61	15.14	61.92	1.24	562.24	0.11
COMP5-10	8.74	0.17	58.56	1.17	569.85	14.18	58.03	1.16	563.22	0.12
COMP5-25	7.97	0.16	55.81	1.12	600.02	12.74	54.46	1.09	583.92	0.12
COMP10-20	16.03	0.16	115.16	1.15	618.20	53.19	114.19	1.14	613.41	0.29
COMP10-50	16.85	0.17	112.70	1.13	568.72	48.59	109.95	1.10	554.42	0.29
COMP10-100	16.11	0.16	112.93	1.13	600.94	48.93	109.94	1.10	582.67	0.30
COMP15-45	24.79	0.17	170.61	1.14	588.17	114.11	167.55	1.12	573.56	0.46
COMP15-100	24.32	0.16	170.58	1.14	601.45	111.12	167.26	1.12	588.28	0.48
COMP15-225	24.82	0.17	171.82	1.15	592.36	107.07	169.09	1.13	582.21	0.50

TABLE 4.5: Results for random networks.

We compare the greedy algorithm with the alternative algorithms reported in what follows. The first three algorithms are well-known algorithms used for link recommendation.

- AdamicAdar (AA): connect the seeds to a set of k nodes with the highest Adamic-Adar index [1] computed for each edge (u, v) as $\sum_{z \in N_x \cap N_y} \frac{1}{log(|N_z|)}$;
- PrefAtt (PA): connect the seeds to a set of k nodes chosen according to the Preferential Attachment model [10, 60] and assigns to each edge a score $score(u, v) = |N_u| \cdot |N_v|$;
- Jaccard (J): connect the seeds to a set of k nodes with the highest Jaccard coefficient [38] defined as $\frac{|N_u \cap N_v|}{|N_u \cup N_v|}$;
- Degree (D): connect the seeds to a set of k nodes with the highest out-degree;
- Topk (TopK): connect the seeds to a set of k nodes with the highest harmonic centrality [9] $h(u) = \sum_{v \neq u} \frac{1}{d(u,v)}$ where $d(u, v)$ is the distance from node u to v;
- Prob (Prob): connect the seeds to a set of k nodes adding the arcs with highest probability;
- Seed (KKT): connect the seeds to a set of k nodes chosen as seeds by the greedy algorithm proposed by Kempe et al. [41];

by N_x we denote the set of outgoing neighbours of x, i.e. $N_x = \{v \mid (x, v) \in E\}$.

The nodes computed applying the previous baselines are connected to the given seeds set A adding the edges with highest probability according to the weighted model.

In Figure 4.6, we compare the greedy algorithm with the other approaches on the Slashdot-Zoo network. The plots show the average number of affected nodes as a function of the number of added edges. The experiments clearly show that the greedy algorithm outperforms all the alternative baselines, in terms of expected number of active nodes. Indeed, all the other competitive algorithms require to add a large number of edges to the set A of seeds in order to significantly increase the expected number of influenced nodes with respect to the initial value (see value at $k = 0$), whereas our algorithm increases $\sigma(A, S)$ by a greater amount with only few added edges. We observe that the number of affected nodes in the greedy solution is more than five times that of the original graph.

Figure 4.7 reports the average computational time, in seconds, of the greedy algorithm, on the top, and the computational time of SKIM, on the bottom, as a function of the number of added edges in the Slashdot-Zoo network. It is worth to note the impact of the heuristics on the running time of the greedy algorithm. The exploitation of the submodularity ensures that most of computational effort is paid at the first iteration (see $k = 1$) of the algorithm and it is negligible for the subsequent iterations (see $k > 1$). This is due to the fact that such heuristic is able to prune a large number of candidate edges after the first iteration.

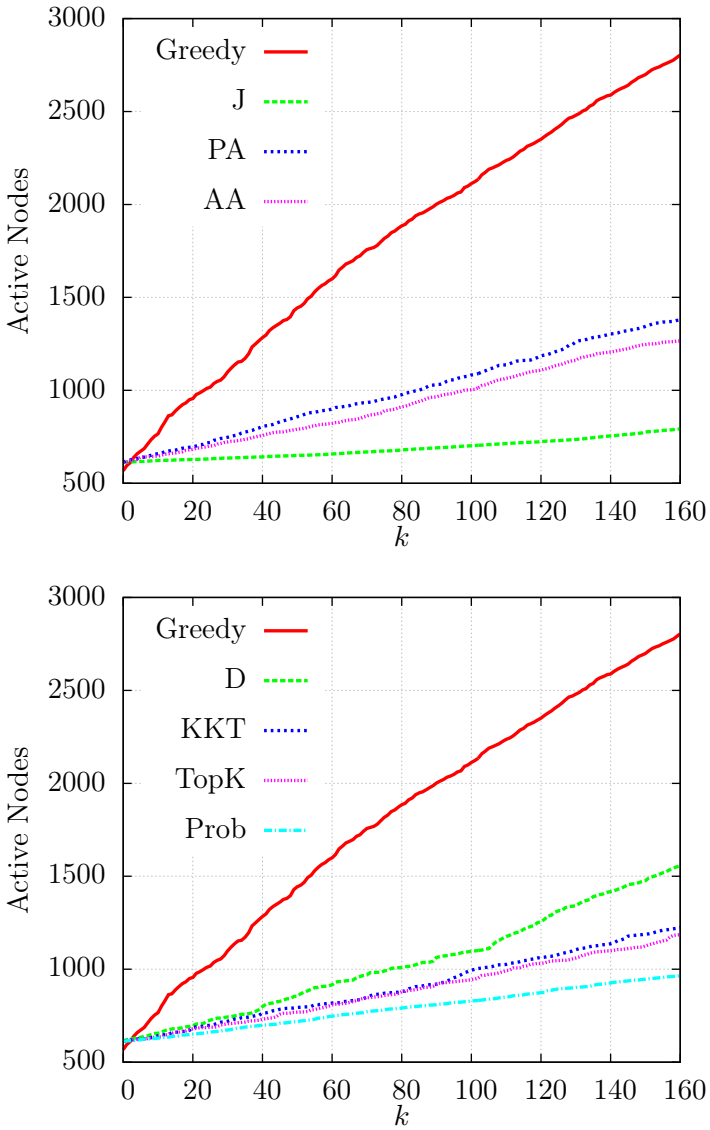

FIGURE 4.6: Results for the Slashdot-Zoo network.

The results of all the other networks are similar and they are reported in Table 4.6 and Table 4.7. For each graph we report the relative increment of active nodes computed as $I = \frac{\sigma_B(A,S)-\sigma(A,G)}{\sigma(A,G)} \times 100$, where $\sigma_B(A,S)$ is the expected number of active nodes computed using algorithm B and $k = 2 \cdot |A|$. It is clear from the results on the table that the number of nodes infected by using the greedy solution is significantly high and that the greedy algorithm outperforms all the other algorithms by order of magnitude.

It is worth to note that the second best algorithm is KKT, however we will show in Chapter 5 that this baseline does not perform well under the budget constraint. The main motivation of this performance is that, even if the nodes u computed by this baseline lead to a high number of influenced nodes, their activation depends on the activation probability of the edge (a, u) connecting a seed $a \in A$ to u.

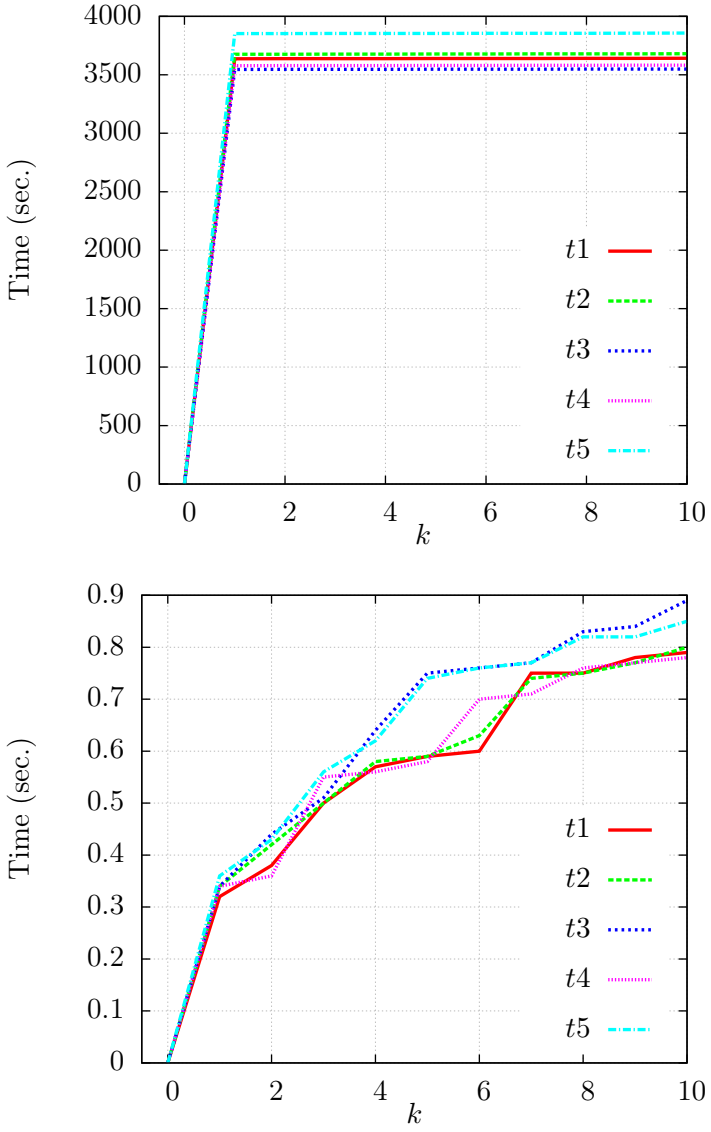

FIGURE 4.7: Computational time for the Slashdot-Zoo network.

G	$k=0$		SKIM	AA	PA	J	D	TopK	Prob	KKT
	$\sigma(A,G)$	$\sigma(A,G)\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$
SE	13.38	0.43	670.56	101.66	57.67	66.03	53.30	353.60	156.58	396.32
TCS	9.49	0.23	928.26	114.78	100.74	115.99	102.56	82.94	351.27	343.37
HPC	9.36	0.19	1662.23	176.33	15.33	153.93	16.15	15.99	363.73	896.97
Wiki	10.07	0.14	3266.84	330.02	1404.37	157.00	2078.44	2054.02	228.10	2284.13
CGM	20.34	0.24	1148.13	158.46	43.18	124.91	48.38	45.97	238.87	271.13
CN	22.21	0.24	1740.10	153.82	375.71	102.83	661.84	662.23	221.30	575.31
AI	68.94	0.25	1431.44	78.30	24.42	72.80	30.69	143.37	192.22	522.45
Sl	126.11	0.25	393.01	60.30	80.95	79.39	76.16	83.46	131.67	95.24
Epi	352.17	0.46	251.17	62.65	123.09	48.13	94.54	102.32	75.97	115.44
Sl-z	570.84	0.72	510.50	106.56	125.05	29.39	153.94	93.42	57.57	99.65
Digg	3848.57	1.38	285.48	126.80	149.44	10.58	145.44	121.71	39.07	86.65
Citeseer	683.16	0.18	1813.52	124.74	446.49	91.89	931.10	498.96	215.97	389.25
Twitter	2861.20	0.62	7136.87	1718.20	6286.58	5721.36	6510.76	6649.86	75.99	5148.94

TABLE 4.6: Baseline results for real-world networks.

G	$k=0$		SKIM	AA	PA	J	D	TopK	Prob	KKT
	$\sigma(A,G)$	$\sigma(A,G)\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$	$I\%$
PA5	7.30	0.17	4467.18	1996.92	2691.24	389.35	2877.88	3957.10	216.01	3900.63
PA10	15.05	0.17	4474.88	1785.58	3114.64	150.66	3183.03	3807.15	178.39	4030.14
PA15	20.51	0.16	5237.70	2587.99	3247.53	196.43	2049.33	4284.63	202.95	4079.94
FF5	10.07	0.20	1269.51	58.74	263.83	59.62	325.81	401.58	206.53	5079.94
FF10	19.08	0.19	1348.45	71.36	278.57	67.69	362.02	470.00	221.63	981.44
FF15	28.34	0.19	1426.96	63.68	322.10	68.29	425.73	515.21	225.89	929.24
COPY5-5	7.39	0.15	549.97	83.43	9.86	83.43	191.46	282.70	215.58	414.55
COPY5-10	9.12	0.18	795.56	43.29	3.70	50.57	168.26	477.72	206.25	537.71
COPY5-25	11.97	0.24	1218.38	52.57	0.72	51.10	137.23	785.68	246.65	990.36
COPY10-20	17.99	0.18	830.95	51.69	3.30	53.28	165.90	408.71	211.38	737.68
COPY10-50	26.51	0.27	1166.53	51.84	0.77	51.45	127.60	709.52	222.21	1038.17
COPY10-100	31.98	0.32	1649.14	55.77	0.29	51.15	76.79	788.38	317.94	1415.66
COPY15-45	30.41	0.20	983.29	53.59	1.84	52.69	172.67	567.75	225.81	880.44
COPY15-100	40.40	0.27	1459.87	45.39	0.54	41.39	109.73	995.94	297.03	1275.84
COPY15-225	52.00	0.35	1128.13	48.13	0.21	40.90	50.68	929.98	383.81	1073.52
COMP5-5	9.39	0.19	564.61	68.48	17	68.48	202.41	276.67	211.75	503.82
COMP5-10	8.74	0.17	569.85	60.91	2.83	60.91	186.71	268.85	217.59	530.11
COMP5-25	7.97	0.16	600.02	60.40	1.55	60.40	178.28	297.03	224.21	592.42
COMP10-20	16.03	0.16	618.20	61.62	3.86	61.62	195.86	254.23	221.14	589.87
COMP10-50	16.85	0.17	568.72	68.12	0.91	68.12	204.75	268.39	236.17	555.31
COMP10-100	16.11	0.16	600.94	68.44	1.29	68.44	185.98	272.92	227.66	568.24
COMP15-45	24.79	0.17	588.17	61.15	1.51	61.15	179.62	233.62	233.02	557.50
COMP15-100	24.32	0.16	601.45	57.70	1.35	57.70	178.75	249.61	228.20	597.35
COMP15-225	24.82	0.17	592.36	59.48	0.71	59.48	176.23	244.07	225.91	572.52

TABLE 4.7: Baseline results for random networks.

Chapter 5

Link addition problem with costs

In this chapter, we study the problem of *Budgeted Influence Maximization with Link addition*. We analyse the general cost version of the problem extending the results of the previous chapter.

5.1 Problem definition

The BIM-L problem is defined as follows: given a graph G, a budget k and a set A of seeds, find a set S of edges such that $S \subseteq (A \times V) \setminus E$, $c(S) \leq k$, and $\sigma(A, S)$ is maximum, where $c(S) = \sum_{e \in S} c_e$ and each arc $e \in (V \times V) \setminus E$ can be selected with cost $c_e \in [0, 1]$.

This problem introduces a more flexible and general model for the application to link recommendation even though the greedy heuristic of IM-L can not be trivially generalized to achieve the same approximation for the budgeted version of the problem. In the following, we will show how to improve the approximation factor to $1 - \frac{1}{e}$ using the enumeration technique.

We notice, following the reduction in Theorem 5, that also BIM-L is NP-hard to approximate within a constant factor grater then $1 - \frac{1}{2e}$.

5.2 Approximation algorithm

In this section we introduce our approximation algorithm for the BIM-L problem. First we propose a greedy algorithm that achieves an approximation factor of $\frac{1}{2}\left(1 - \frac{1}{e}\right)$, then we improve such approximation factor to $\left(1 - \frac{1}{e}\right)$ by using an enumeration technique.

Our algorithm, whose pseudocode is reported in Algorithm 5, outputs a solution that maximizes the expected number of affected nodes between two possible solutions. The first one is found at line 3 and is made of a single edge (a_M, v_M) for which $\sigma(A, S \cup \{(a_M, v_M)\})$ is maximized; the second solution is obtained by a greedy algorithm at lines 4–9.

In particular, the greedy phase, selects at each step an edge $\hat{e}$ to be added to the solution S obtained at the previous iteration, such that the ratio between $\delta(S \cup \{\hat{e}\}, S)$ and $c_{\hat{e}}$ is maximized. Then, if cost $c(S \cup \{\hat{e}\})$ does not violate the budget, the edge $\hat{e}$ is added to S, otherwise the edge is discarded.

Algorithm 5: Greedy BIM-L algorithm.

Input : A directed graph $G = (V, E, p, c)$, $k \in \mathbb{R}^+$; a seed set A
Output: A set of edges $S \subseteq (A \times V) \setminus E$ such that $c(S) \leq k$, where $c(S) = \sum_{e \in S} c_e$

```
S := ∅;
T := (A × V) \ E;
e_M := arg max_{e∈(A×V)\E} {σ(A, S ∪ {e})};
while T ≠ ∅ do
    ê := arg max_{e∈T} { δ(S∪{e},S) / c_e };
    if k − c_ê ≥ 0 then
        S := S ∪ {ê};
        k := k − c_ê;
    T := T \ {ê};
return arg max{σ(A, S), σ(A, {e_M})};
```

Next, we analyse the performance guaranteed by Algorithm 5. We denote by S^* an optimal solution to the problem. Let us consider the iterations i executed by the greedy algorithm, for $i \geq 1$, let us denote by j_i the index of such iterations, $j_i < j_{i+1}$. Let j_l be the index of iterations until an edge is added to the solution without exceeding the given budged and let j_{l+1} be the index of the first iteration in which an element in S^* is considered (i.e. it maximizes the above ratio) but not added to S because it violates the budget constraint. We denote by S_i the solution at the end of iteration j_i and by $\bar{c}_i$ the marginal cost of S_i as computed in the above ratio, $\bar{c}_i = c_{\hat{e}}$, where $\hat{e}$ is the edge selected at iteration i.

The next lemmas are the core of our analysis, note that the statements are similar to lemmas in [43]. In particular, Lemma 11 is applied to prove the base case of Lemma 12 which Lemma 12 gives us a lower bound for $\sigma(A, S)$. Finally we exploit these technical lemmas to prove the approximation ratio of our greedy algorithm.

Lemma 11. *After each iteration j_i, $i = 1, 2, \ldots, l+1$,*

$$\sigma(A, S_i) - \sigma(A, S_{i-1}) \geq \frac{\bar{c}_i}{k}(\sigma(A, S^*) - \sigma(A, S_{i-1})).$$

Proof. Firs we define δ_i to be the expected number of nodes affected by solution S_i and not affected by solution S_{i-1}, $\delta_i = \delta(S_i, S_{i-1})$.

It is easy to see that the following inequality holds

$$\sigma(A, S^*) - \sigma(A, S_{i-1}) \leq \sum_{e \in S^* \setminus S_{i-1}} \delta(S_{i-1} \cup \{e\}, S_{i-1}), \tag{5.1}$$

i.e. the value $\sigma(A, S^*) - \sigma(A, S_{i-1})$ is at most the sum, for each edge in S^* and not in S_{i-1}, of the expected number of nodes affected by such edge and not affected by solution (A, S_{i-1}).

Since the greedy algorithm selects at each step the element that maximizes the ratio between δ_i and $\bar{c}_i$, for each $e \in S^* \setminus S_{i-1}$ the following holds,

$$\frac{\delta(S_{i-1} \cup \{e\}, S_{i-1})}{c_e} \leq \frac{\delta_i}{\bar{c}_i}.$$

Therefore,

$$\sum_{e \in S^* \setminus S_{i-1}} \delta(S_{i-1} \cup \{e\}, S_{i-1}) \leq \sum_{e \in S^* \setminus S_{i-1}} \frac{\delta_i}{\bar{c}_i} c_e = \frac{\delta_i}{\bar{c}_i} \left(\sum_{e \in S^* \setminus S_{i-1}} c_e \right) \leq k \frac{\delta_i}{\bar{c}_i}.$$

To conclude the proof, we need to show that $\delta_i = \sigma(A, S_i) - \sigma(A, S_{i-1})$. Indeed, if $S_i \setminus S_{i-1} = \{e\}$,

$$\begin{aligned}
\delta_i &= \sum_{X \in \chi(S_i)} \mathbb{P}[X] \cdot (|R(A, X)| - |R(A, X^e)|) \\
&= \sigma(A, S_i) - \sum_{X \in \chi(S_{i-1})} (p_e |R(A, X)| + (1 - p_e)|R(A, X)|) \\
&= \sigma(A, S_i) - \sigma(A, S_{i-1}).
\end{aligned}$$

□

Armed with Lemma 11, we prove the next lemma by induction on iterations j_i.

Lemma 12. *After each iteration j_i, $i = 1, 2, \ldots, l+1$,*

$$\sigma(A, S_i) \geq \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \right) \right] \sigma(A, S^*).$$

Proof. For $i = 1$, form Lemma 11, $\sigma(A, S_1) \geq \frac{\bar{c}_1}{k} \sigma(A, S^*) = \left[1 - \left(1 - \frac{\bar{c}_1}{k} \right) \right] \sigma(A, S^*)$. Let us assume that the statement holds for $j_1, j_2, \ldots, j_{i-1}$, then

$$\begin{aligned}
\sigma(A, S_i) &= \sigma(A, S_{i-1}) + [\sigma(A, S_i) - \sigma(A, S_{i-1})] \\
&\geq \sigma(A, S_{i-1}) + \frac{\bar{c}_i}{k} [\sigma(A, S^*) - \sigma(A, S_{i-1})] \\
&= \sigma(A, S_{i-1}) \left(1 - \frac{\bar{c}_i}{k} \right) + \frac{\bar{c}_i}{k} \sigma(A, S^*)
\end{aligned}$$

where the inequalities follows from Lemma 11.

To conclude the proof we apply the inductive hypothesis:

$$\begin{aligned}
\sigma(A, S_{i-1}) \left(1 - \frac{\bar{c}_i}{k} \right) + \frac{\bar{c}_i}{k} \sigma(A, S^*) &\geq \left[1 - \prod_{\ell=1}^{i-1} \left(1 - \frac{\bar{c}_\ell}{k} \right) \right] \left(1 - \frac{\bar{c}_i}{k} \right) \sigma(A, S^*) + \frac{\bar{c}_i}{k} \sigma(A, S^*) \\
&= \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \right) \right] \sigma(A, S^*).
\end{aligned}$$

□

Theorem 13. *Algorithm 5 achieves an approximation factor of* $\frac{1}{2}\left(1-\frac{1}{e}\right)$ *for the* BIM-L *problem.*

Proof. We first observe two facts:

1. since (A, S_{l+1}) violates the budget, then $c(A, S_{l+1}) > k$,
2. for a sequence of numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = A$, the following holds: $\prod_{i=1}^{n}\left(1-\frac{a_i}{A}\right) \leq \left(1-\frac{1}{n}\right)^n$.

Therefore, by applying Lemma 12 for $i = l+1$ we obtain:

$$\begin{aligned}
\sigma(A, S_{l+1}) &\geq \left[1 - \prod_{\ell=1}^{l+1}\left(1-\frac{\bar{c}_\ell}{k}\right)\right]\sigma(A, S^*) \\
&\geq \left[1 - \prod_{\ell=1}^{l+1}\left(1-\frac{\bar{c}_\ell}{c(S_{l+1})}\right)\right]\sigma(A, S^*) \\
&\geq \left[1 - \left(1-\frac{1}{l+1}\right)^{l+1}\right]\sigma(A, S^*) \\
&\geq \left(1-\frac{1}{e}\right)\sigma(A, S^*).
\end{aligned}$$

It follows that:

$$\sigma(A, S_{l+1}) = \sigma(A, S_l) + \delta_{l+1} \geq \left(1-\frac{1}{e}\right)\sigma(A, S^*). \tag{5.2}$$

Moreover, since $\delta_{l+1} \leq \sigma(A, \{e_M\})$, we get:

$$\sigma(A, S_l) + \sigma(A, \{e_M\}) \geq \left(1-\frac{1}{e}\right)\sigma(A, S^*).$$

Finally, note that $\max\{\sigma(A, S_l), \sigma(A, \{e_M\})\} \geq \frac{1}{2}\left(1-\frac{1}{e}\right)\sigma(A, S^*)$. □

We now propose an algorithm which improves the performance guarantee of algorithm 5. Let M a fixed integer, we consider all the solutions of cardinality M (i.e. $|S| = M$) which have cost at most k, $c(S) \leq k$, and we complete each solution by using the greedy algorithm. The pseudocode is reported in Algorithm 6.

Theorem 14. *For* $M \geq 3$ *Algorithm 6 achieves an approximation factor of* $\left(1-\frac{1}{e}\right)$ *for the* BIM-L *problem.*

Proof. We assume that $|S^*| > k$ since otherwise algorithm 6 finds an optimal solution.

We sort the edges in S^* in decreasing order according to their marginal increment in the objective function.

Let S_Z be the first M elements in this order. We now consider the iteration of Algorithm 6 in which element Z is considered. We define $S_{Z'}$ such that $S = S_Z \cup S_{Z'}$, where S is the solution obtained after applying the greedy algorithm. It follows that:

$$\sigma(A, S) = \sigma(A, S_Z) + \delta(S_Z \cup S_{Z'}, S_Z).$$

Algorithm 6: Greedy IM-L algorithm with enumeration technique.

Input : A directed graph $G = (V, E, p, c)$, integer $M \in \mathbb{N}$ and an integer $k \in \mathbb{N}$
Output: A set of edges $S \subseteq (A \times V) \setminus E$ such that $c(S) \leq k$

```
S_1 := arg max{σ(A, S) : |S| < M, c(S) ≤ k};
S_2 := ∅;
T := (A × V) \ E;
foreach S ⊆ T such that |S| = M, c(S) ≤ k do
    T := T \ S;
    Complete S by using algorithm 5 with T as possible edge set;
    if σ(A, S) > σ(A, S_2) then
        S_2 := S;
return arg max{σ(A, S_1), σ(A, S_2)};
```

The completion of S_Z to S is an application of the greedy heuristic from Algorithm 5 therefore, we can use the result from the previous theorems. Let us consider the iterations executed by the greedy algorithm during the completion of S_Z to S. For $i \geq 1$, let us denote by j_i the index of such iterations, $j_i < j_{i+1}$, and let j_{l+1} be the index of the first iteration in which an edge in $S^* \setminus S_Z$ is considered but not added to $S_{Z'}$ because it violates the budget constraint. Applying Inequality (5.2), we get:

$$\delta(S_Z \cup S_{Z'}, S_Z) + \delta_{l+1} \geq \left(1 - \frac{1}{e}\right) \sigma(A, S^* \setminus S_Z) \tag{5.3}$$

we observe that, since we ordered the elements in S^*, $\delta_{l+1} \leq \frac{\sigma(A,S_Z)}{M}$.

Therefore, applying Inequality (5.3) and the previous observation:

$$\begin{aligned}
\sigma(A, S) &= \sigma(A, S_Z) + \delta(S_Z \cup S_{Z'}, S_Z) \\
&\geq \sigma(A, S_Z) + \left(1 - \frac{1}{e}\right) \sigma(A, S^* \setminus S_Z) - \delta_{l+1} \\
&\geq \sigma(A, S_Z) + \left(1 - \frac{1}{e}\right) \sigma(A, S^* \setminus S_Z) - \frac{\sigma(A, S_Z)}{M} \\
&\geq \left(1 - \frac{1}{M}\right) \sigma(A, S_Z) + \left(1 - \frac{1}{e}\right) \sigma(A, S^* \setminus S_Z)
\end{aligned}$$

But, $\sigma(A, S_Z) + \sigma(A, S^* \setminus S_Z) \geq \sigma(A, S^*)$, and we get:

$$\begin{aligned}
\sigma(A, S) &\geq \left(1 - \frac{1}{e}\right) \sigma(A, S^*) + \left(\frac{1}{e} - \frac{1}{M}\right) \sigma(A, S_Z) \\
&\geq \left(1 - \frac{1}{e}\right) \sigma(A, S^*), \quad \text{for } M \geq 3
\end{aligned}$$

proving the theorem. □

As in the previous section, Theorem 14 can be generalized to the case in which each step of the greedy algorithm uses repeated sampling to estimate a $(1+\lambda)$-approximation

of $\sigma(A, S)$ with probability $(1 - \delta)$. In this case, the greedy algorithm guarantees a $\left(1 - \frac{1}{e} - \epsilon\right)$-approximation, where $\epsilon \to 0$ as $\lambda \to 0$.

It is easy to see that Algorithm 6 is computationally very expensive in practice. Therefore we consider the techniques described in Chapter 4 to speed-up the algorithm but we are not able to adapt the sketch based algorithm for this case. For this reason, in the following section, we present our experimental results on random graphs and real-world networks of smaller dimension than those considered in Chapter 4.

5.3 Experimental results

In this set of experiments we evaluate the solution quality and the performance of the greedy algorithm.

As in Chapter 4, all our experiments have been performed on a computer equipped with an AMD Opteron 6376 CPU with 16 cores clocked at 2.30GHz and 64GB of main memory. The programs have been coded in `C++` (gcc compiler v4.8.2 with optimization level O3).

We evaluate the performance of the algorithm on five types of randomly generated directed networks. For each combination (n, m), we generated five random directed graphs, the size of the graphs is reported in Table 5.2. We choose also different real-world graphs that are suitable for our problem, taken from the ArnetMiner [2] repository. The size of the graphs is reported in Table 5.1. Notice that Table 5.1 and Table 5.2 contain a subset of graphs already described in Table 4.2 and Table 4.3 in the previous chapter. We report these information again in order to improve the readability of this section.

Name	$n = \|V\|$	$m = \|E\|$
Software Engineering (SE)	3141	14787
Theoretical CS (TCS)	4172	14272
High-Performance Comp. (HPC)	4869	35036
Computer Graphic (CGM)	8336	41925
Computer Networks (CN)	9420	53003
Artificial Intelligence (AI)	27617	268460

Table 5.1: Real-world networks.

We chose the 0.1% of nodes at random as the initial seed set A and we generated 5 random initial seed sets. We perform experiment using the weighted model fixing the budget to $k = 10$ and assigning to each edge $e = (u, v) \in V \times V$ a cost $c_e \in [0, 1]$ uniformly at random.

As a measure of the quality of the solution, we adopt the expected number of active nodes $\sigma(A, S)$. We choose 500 samples to approximate the value of the objective function for our experiments, while for the evaluation of all algorithms and baselines we run 500 trial to estimate the value of σ in the final solution.

In Figure 5.1 we plot the number of affected nodes as a function of the budget k for each of the five randomly generated seed set of nodes for the High-Performance

Name	$n = \|V\|$	$m = \|E\|$
PA5	5000	6500
PA10	10000	13000
PA15	15000	20000
FF5	5000	10000
FF10	10000	20000
FF15	15000	30000
COPY5	5000	5000, 10000, 25000
COPY10	10000	20000
COMP5	5000	5000, 10000, 25000
COMP10	10000	20000
ER5	5000	10000

TABLE 5.2: Random networks.

Computing network. The experiments clearly show that, using a small budget, our greedy algorithm significantly increases the expected number of influenced nodes with respect to the initial value $k = 0$. Indeed, starting with $\sigma(A, G) = 9.72$ influenced nodes we can reach $\sigma(A, S) = 589.86$ increasing the initial value of 5970.04%.

Our link recommender system succeeds in suggesting links that increase the content spreading capabilities of the given set of nodes using a small budget k.

Figure 5.2 reports the computational time of the greedy algorithm as a function of the budget k for each of the five randomly generated seed set of nodes for the High-Performance Computing network. Also in this general version of the IM-L problem, the exploitation of the submodularity ensures that most of computational effort is paid at the first iteration of the algorithm and it is smaller for the subsequent iterations. This is due to the fact that such heuristic is able to prune a large number of candidate edges after the first iteration. The other two heuristics, live-edge graph reduction and low probability candidate edge pruning, scale down the plots by a constant factor. We do not report the computational time of the greedy algorithm without heuristics because it is not able to run in reasonable time in the graphs of Table 5.1.

The results of all the other networks are similar and they are reported in Table 5.3 and Table 5.4. Columns $\sigma(A, G)$ and $\sigma(A, G)\%$ report the absolute and relative initial number of infected nodes, respectively. Column $k = 10$ reports the results for the greedy algorithm and it is divided into $\sigma(A, S)$, $\sigma(A, S)\%$, I, arcs and time that represent the absolute number of infected nodes, relative number of infected nodes, the relative increment, the number of edges added by the algorithm and the computational time, respectively.

The relative increment of active nodes is computed as

$$I = \frac{\sigma(A, S) - \sigma(A, G)}{\sigma(A, G)} \times 100.$$

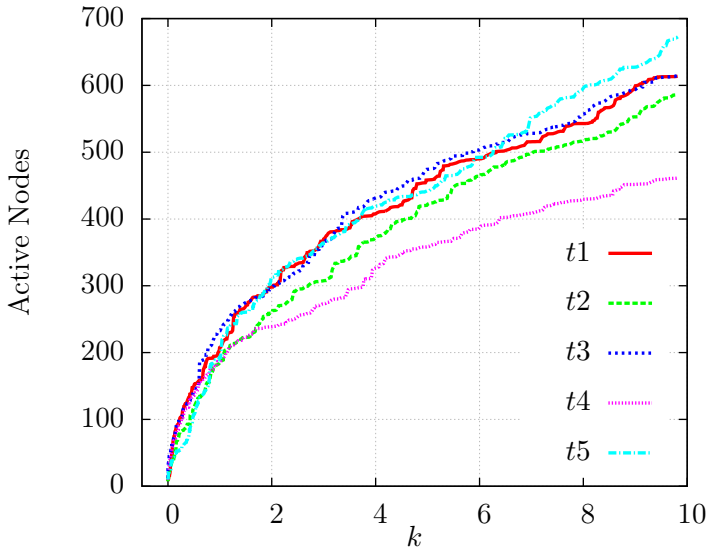

FIGURE 5.1: Number of affected nodes for the graph HPC for $k = 10$.

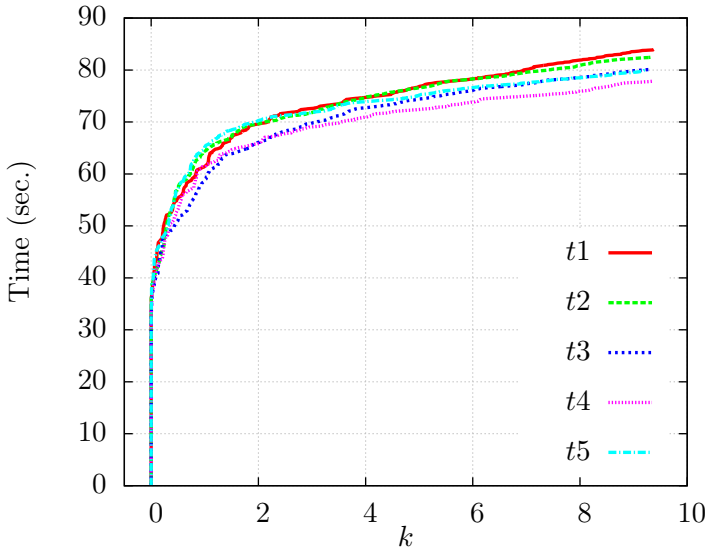

FIGURE 5.2: Running time in seconds for the graph HPC for $k = 10$.

G	$k=0$		$k=10$				
	$\sigma(A,G)$	$\sigma(A,G)\%$	$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	arcs	time (sec.)
SE	13.44	0.43	423.87	13.49	3053.27	347	31.53
TCS	9.48	0.23	560.91	13.44	5815.76	472	72.50
HPC	9.45	0.19	597.59	12.27	6223.36	488	80.36
CGM	20.56	0.25	1056.846	12.68	5041.34	852	409.44
CN	22.44	0.23	1205.82	12.80	5273.29	941	517.52
AI	68.64	0.24	2946.20	10.67	4192.31	2350	9839.6

Table 5.3: Greedy results for real-world networks.

G	$k=0$		$k=10$				
	$\sigma(A,G)$	$\sigma(A,G)\%$	$\sigma(A,S)$	$\sigma(A,S)\%$	$I\%$	arcs	time (sec.)
PA5	7.36	0.17	490.77	11.17	6857.57	375	105.24
PA10	15.10	0.17	854.90	9.75	5623.77	645	712.78
PA15	20.34	0.15	1110.01	8.45	5443.05	871	2334.66
FF5	10.08	0.20	783.14	15.66	7739.26	460	156.58
FF10	19.07	0.19	1458.52	14.59	7575.28	849	1081.38
FF15	28.28	0.19	2137.00	14.25	7477.24	1266	3495.22
COPY5-5	7.41	0.15	622.34	12.45	8306.54	483	131.83
COPY5-10	9.10	0.18	738.23	14.76	8035.53	467	150.06
COPY5-25	11.98	0.24	949.83	19	8231.65	417	137.39
COPY10	17.97	0.18	1362.06	13.62	7510.31	877	1058.92
COMP5-5	9.35	0.19	720.66	14.41	7645.89	484	185.02
COMP5-10	8.73	0.17	697.23	13.94	7928.81	472	160.62
COMP5-25	7.95	0.16	687.51	13.75	8707.00	461	148.09
COMP10	16.03	0.16	1285.73	12.86	7931.59	873	1134.45
ER5	29.48	0.60	1135.73	23.12	3752.27	536	168.06

Table 5.4: Greedy results for random networks.

Moreover we compare the greedy algorithm with the *Greedy-Seed* baseline in which we use all the given budget k to buy new seeds to be added to the initial set of seed nodes A. We assume that each seed has cost $c_a = 1$ as in the well-known Influence Maximization problem. In Figure 5.3 we plot the expected number of active node as a function of the budget for the HPC network for a given seed set. The red line represents the result obtained buying edges while the green one represents the nodes activated buying seeds according to the *Greedy-Seed* baseline. The experiments show that the greedy algorithm outperforms the baseline, indeed, the expected number of active nodes obtained buying new links is greater than that obtained buying new seeds despite the fact that the links get activated according to some probability and the seeds, when chosen, remain always active.

The result for the other networks are similar and we report them in Table 5.5 and Table 5.6. The column $k = 0$ reports two values: the number and the percentage of active nodes in the original graph; column $k = 0$ reports the relative increment I_E given buying edges and the relative increment I_S given buying new seeds. The relative increment I_S of active nodes is computed as $I_S = \frac{\sigma_B(A,S)-\sigma(A,G)}{\sigma(A,G)} \times 100$ where $\sigma_B(A,S)$ is the expected number of active nodes computed using the baseline.

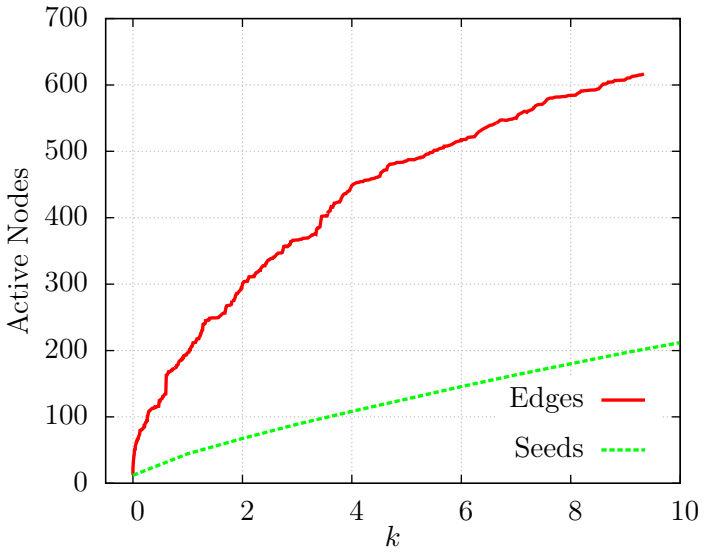

FIGURE 5.3: Number of affected nodes for the graph HPC for $k = 10$.

Hence, it is worth to design more efficient algorithms to solve the BIM-L problem in order to be able to analyse larger real-worlds networks.

As a future work, it would be interesting to analyse the BIM-L problem assigning costs to edges using different distributions and also to learn the edge costs from real scenarios.

G	$k = 0$		$k = 10$	
	$\sigma(A, G)$	$\sigma(A, G)\%$	$I_E\%$	$I_S\%$
SE	13.44	0.43	3053.27	1000.49
TCS	9.48	0.23	5815.76	1191.30
HPC	9.45	0.19	6223.36	2129.05
CGM	20.56	0.25	5041.34	1138.64
CN	22.44	0.23	5273.29	1375.22
AI	68.64	0.24	4192.31	662.75

TABLE 5.5: Greedy results for real-world networks.

G	$k = 0$		$k = 10$	
	$\sigma(A, G)$	$\sigma(A, G)\%$	$I_E\%$	$I_S\%$
PA5	7.36	0.17	6857.57	5092.28
PA10	15.10	0.17	5623.77	3855.90
PA15	20.34	0.15	5443.05	3569.45
FF5	10.08	0.20	7739.26	2142.55
FF10	19.07	0.19	7575.28	1368.87
FF15	28.29	0.19	7477.24	1105.97
COPY5-5	7.41	0.15	8306.54	1052.40
COPY5-10	9.10	0.18	8035.53	1457.38
COPY5-25	11.98	0.24	8231.65	2259.74
COPY10	17.97	0.18	7510.31	834.39
COMP5-5	9.35	0.19	7645.89	1070.67
COMP5-10	8.73	0.17	7928.81	1079.00
COMP5-25	7.95	0.16	8707.00	1163.13
COMP10	16.03	0.16	7931.59	618.96
ER5	29.48	0.60	3752.27	563.64

TABLE 5.6: Greedy results for random networks.

Chapter 6

Budgeted seed selection and link addition problem

In this chapter, we investigate a problem which is more general and challenging than the previous two, we refer to that as the *Budgeted Influence Maximization with Seeds selection and Link addition problem* (BIM-SL).

We observe that the BIM-SL problem cannot be approximated within a factor greater than $1 - \frac{1}{e}$, unless $P = NP$, because it is a generalization of the problem proposed in [40]. We then focus on approximation algorithms. First, in Section 6.2, we focus on the case in which $k = O(1)$ and provide an algorithm that guarantees an approximation factor of $1 - \frac{1}{e}$. Moreover, we show that this result is the best achievable one by a polynomial time algorithm, unless $P = NP$.

We then focus on the case in which k is not a constant. In Section 6.3, we propose two approximation algorithms that guarantee approximation factors of $1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}$ and $\left(1 - \frac{1}{e}\right) c_{\min}$, respectively. For both of them, we assume that the edge costs are all greater that a given constant $c_{min} \leq 1$. In Section 6.4, we focus on the general case of arbitrary small values of c_e and propose an algorithm that guarantees a constant approximation ratio of about 0.0878. This algorithm outperforms the other two algorithms when the cost is allowed to be small.

6.1 Problem definition

The BIM-SL problem is the most general problem we study since it includes not only edges addition, but also seeds selection. In particular, we look for a set of seeds A and a set of edges S, to be added to G, incident to these seeds that maximize $\sigma(A, S)$. W.l.o.g. we assume that each seed node can be selected with cost 1, while each edge $e \in (V \times V) \setminus E$ can be selected with cost $c_e \in [0, 1]$. More formally the BIM-SL problem is defined as follows: given a graph G and a budget k, find a set A of seeds and a set S of edges such that $S \subseteq (A \times V) \setminus E$, $c(A, S) \leq k$, and $\sigma(A, S)$ is maximum, where $c(A, S) = |A| + \sum_{e \in S} c_e$.

Algorithm 7: Algorithm for BIM-SL with $k = O(1)$

Input : A directed graph $G = (V, E, p, c)$ and an integer $k = O(1)$
Output: A set of nodes A and a of edges $S \subseteq (A \times V) \setminus E$ such that $c(A, S) \leq k$

```
(A_M, S_M) := (∅, ∅);
foreach A ⊆ V such that |A| ≤ k do
    Let S be the set of edges returned by the Algorithm 6 on nodes A, with
    budget k − |A|;
    if σ(A, S) > σ(A_M, S_M) then
        (A_M, S_M) := (A, S);
return (A_M, S_M);
```

6.2 Constant budget

In this section we focus on the case in which $k = O(1)$. We give a simple algorithm, reported in Algorithm 7, that guarantees an approximation factor of $1 - \frac{1}{e}$ and we show that this is the best approximation ratio that a polynomial time algorithm can achieve, unless $P = NP$. Besides being of its own interest, the results in this section allows us to focus on the case in which k is not a constant (see Sections 6.3 and 6.4).

Our solution exploits, as a subroutine, the algorithm to solve the BIM-L problem [27] defined in Chapter 5. We showed that BIM-L can be approximated within a factor of $1 - \frac{1}{e}$ by using a greedy algorithm.

In detail, Algorithm 7 considers each set of nodes A with cardinality at most k and runs Algorithm 6 for the BIM-L problem, by using A as set of seed nodes and $k - |A|$ as budget. The algorithm returns a set S of edges incident to nodes in A such that the cost of S is at most $k - |A|$ and $\sigma(A, S) \geq \left(1 - \frac{1}{e}\right) \sigma(A, S')$, where S' is the set that maximizes $\sigma(A, S')$, where A is given in input.

Since $k = O(1)$, then there are at most $n^{O(1)}$ possible sets of nodes with cardinality at most k and then the algorithm requires a polynomial computational time.

Theorem 15. *Algorithm 7 achieves an approximation factor of* $1 - \frac{1}{e}$.

Proof. Let (A^*, S^*) be an optimal solution for BIM-SL and let us consider the iteration of Algorithm 7 that selects set A^*. The set S returned by the algorithm in [27] returns a set S such that $\sigma(A^*, S) \geq \left(1 - \frac{1}{e}\right) \sigma(A^*, S')$, where S' is an optimal solution for the BIM-L instance with A^* as a seed set and $k - |A^*|$ as budget, that is S' maximizes $\sigma(A^*, S')$. We claim that $\sigma(A^*, S') = \sigma(A^*, S^*)$. Indeed, if there exists a set S' for which $\sigma(A^*, S') > \sigma(A^*, S^*)$, then (A^*, S^*) would not be an optimal solution for BIM-SL. □

The next theorem shows that the approximation factor achieved by Algorithm 7 is tight.

Theorem 16. *It is NP-hard to approximate* BIM-SL *to within a factor greater than* $1 - \frac{1}{e}$, *even if* $k = O(1)$.

Proof. Let us consider the Maximum Coverage problem in which we are given a ground set $X = \{x_i, x_2, \ldots, x_n\}$, a collection of subsets $S_1, S_2, \ldots, S_m$ of X, and a budget B,

and we want to select B subsets in order to maximize the size of their union. We can assume, without loss of generality, that $B < n < m$.

We construct an instance of BIM-SL as follows. We have a node i, for each subset S_i, n nodes $j_1, j_2, \ldots, j_n$, for each element x_j in X, and a further node v. There is a directed edge from i to node j_l, for all $l = j_1, j_2, \ldots, j_n$, whenever $x_j \in S_i$. We set the budget k equal to 2, and the probability on the edges equal to 1 for all pair of nodes in G. The cost of each pair of nodes is set to 1, except for pairs (v, i), $i = 1, 2, \ldots, m$, for which $c_{(v,i)} = \frac{1}{B}$.

Let us consider a solution to the maximum coverage problem T that covers $Y \subseteq X$ elements of X, then a solution made of node v and the edges connecting v to the nodes corresponding to the sets in T, costs 2 and activates $1 + B + |Y|n$ nodes.

Conversely, let us consider a solution to the BIM-SL instance and let us assume, without loss of generality, that this solution consists of node v and a set E' of B edges outgoing v. Indeed, any other feasible solution can be converted into a feasible solution with this shape and a grater number of active nodes. Let T be the sets corresponding to the tail nodes of the edges in E' and let Y be the elements of X covered by the union of the sets in T. The number of activated nodes in the BIM-SL instance is $1 + B + |Y|n$.

We have shown that, in any instance of the maximum coverage problem with n elements and budget B, we can cover M elements if and only if there exists an instance of the BIM-SL problem with budget $k = 2$ in which we can activate $1 + B + Mn$ nodes. We can assume that $B < M$, then $1 + B + Mn \leq M(n + 1)$. Moreover, $1 + B + Mn > Mn$. This implies that any α approximation algorithm for BIM-SL corresponds to an $\alpha - \frac{1}{n}$ approximation algorithm for the maximum coverage problem. Since this latter is NP-hard to approximate BIM-SL to within a factor greater than $1 - \frac{1}{e}$, we obtain the statement. □

In the following sections, we focus on the case in which k is not a constant and we provide two approximation algorithms.

6.3 Algorithm for lower bounded edge costs

In this section we consider the case in which the edge costs are at least a given value $c_{\min} \leq 1$, that is for each $e \in V \times V, c_e \geq c_{\min}$.

We first observe that in this case selecting a set A of k seed nodes that maximizes $\sigma(A, \emptyset)$ guarantees an approximation factor of $c_{\min}$. Therefore, we can obtain an overall $\left(1 - \frac{1}{e}\right) c_{\min}$ approximation since the seed selection problem can be optimally approximated within $1 - \frac{1}{e}$ as shown in [41].

In what follows we investigate the case where $c_{\min}$ has a small value and we give an approximation algorithm that improves over the previous bound.

We notice that a greedy algorithm, like those in [**DSV16**, 24], that at each iteration selects the seed or the edge that maximizes the expected number of affected nodes is

not suitable for the BIM-SL problem because it does not take into account the cost of the added edges.

Algorithm 8: Greedy algorithm for BIM-SL with lower bounded edge costs.

Input : A directed graph $G = (V, E, p, c)$ and $k \in \mathbb{R}^+$
Output: A set of nodes A and a of edges $S \subseteq (A \times V) \setminus E$ such that $c(A, S) \leq k$, where $c(A, S) = |A| + \sum_{e \in S} c_e$

1 $A := \emptyset$; $S := \emptyset$; $U := V$; $T := (V \times V) \setminus E$;
2 **while** $T \neq \emptyset$ *or* $U \neq \emptyset$ **do**
3 $\quad r_1 = \max_{a \in U} \{\delta(A \cup \{a\}, S, A, S)\}$;
4 $\quad r_2 = \max_{(a,v) \in (A \times V) \cap T} \left\{ \frac{\delta(A, S \cup \{(a,v)\}, A, S)}{c_{(a,v)}} \right\}$;
5 $\quad r_3 = \max_{a \in U, (a,v) \in (\{a\} \times V) \cap T} \left\{ \frac{\delta(A \cup \{a\}, S \cup \{(a,v)\}, A, S)}{(1 + c_{(a,v)})} \right\}$;
6 $\quad$ **if** $\max\{r_1, r_2, r_3\} = r_1$ **then**
7 $\quad\quad \hat{a} = \arg\max_{a \in U} \{\delta(A \cup \{a\}, S, A, S)\}$;
8 $\quad\quad$ **if** $k - 1 \geq 0$ **then**
9 $\quad\quad\quad A := A \cup \{\hat{a}\}$;
10 $\quad\quad\quad k := k - 1$;
11 $\quad\quad U := U \setminus \{\hat{a}\}$;
12 $\quad$ **else**
13 $\quad\quad$ **if** $\max\{r_1, r_2, r_3\} = r_2$ **then**
14 $\quad\quad\quad (\hat{a}, \hat{v}) := \arg\max_{(a,v) \in (A \times V) \cap T} \left\{ \frac{\delta(A, S \cup \{(a,v)\}, A, S)}{c_{(a,v)}} \right\}$;
15 $\quad\quad\quad$ **if** $k - c_{(\hat{a},\hat{v})} \geq 0$ **then**
16 $\quad\quad\quad\quad S := S \cup \{(\hat{a}, \hat{v})\}$;
17 $\quad\quad\quad\quad k := k - c_{(\hat{a},\hat{v})}$;
18 $\quad\quad\quad T := T \setminus \{(\hat{a}, \hat{v})\}$;
19 $\quad\quad$ **else**
20 $\quad\quad\quad (\hat{a}, (\hat{a}, \hat{v})) := \arg\max_{a \in U, (a,v) \in (\{a\} \times V) \cap T} \left\{ \frac{\delta(A \cup \{a\}, S \cup \{(a,v)\}, A, S)}{(1 + c_{(a,v)})} \right\}$;
21 $\quad\quad\quad$ **if** $k - c_{(\hat{a},\hat{v})} - 1 \geq 0$ **then**
22 $\quad\quad\quad\quad (A, S) := (A \cup \{\hat{a}\}, S \cup \{(\hat{a}, \hat{v})\})$;
23 $\quad\quad\quad\quad U := U \setminus \{\hat{a}\}$;
24 $\quad\quad\quad\quad k := k - 1 - c_{(\hat{a},\hat{v})}$;
25 $\quad\quad\quad T := T \setminus \{(\hat{a}, \hat{v})\}$;
26 $(a_M, (a_M, v_M)) := \arg\max_{a \in V, (a,v) \in (\{a\} \times V) \setminus E} \{\sigma(A \cup \{a\}, S \cup \{(a,v)\})\}$;
27 **return** $\arg\max\{\sigma(A, S), \sigma(\{a_M\}, \{(a_M, v_M)\})\}$;

Our algorithm, whose pseudocode is reported in Algorithm 8, finds two candidate solutions: the first solution is obtained by a greedy algorithm at lines 2–25, the second solution is found at line 26 and is made of a single node a_M and a single edge (a_M, v_M) for which $\sigma(\{a_M\}, \{(a_M, v_M)\})$ is maximized. Then, the algorithm outputs the solution that maximizes the expected number of affected nodes (line 27).

At each iteration of the greedy phase, the algorithm builds a solution (A, S) from the current one (A', S') in three different ways: adding one node, adding one node and one edge or adding a single edge. The greedy algorithm selects the solution (A, S) that maximizes the ratio between $\delta(A, S, A', S')$ and the marginal cost of (A, S).

In details, the three alternatives to obtain (A, S) from (A', S') are:

line 3: to select a seed node a that maximizes $r_1 = \delta(A' \cup \{a\}, S', A', S')$, $(A, S) = (A' \cup \{a\}, S')$;

line 4: to select an edge (a, v) incident to a seed a in A' that maximizes $r_2 = \frac{\delta(A', S' \cup \{(a,v)\}, A', S')}{c_{(a,v)}}$, $(A, S) = (A', S' \cup \{(a, v)\})$;

line 5: to select a seed node a not in A' and edge (a, v) incident to a that maximize $r_3 = \frac{\delta(A' \cup \{a\}, S' \cup \{(a,v)\}, A', S')}{1+c_{(a,v)}}$, $(A, S) = (A' \cup \{a\}, S' \cup \{(a, v)\})$.

If (A, S) does not violate the budget, i.e. cost $c(A, S)$ is at most k, then it is chosen as new solution.

We denote by (A^*, S^*) an optimal solution to the BIM-SL problem. Let us consider the iterations executed by the greedy algorithm in which an element is added to (A, S). For $i \geq 1$, let us denote by j_i the index of these iterations, $j_i < j_{i+1}$, and let j_{l+1} be the index of the first iteration in which an element in (A^*, S^*) is considered (i.e. it maximizes the above ratios) but not added to (A, S) because it violates the budget constraint. We denote by (A_i, S_i) the solution at the end of iteration j_i and by $\bar{c}_i$ the marginal cost of (A_i, S_i) as computed in the above three ratios,

$$\bar{c}_i = \begin{cases} 1 & \text{if } A_i \setminus A_{i-1} = \{a\} \text{ and } S_i = S_{i-1} \qquad (\max\{r_1, r_2, r_3\} = r_1) \\ c_e & \text{if } A_i = A_{i-1} \text{ and } S_i \setminus S_{i-1} = \{(a, v)\} \qquad (\max\{r_1, r_2, r_3\} = r_2) \\ 1 + c_e & \text{if } A \setminus A_{i-1} = \{a\} \text{ and } S_i \setminus S_{i-1} = \{(a, v)\} \ (\max\{r_1, r_2, r_3\} = r_3) \end{cases}$$

The next two lemmas are the core of our analysis. The statements are similar to Lemma 1 and Lemma 2 in [43] but we modify them in order to be applied to our setting. We develop new techniques which take into account that, after each iteration of the algorithm, the probability space of the solution changes.

In particular, Lemma 17 is applied to prove the base case of Lemma 18 which gives us a lower bound for $\sigma(A, S)$. Finally we exploit these technical lemmas to prove the approximation ratio of our greedy algorithm.

Lemma 17. *After each iteration j_i, $i = 1, 2, \ldots, l + 1$,*

$$\sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1}) \geq \frac{\bar{c}_i}{k} \frac{c_{\min}}{1 + c_{\min}} (\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})).$$

Proof. We denote by δ_i the expected number of nodes affected by solution (A_i, S_i) and not affected by solution (A_{i-1}, S_{i-1}), $\delta_i = \delta(A_i, S_i, A_{i-1}, S_{i-1})$. We first show that the value $\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})$ is at most the sum, for each element in (A^*, S^*) and not in (A_{i-1}, S_{i-1}), of the expected number of nodes affected by this element and not

affected by solution (A_{i-1}, S_{i-1}), that is the following inequality holds:

$$\begin{aligned}\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1}) \leq & \sum_{a \in A_1^*} \delta(A_{i-1} \cup \{a\}, S_{i-1}, A_{i-1}, S_{i-1}) + \\ & \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} \delta(A_{i-1}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1}) + \\ & \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1}}} \delta(A_{i-1} \cup \{a\}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1}),\end{aligned} \tag{6.1}$$

where we divided the set $A^* \setminus A_{i-1}$ into two subsets: A_1^* is the subset of $A^* \setminus A_{i-1}$ that contains the seeds a with no incident edges in S^* (i.e. $\nexists (a, v) \in S^*$),

and $A_2^* = A^* \setminus (A_{i-1} \cup A_1^*)$. The difference $\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})$ is at most

$$\sigma(A^* \cup A_{i-1}, S^* \cup S_{i-1}) - \sigma(A_{i-1}, S_{i-1})$$

and therefore upper-bounded by:

$$\begin{aligned}& \sum_{X \in \chi(S^* \cup S_{i-1})} \mathbb{P}[X] |R(A^* \cup A_{i-1}, X)| - \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] |R(A_{i-1}, X)| \\ &= \sum_{X \in \chi(S_{i-1})} \sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y] |R(A^* \cup A_{i-1}, Y)| - \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] |R(A_{i-1}, X)| \\ &= \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] \sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] |R(A^* \cup A_{i-1}, Y)| - \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] |R(A_{i-1}, X)| \\ &= \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] \left[\sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] |R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)| \right] \\ &= \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] \left[\sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] |R(A^* \cup A_{i-1}, Y)| - \sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] |R(A_{i-1}, X)| \right] \\ &= \sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] \sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] \left(|R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)| \right)\end{aligned} \tag{6.2}$$

For each $X \in \chi(S_{i-1})$ and $Y \in \chi(S^* \cup S_{i-1}, X)$, the difference $|R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)|$ between the nodes reachable from $A^* \cup A_{i-1}$ in Y and those reachable from A_{i-1} in X can be bounded as follows:

$$\begin{aligned}|R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)| \leq & \sum_{a \in A_1^*} \left(|R(A_{i-1} \cup \{a\}, X)| - |R(A_{i-1}, X)| \right) \\ & + \sum_{\substack{e=(a,v) \in Y \setminus X \\ \text{s.t. } a \in A_{i-1}}} \left(|R(A_{i-1}, X \cup \{e\})| - |R(A_{i-1}, X)| \right) \\ & + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in Y \setminus X}} \left(|R(A_{i-1} \cup \{a\}, X \cup \{e\})| - |R(A_{i-1}, X)| \right).\end{aligned} \tag{6.3}$$

Combining (6.2) and the first term of (6.3), we obtain the first term of (6.1). To show the second and third term of (6.1), observe that for a function $f : V \times V \to \mathbb{N}$ and for each $X \in \chi(S_{i-1})$,

$$\sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X] \sum_{e \in Y \setminus X} f(e) \leq \sum_{e \in S^* \setminus S_{i-1}} p_e \sum_{Y \in \chi(S^* \cup S_{i-1} \setminus \{e\}, X)} \mathbb{P}[Y|X \cup \{e\}] f(e) = \sum_{e \in S^* \setminus S_{i-1}} p_e f(e).$$

This shows inequality (6.1).

Since the greedy phase of the algorithm selects a solution that maximizes the ratio between the marginal increment in the objective function and the cost, the following holds:

- For each $a \in A_1^*$, $\delta(A_{i-1} \cup \{a\}, S_{i-1}, A_{i-1}, S_{i-1}) \leq \frac{\delta_i}{\bar{c}_i}$;
- For each $e = (a, v) \in S^* \setminus S_{i-1}$ such that $a \in A_{i-1}$, $\frac{\delta(A_{i-1}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1})}{c_e} \leq \frac{\delta_i}{\bar{c}_i}$;
- For each $a \in A_2^*$ and $e = (a, v) \in S^* \setminus S_{i-1}$, $\frac{\delta(A_{i-1} \cup \{a\} S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1})}{1 + c_e} \leq \frac{\delta_i}{\bar{c}_i}$.

Since the edge costs are at least $c_{\min}$, then the number of edges in $S^* \setminus S_{i-1}$ incident to each $a \in A_2^*$ is at most $\left\lfloor \frac{k}{c_{\min}} \right\rfloor \leq \frac{k}{c_{\min}}$. Therefore, the right hand side of (6.1) is at most:

$$\sum_{a \in A_1^*} \frac{\delta_i}{\bar{c}_i} + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} \frac{\delta_i}{\bar{c}_i} c_e + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1}}} \frac{\delta_i}{\bar{c}_i}(1 + c_e)$$

$$\leq \frac{\delta_i}{\bar{c}_i} \left(|A_1^*| + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} c_e + \frac{k}{c_{\min}} + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1}}} c_e \right) \leq \left(1 + \frac{1}{c_{\min}}\right) k \frac{\delta_i}{\bar{c}_i}.$$

Where the last inequality is due to

$$|A_1^*| + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} c_e + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1}}} c_e \leq k.$$

To conclude the proof, $\delta_i = \sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1})$ follows from Proposition 1. □

Lemma 18. *After each iteration j_i, $i = 1, 2, \ldots, l+1$,*

$$\sigma(A_i, S_i) \geq \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \frac{c_{\min}}{(1 + c_{\min})}\right)\right] \sigma(A^*, S^*).$$

Proof. We show the statement by induction on iterations j_i. For $i = 1$, by Lemma 17, $\sigma(A_1, S_1) \geq \frac{\bar{c}_1}{k} \frac{c_{\min}}{(1+c_{\min})} \sigma(A^*, S^*) = \left[1 - \left(1 - \frac{\bar{c}_1}{k} \frac{c_{\min}}{(1+c_{\min})}\right)\right] \sigma(A^*, S^*)$. Let us assume

that the statement holds for $j_1, j_2, \ldots, j_{i-1}$, then

$$\begin{aligned}
\sigma(A_i, S_i) &= \sigma(A_{i-1}, S_{i-1}) + [\sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1})] \\
&\geq \sigma(A_{i-1}, S_{i-1}) + \frac{\bar{c}_i}{k} \frac{c_{\min}}{(1 + c_{\min})} [\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})] \\
&= \sigma(A_{i-1}, S_{i-1}) \left(1 - \frac{\bar{c}_i}{k} \frac{c_{\min}}{(1 + c_{\min})}\right) + \frac{\bar{c}_i}{k} \frac{c_{\min}}{(1 + c_{\min})} \sigma(A^*, S^*) \\
&\geq \left[1 - \prod_{\ell=1}^{i-1} \left(1 - \frac{\bar{c}_\ell}{k} \frac{c_{\min}}{(1 + c_{\min})}\right)\right] \left(1 - \frac{\bar{c}_i}{k} \frac{c_{\min}}{(1 + c_{\min})}\right) \sigma(A^*, S^*) \\
&\qquad + \frac{\bar{c}_i}{k} \frac{c_{\min}}{(1 + c_{\min})} \sigma(A^*, S^*) \\
&= \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \frac{c_{\min}}{(1 + c_{\min})}\right)\right] \sigma(A^*, S^*),
\end{aligned}$$

where the two inequalities follow from Lemma 17 and the inductive hypothesis, respectively. □

Theorem 19. *Algorithm 8 achieves an approximation factor of*

$$\frac{1}{2} \left(1 - \frac{1}{e^{\frac{c_{\min}}{1 + c_{\min}}}}\right) \sigma(A^*, S^*).$$

Proof. We observe that since (A_{l+1}, S_{l+1}) violates the budget, then $c(A_{l+1}, S_{l+1}) > k$. Moreover, for a sequence of numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = B$, the function $\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i}{B \cdot \beta}\right)\right]$ achieves its minimum when $a_i = \frac{B}{n}$ and that $\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i}{B \cdot \beta}\right)\right] \geq 1 - \left(1 - \frac{1}{n \cdot \beta}\right)^n \geq 1 - e^{-\frac{1}{\beta}}$. Therefore, by applying Lemma 18 for $i = l + 1$ and observing that $\sum_{\ell=1}^{l+1} \bar{c}_\ell = c(A_{l+1}, S_{l+1})$, we obtain:

$$\begin{aligned}
\sigma(A_{l+1}, S_{l+1}) &\geq \left[1 - \prod_{\ell=1}^{l+1} \left(1 - \frac{\bar{c}_\ell}{k \left(\frac{1 + c_{\min}}{c_{\min}}\right)}\right)\right] \sigma(A^*, S^*) \\
&\geq \left[1 - \prod_{\ell=1}^{l+1} \left(1 - \frac{\bar{c}_\ell}{c(A_{l+1}, S_{l+1}) \left(\frac{1 + c_{\min}}{c_{\min}}\right)}\right)\right] \sigma(A^*, S^*) \\
&\geq \left[1 - \left(1 - \frac{1}{(l+1) \frac{1 + c_{\min}}{c_{\min}}}\right)^{l+1}\right] \sigma(A^*, S^*) \\
&\geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1 + c_{\min}}}}\right) \sigma(A^*, S^*).
\end{aligned}$$

By Proposition 1, it follows that:

$$\sigma(A_{l+1}, S_{l+1}) = \sigma(A_l, S_l) + \delta_{l+1} \geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1 + c_{\min}}}}\right) \sigma(A^*, S^*). \tag{6.4}$$

Since $\delta_{l+1} \leq \sigma(\{a_M\}, \{(a_M, v_M)\})$, we get

$$\sigma(A_l, S_l) + \sigma(\{a_M\}, \{(a_M, v_M)\}) \geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^*, S^*).$$

Hence, $\max\{\sigma(A_l, S_l), \sigma(\{a_M\}, \{(a_M, v_M)\})\} \geq \frac{1}{2}\left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^*, S^*)$. □

Next, we propose an enumeration technique which improves the performance guarantee of Algorithm 8. Let M a fixed integer, we consider all the solutions (A, S) with cardinality M (i.e. $|A| + |S| = M$) and cost at most k ($c(A, S) \leq k$), and we complete all these solutions by using the greedy algorithm. The pseudocode is reported in Algorithm 9.

Algorithm 9: Enumeration technique for BIM-SL with lower bounded edge costs.

Input : A directed graph $G = (V, E, p, c)$, integer $M \in \mathbb{N}$ and $k \in \mathbb{R}^+$
Output: A set of nodes A and a of edges $S \subseteq (A \times V) \setminus E$ such that $c(A, S) \leq k$, where $c(A, S) = |A| + \sum_{e \in S} c_e$

```
(A_1, S_1) := arg max{σ(A, S) : |A| + |S| < M, c(A, S) ≤ k};
A_2 := ∅; S_2 := ∅; U := V; T := (V × V) \ E;
foreach A ⊆ U, S ⊆ (A × V) \ E such that |A| + |S| = M and c(A, S) ≤ k do
    U := U \ A; T := T \ S;
    Complete (A, S) by using Algorithm 8 with U and T as possible nodes and edges;
    if σ(A, S) > σ(A_2, S_2) then
        A_2 := A;
        S_2 := S;
return arg max{σ(A_1, S_1), σ(A_2, S_2)};
```

Theorem 20. *For $M \geq 4$ Algorithm 9 achieves an approximation factor of*

$$1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}.$$

Proof. We assume that $|A^*| + |S^*| > M$ since otherwise Algorithm 9 finds an optimal solution. We sort the elements in (A^*, S^*) by selecting at each step the element, which can be either a seed or an edge, that maximizes the marginal increment in the number of influenced nodes. Let $Z = (A_Z, S_Z)$ be the first M elements in this order. We now consider the iteration of Algorithm 9 in which element Z is considered. We define $(A_{Z'}, S_{Z'})$ as the elements added by the algorithm to (A_Z, S_Z) and $(A, S) = (A_Z \cup A_{Z'}, S_Z \cup S_{Z'})$. By Proposition 1, it follows that $\sigma(A, S) = \sigma(A_Z, S_Z) + \delta(A_Z \cup A_{Z'}, S_Z \cup S_{Z'}, A_Z, S_Z)$.

The completion of (A_Z, S_Z) to (A, S) is an application of the greedy algorithm and therefore, we can use the result from the previous theorems. Let us consider the iterations executed by the greedy algorithm during the completion of (A_Z, S_Z) to (A, S). For $i \geq 1$, let us denote by j_i the index of these iterations, $j_i < j_{i+1}$, and let j_{l+1} be the

index of the first iteration in which an element in $(A^* \setminus A_Z, S^* \setminus S_Z)$ is considered but not added to $(A_{Z'}, S_{Z'})$ because it violates the budget constraint. Applying inequality (6.4) to the instance of the problem obtained removing the nodes covered by solution Z we get:

$$\delta(A_Z \cup A_{Z'}, S_Z \cup S_{Z'}, A_Z, S_Z) + \delta_{l+1} \geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^* \setminus A_Z, S^* \setminus S_Z).$$

Moreover, since we ordered the elements in (A^*, S^*) and in iteration j_{l+1} and most 2 elements are selected, then $\delta_{l+1} \leq \frac{2\sigma(A_Z, S_Z)}{M}$ and

$$\begin{aligned}
\sigma(A, S) &= \sigma(A_Z, S_Z) + \delta(A_Z \cup A_{Z'}, S_Z \cup S_{Z'}, A_Z, S_Z) \\
&\geq \sigma(A_Z, S_Z) + \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^* \setminus A_Z, S^* \setminus S_Z) - \delta_{l+1} \\
&\geq \sigma(A_Z, S_Z) + \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^* \setminus A_Z, S^* \setminus S_Z) - \frac{2\sigma(A_Z, S_Z)}{M} \\
&\geq \left(1 - \frac{2}{M}\right) \sigma(A_Z, S_Z) + \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^* \setminus A_Z, S^* \setminus S_Z)
\end{aligned}$$

But, $\sigma(A_Z, S_Z) + \sigma(A^* \setminus A_Z, S^* \setminus S_Z) \geq \sigma(A^*, S^*)$, and we get:

$$\begin{aligned}
\sigma(A, S) &\geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^*, S^*) + \left(\frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}} - \frac{2}{M}\right) \sigma(A_Z, S_Z) \\
&\geq \left(1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}\right) \sigma(A^*, S^*),
\end{aligned}$$

for $M \geq 2e^{\frac{c_{\min}}{1+c_{\min}}}$. Since $2e^{\frac{c_{\min}}{1+c_{\min}}} < 4$ for $c_{\min} \in [0, 1]$, the theorem follows. □

6.4 Algorithm for general costs

In this section we give the results for the more challenging problem when the cost function $c : V \times V \to [0, 1]$ includes small values, therefore the cost on each edge can be arbitrary small.

We introduce a greedy algorithm for the BIM-SL problem that guarantees a constant approximation ratio since it does not depend on the edge costs.

The algorithm is similar to Algorithm 8, and it is reported in Algorithm 10.

The algorithm finds a solution by means of a greedy approach that at each step maximizes the ratio between the marginal increment in the objective function of a candidate solution and its marginal cost.

The main difference with Algorithm 8 consists in the way in which the greedy algorithm computes a candidate solution. In fact, the algorithm presented here might select more than one edge at one time. In particular, at each iteration the greedy algorithm

Algorithm 10: Greedy algorithm for the general BIM-SL problem.

Input : A directed graph $G = (V, E)$, $k \in \mathbb{R}^+$ and $b = O(1)$
Output: A set of nodes A and a of edges $S \subseteq A \times V \setminus E$ such that $c(A, S) \leq k$

$A := \emptyset$; $S := \emptyset$; $U := V$; $T := (V \times V) \setminus E$;
while $k > 0$ **do**
  $r_1 = \max_{a \in U}\{\delta(A \cup \{a\}, S, A, S)\}$;
  $r_2 = \max_{(a,v) \in A \times V \cap T} \left\{ \frac{\delta(A, S \cup \{(a,v)\}, A, S)}{c_{(a,v)}} \right\}$;
  $r_3 = \max_{a \in U, (a,v) \in \{a\} \times V \cap T | c_{(a,v)} \geq b} \left\{ \frac{\delta(A \cup \{a\}, S \cup \{(a,v)\}, A, S)}{1 + c_{(a,v)}} \right\}$;
  $r_4 = \max_{a \in U, I = \{(a,v) | (a,v) \in \{a\} \times V \cap T, c_{(a,v)} < b\}, \sum_{(a,v) \in I} c_{(a,v)} < b} \left\{ \frac{\delta(A \cup \{a\}, S \cup I, A, S)}{1 + \sum_{(a,v) \in I} c_{(a,v)}} \right\}$;
  if $\max\{r_1, r_2, r_3, r_4\} = r_1$ **then**
    $\hat{a} = \arg\max_{a \in U}\{\delta(A \cup \{a\}, S, A, S)\}$;
    if $k - 1 \geq 0$ **then**
      $A := A \cup \{\hat{a}\}$;
      $k := k - 1$;
    $U := U \setminus \{\hat{a}\}$;
  else
    if $\max\{r_1, r_2, r_3, r_4\} = r_2$ **then**
      $(\hat{a}, \hat{v}) := \arg\max_{(a,v) \in A \times V \cap T} \left\{ \frac{\delta(A, S \cup \{(a,v)\}, A, S)}{c_{(a,v)}} \right\}$;
      if $k - c_{(\hat{a},\hat{v})} \geq 0$ **then**
        $S := S \cup \{(\hat{a}, \hat{v})\}$;
        $k := k - c_{(\hat{a},\hat{v})}$;
      $T := T \setminus \{(\hat{a}, \hat{v})\}$;
    else
      if $\max\{r_1, r_2, r_3, r_4\} = r_3$ **then**
        $(\hat{a}, (\hat{a}, \hat{v})) := \arg\max_{a \in U, (a,v) \in \{a\} \times V \cap T | c_{(a,v)} \geq b} \left\{ \frac{\delta(A \cup \{a\}, S \cup \{(a,v)\}, A, S)}{1 + c_{(a,v)}} \right\}$;
        if $k - c_{(\hat{a},\hat{v})} - 1 \geq 0$ **then**
          $A := A \cup \{\hat{a}\}$;
          $S := S \cup \{(\hat{a}, \hat{v})\}$;
          $k := k - 1 - c_{(\hat{a},\hat{v})}$;
        $U := U \setminus \{\hat{a}\}$;
        $T := T \setminus \{(\hat{a}, \hat{v})\}$;
      else
        $(\hat{a}, \hat{I}) :=$
        $\max_{a \in U, I = \{(a,v) | (a,v) \in \{a\} \times V \cap T, c_{(a,v)} < b\}, \sum_{(a,v) \in I} c_{(a,v)} < b} \left\{ \frac{\delta(A \cup \{a\}, S \cup I, A, S)}{1 + \sum_{(a,v) \in I} c_{(a,v)}} \right\}$;
        if $k - c_{(\hat{a},\hat{I})} - 1 \geq 0$ **then**
          $A := A \cup \{\hat{a}\}$;
          $S := S \cup \hat{I}$;
          $k := k - 1 - c_{(\hat{a},\hat{I})}$;
        $U := U \setminus \{\hat{a}\}$;
        $T := T \setminus \hat{I}$;
return $\sigma(A, S)$;

computes four candidate solutions starting from the current solution (A', S'). The first two solutions correspond to cases r_1 and r_2 of Algorithm 8, and differ from (A', S') by a single seed node or a single edge incident to a seed in A', respectively.

- Select a seed node a not in A' that maximizes $r_1 = \delta(A' \cup \{a\}, S', A', S')$, $(A, S) = (A' \cup \{a\}, S')$;
- Select an edge (a, v) incident to a seed a in A' that maximizes $r_2 = \frac{\delta(A', S' \cup \{(a,v)\}, A', S')}{c_{(a,v)}}$, $(A, S) = (A', S' \cup \{(a, v)\})$;

To compute the two further candidate solutions, the algorithm divides the edges into two sets: those whose cost is at least b and those whose cost is smaller than b, for some given constant b, and proceeds as follows.

- Select a seed node a not in A' and an edge (a, v), which cost is $c_{(a,v)} \geq b$, incident to a that maximize $\frac{\delta(A' \cup \{a\}, S' \cup \{(a,v)\}, A', S')}{1 + c_{(a,v)}}$;
- Select a seed node a not in A' and a set S of edges incident to a, whose overall cost is smaller than b (i.e. $\sum_{(a,v) \in S} c_{(a,v)} < b$), that maximize $\frac{\delta(A' \cup \{a\}, S' \cup S, A', S')}{1 + \sum_{(a,v) \in S} c_{(a,v)}}$.

The greedy phase of the algorithm selects the candidate solution that gives the maximum ratio among the above four possibilities.

To compute the fourth candidate solution we need to solve an optimization problem, which we call RBIM-SL (where R stands for ratio). In the following, we assume that we can exploit an α-approximation algorithm for this sub-problem. Formally RBIM-SL is defined as follows:

RatioBIM-SL	
Given:	A weighted directed graph $G = (V, E, p)$; and an integer $b \in \mathbb{N}$
Solution:	A seed $a \in V$, a set of edges $S \subseteq (\{a\} \times V) \setminus E$ such that $\sum_{(a,v) \in S} c_{(a,v)} < b$
Goal:	Maximize $\frac{\sigma(\{a\}, S)}{1 + \sum_{(a,v) \in S} c_{(a,v)}}$

The analysis of the algorithm is similar to that in the previous section. In particular, the next lemma is the core of the analysis and corresponds to Lemma 17. We use the same notation used in the previous section. We denote by j_i an iteration of the greedy algorithm in which an element is added to the solution, by j_{l+1} the first iteration in which an element of the optimum is not added to the solution and by $\bar{c}_i$ the marginal cost of the solution (A_i, S_i) computed at the iteration j_i.

Lemma 21. *After each iteration j_i, $i = 1, 2, \ldots, l + 1$,*

$$\sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1}) \geq \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2} (\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})).$$

Proof. We denote by δ_i the expected number of nodes affected by solution (A_i, S_i) and not affected by solution (A_{i-1}, S_{i-1}), $\delta_i = \delta(A_i, S_i, A_{i-1}, S_{i-1})$. Moreover, we divide

the set $A^* \setminus A_{i-1}$ into two subsets: A_1^* is the subset of $A^* \setminus A_{i-1}$ that contains the seeds a with no incident edges in S^*, and $A_2^* = A^* \setminus (A_{i-1} \cup A_1^*)$. For each seed $a \in A_2^*$ let us consider the subset of S^* containing edges incident to a such that $c_e < b$.

We partition this set into sets of edges whose overall cost is smaller than b in such a way that the number of sets in the partition is minimized. We denote by $\mathcal{S}_a$ this partition and observe that the overall number of sets in $\mathcal{S}_a$, for all $a \in A_2^*$, is at most $2\frac{k}{b}$ as it is equivalent to the minimum number of bins of size b needed to pack a set of items of overall size k.

By using similar arguments as in Lemma 17, we can show that

$$\begin{aligned}\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1}) \leq & \sum_{a \in A_1^*} \delta(A_{i-1} \cup \{a\}, S_{i-1}, A_{i-1}, S_{i-1}) \\ & + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} \delta(A_{i-1}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1}) \qquad (6.5) \\ & + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1} \\ c_e \geq b}} \delta(A_{i-1} \cup \{a\}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1}) \\ & + \sum_{\substack{a \in A_2^* \\ S \in \mathcal{S}_a}} \delta(A_{i-1} \cup \{a\}, S_{i-1} \cup S, A_{i-1}, S_{i-1}).\end{aligned}$$

Indeed,

$$\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1}) \leq$$

$$\sum_{X \in \chi(S_{i-1})} \mathbb{P}[X] \sum_{Y \in \chi(S^* \cup S_{i-1}, X)} \mathbb{P}[Y|X](|R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)|)$$

and for each $X \in \chi(S_{i-1})$ and $Y \in \chi(S^* \cup S_{i-1}, X)$,

$$\begin{aligned}|R(A^* \cup A_{i-1}, Y)| - |R(A_{i-1}, X)| \leq & \sum_{a \in A_1^*} (|R(A_{i-1} \cup \{a\}, X)| - |R(A_{i-1}, X)|) \\ & + \sum_{\substack{e=(a,v) \in Y \setminus X \\ \text{s.t. } a \in A_{i-1}}} (|R(A_{i-1}, X \cup \{e\})| - |R(A_{i-1}, X)|) \\ & + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in Y \setminus X \\ c_e \geq b}} (|R(A_{i-1} \cup \{a\}, X \cup \{e\})| - |R(A_{i-1}, X)|) \\ & + \sum_{\substack{a \in A_2^* \\ S \in \mathcal{S}_a}} (|R(A_{i-1} \cup \{a\}, X \cup (S \cap Y))| - |R(A_{i-1}, X)|)\,.\end{aligned}$$

The following holds since the greedy phase of the algorithm selects a solution that maximizes the ratio between the marginal increment in the objective function and the cost:

- For each $a \in A_1^*$, $\delta(A_{i-1} \cup \{a\}, S_{i-1}, A_{i-1}, S_{i-1}) \leq \frac{\delta_i}{c_i}$;

- For each $e = (a, v) \in S^* \setminus S_{i-1}$ such that $a \in A_{i-1}$, $\frac{\delta(A_{i-1}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1})}{c_e} \leq \frac{\delta_i}{\bar{c}_i}$;
- For each $a \in A_2^*$ and $e = (a, v) \in S^* \setminus S_{i-1}$ s.t. $c_e \geq b$, $\frac{\delta(A_{i-1} \cup \{a\}, S_{i-1} \cup \{e\}, A_{i-1}, S_{i-1})}{1 + c_e} \leq \frac{\delta_i}{\bar{c}_i}$;
- For each $a \in A_2^*$ and $S \in \mathcal{S}_a$, $\frac{\delta(A_{i-1} \cup \{a\}, S_{i-1} \cup S, A_{i-1}, S_{i-1})}{1 + \sum_{e \in S} c_e} \leq \frac{\delta_i}{\bar{c}_i} \cdot \frac{1}{\alpha}$,

where the term $\frac{1}{\alpha}$ in the last inequality is due to the use of an α-approximation algorithm in the computation of the fourth candidate solution of the greedy phase. The number of edges incident to each $a \in A_2^*$ with cost at least b is at most $\frac{k}{b}$. Therefore, the right hand side of (6.5) is at most:

$$\sum_{a \in A_1^*} \frac{\delta_i}{\bar{c}_i} + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} \frac{\delta_i}{\bar{c}_i} c_e + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1} \\ c_e \geq b}} \frac{\delta_i}{\bar{c}_i}(1 + c_e) + \sum_{\substack{a \in A_2^* \\ S \in \mathcal{S}_a}} \frac{1}{\alpha} \frac{\delta_i}{\bar{c}_i} \left(1 + \sum_{e \in S} c_e\right)$$

$$= \frac{\delta_i}{\bar{c}_i} \left(|A_1^*| + \sum_{\substack{e=(a,v) \in S^* \setminus S_{i-1} \\ \text{s.t. } a \in A_{i-1}}} c_e + \frac{k}{b} + \sum_{\substack{a \in A_2^* \\ e=(a,v) \in S^* \setminus S_{i-1} \\ c_e \geq b}} c_e + \frac{1}{\alpha} \frac{2k}{b} + \frac{1}{\alpha} \sum_{\substack{a \in A_2^* \\ S \in \mathcal{S}_a}} \sum_{e \in S} c_e \right)$$

$$\leq \left(1 + \frac{1}{b} + \frac{2}{b\alpha} + \frac{1}{\alpha}\right) k \frac{\delta_i}{\bar{c}_i} = \left(\frac{b\alpha + \alpha + 2 + b}{b\alpha}\right) k \frac{\delta_i}{\bar{c}_i}.$$

To conclude, we observe that by Proposition 1, it follows that

$$\delta_i = \sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1}).$$

□

Lemma 22. *After each iteration j_i, $i = 1, 2, \ldots, l+1$,*

$$\sigma(A_i, S_i) \geq \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2}\right)\right] \sigma(A^*, S^*).$$

Proof. We show the statement by induction on iterations j_i. For $i = 1$, by Lemma 21, $\sigma(A_1, S_1) \geq \frac{\bar{c}_1}{k} \frac{b\alpha}{b\alpha+b+\alpha+2} \sigma(A^*, S^*) = \left[1 - \left(1 - \frac{\bar{c}_1}{k} \frac{b\alpha}{b\alpha+b+\alpha+2}\right)\right] \sigma(A^*, S^*)$. Let us assume

that the statement holds for $j_1, j_2, \ldots, j_{i-1}$, then

$$\begin{aligned}
\sigma(A_i, S_i) &= \sigma(A_{i-1}, S_{i-1}) + [\sigma(A_i, S_i) - \sigma(A_{i-1}, S_{i-1})] \\
&\geq \sigma(A_{i-1}, S_{i-1}) + \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2} [\sigma(A^*, S^*) - \sigma(A_{i-1}, S_{i-1})] \\
&= \sigma(A_{i-1}, S_{i-1}) \left(1 - \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2}\right) + \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2} \sigma(A^*, S^*) \\
&\geq \left[1 - \prod_{\ell=1}^{i-1} \left(1 - \frac{\bar{c}_\ell}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2}\right)\right] \left(1 - \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2}\right) \sigma(A^*, S^*) \\
&\qquad + \frac{\bar{c}_i}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2} \sigma(A^*, S^*) \\
&= \left[1 - \prod_{\ell=1}^{i} \left(1 - \frac{\bar{c}_\ell}{k} \frac{b\alpha}{b\alpha + b + \alpha + 2}\right)\right] \sigma(A^*, S^*),
\end{aligned}$$

where the two inequalities follows from Lemma 21 and the inductive hypothesis, respectively. □

Therefore, equipped with Lemma 22, we can prove the next theorem.

Theorem 23. *The greedy algorithm achieves an approximation factor of*

$$1 - \frac{1}{e^{\frac{1}{1+\epsilon} \frac{b\alpha}{b\alpha+b+\alpha+2}}},$$

for any constant $\epsilon \in (0, 1]$.

Proof. We recall that, for a sequence of numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = \beta \cdot B$, the function $\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i}{B}\right)\right]$ achieves its minimum when $a_i = \frac{\beta \cdot B}{n}$ and that

$$\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i}{B}\right)\right] \geq 1 - \left(1 - \frac{\beta}{n}\right)^n \geq 1 - e^{-\beta}.$$

Since solution (A_{l+1}, S_{l+1}) exceeds the budget, then $c(A_l, S_l) + \bar{c}_{l+1} > k$. By the algorithm, $\bar{c}_{l+1} \leq 1 + b$, which implies $k < c(A_l, S_l) + 1 + b$. Moreover, since k is not a constant, then for any constant $\epsilon \in (0, 1]$, $\epsilon c(A_l, S_l) \geq 1 + b$. Indeed, if, for some constant $\epsilon \in (0, 1]$, $\epsilon c(A_l, S_l) < 1 + b$, then $k < (1 + b)\left(1 + \frac{1}{\epsilon}\right) = O(1)$, which is a contradiction. This implies that $k < (1 + \epsilon) c(A_l, S_l)$, for any constant $\epsilon \in (0, 1]$.

Then, by Lemma 22 for $i = l$ and the previous property, we obtain:

$$\begin{aligned}
\sigma(A_l, S_l) &\geq \left[1 - \prod_{\ell=1}^{l} \left(1 - \frac{\bar{c}_\ell}{k\left(\frac{b\alpha+b+\alpha+2}{b\alpha}\right)}\right)\right] \sigma(A^*, S^*) \\
&\geq \left[1 - \prod_{\ell=1}^{l} \left(1 - \frac{\bar{c}_\ell}{c(A_l, S_l)(1+\epsilon)\left(\frac{b\alpha+b+\alpha+2}{b\alpha}\right)}\right)\right] \sigma(A^*, S^*) \\
&\geq \left[1 - \left(1 - \frac{1}{l(1+\epsilon)\frac{b\alpha+b+\alpha+2}{b\alpha}}\right)^{l}\right] \sigma(A^*, S^*) \\
&\geq \left(1 - \frac{1}{e^{\frac{1}{1+\epsilon}\frac{b\alpha}{b\alpha+b+\alpha+2}}}\right) \sigma(A^*, S^*).
\end{aligned}$$

□

It remains to give an α-approximation algorithm for the RBIM-SL problem which consists in selecting a seed node a not in the current solution (A', S') and a set S of edges incident to a, which overall cost is smaller than b (i.e. $\sum_{(a,v)\in S} c_{(a,v)} < b$), that maximize $\frac{\delta(A'\cup\{a\}, S'\cup S, A', S')}{1+\sum_{(a,v)\in S} c_{(a,v)}}$.

To this aim recall the BIM-L problem defined in Chapter 5. Given a directed graph $G = (V, E, p, c)$, $k \in \mathbb{R}^+$ and a seed set A, find a set of edges $S \subseteq A \times V \setminus E$ such that $c(S) \leq k$ and $\sigma(A, S)$ is maximized.

Let us consider the instances of the BIM-L problem where $A = \{a\}$ and let m^* be the maximum, over all $a \in V \setminus A'$, among the optima of these instances. It is easy to show that m^* is at least $b + 1$ times the optimum of RBIM-L. We showed that the BIM-L problem can be approximated within a factor of $1 - \frac{1}{e}$ by using a greedy algorithm [27], therefore we obtain an overall approximation of $\alpha = \left(1 - \frac{1}{e}\right)\frac{1}{b+1}$.

Corollary 24. *There exists an algorithm that achieves an approximation factor* > 0.0878 *for the* BIM-SL *problem.*

Proof. It follows by applying Theorem 23 with $\alpha = \left(1 - \frac{1}{e}\right)\frac{1}{b+1}$ and optimizing over b (that is with $b = \sqrt{3 - \frac{1}{e}}$). □

To conclude, in Figure 6.1 we summarize our approximation ratios as a function of $c_{\min}$. As expected, when all the edge costs are high, it is not worth to buy them but it is better to buy a seed node. Indeed the algorithm that selects only seeds outperforms the other algorithms and it reaches the optimal approximation of $1 - \frac{1}{e}$ when all the costs are equal to 1.

However, the challenge of our problem consists in finding a good approximation even when the cost function includes small values. In the general case and when the minimum edge cost can be very small ($c_{\min} < 0.1011$) the best algorithm is the constant factor algorithm given in Section 6.4 which guarantees an approximation factor grater that 0.0878, while when the minimum cost is in $(0.1011, 0.3821)$ the best algorithm is the one presented in Section 6.3 which provides an approximation factor of $1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}$.

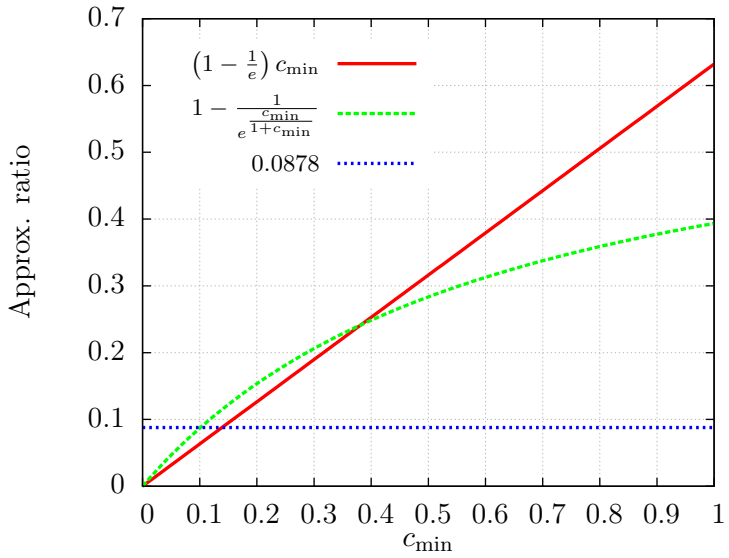

FIGURE 6.1: Approximation ratio of the three algorithm presented as a function of $c_{\min}$. Intersection points are at $c_{\min} \approx 0.1011$ and $c_{\min} \approx 0.3821$

Differently form the previous chapters, we cannot provide an experimental evaluation of BIM-SL. It is easy to see that the algorithm would require too much computational time in practice. Indeed, any of the possible choices the algorithm can perform, whether buying a new seed, buying an edge incident to a seed or buying a seed and a set of edges incident to it, requires at least the same computational time to solve the Influence Maximization problem in [41] and the BIM-L problem. As a future work we plan on considering techniques to speed-up the greedy algorithm, so as to evaluate the quality of its solution.

Chapter 7

Conclusion

In the field of complex networks analysis, the study of social influence has long attracted the attention of psychologists and marketing scientists, it became popular among computer scientists in the last decade only.

Many applications like viral marketing, advertising, recommender systems, analysis of information diffusion in Social Media like Twitter and Facebook and link prediction can benefit from information diffusion analysis.

One of the fundamental problems in the study of influence spreading is how to shape a given diffusion process so as to maximize the number of influenced nodes at the end of the process by taking intervention actions. The most important action is seed selection. Its goal is to identify a small set of users in a network, who, when convinced to adopt a product, will influence others in the network leading to a large number of adoptions at the end of the diffusion process.

In this dissertation, we tackle the problem from a different point of view and consider other intervention actions such as link addition.

We define new and more general problems: IM-L, BIM-L and BIM-SL.

The IM-L problem consists on adding, to the graph, a limited number of edges incident to an initial set of given nodes in order to maximize the information diffusion. We have shown that IM-L admits a constant factor approximation algorithm by proving that the expected number of activated nodes is monotonically increasing and submodular with respect to the possible set of edges incident to the seeds. We further provide an upper bound to the approximation factor that is slightly higher than that guaranteed by our algorithm.

In BIM-L, instead, we consider the cost version of the previous problem, i.e. there is a cost to be paid in order to add an edge to the graph, and the goal is to maximize the spread of influence without exceeding a given budget. We have shown how to approximate the more general BIM-L problem: we first provided an algorithm which achieves a $\frac{1}{2}\left(1-\frac{1}{e}\right)$ approximation factor and then, using the enumeration technique, we improved the performance guarantee within a factor of $1-\frac{1}{e}$.

Moreover, by means of experiments, we showed that our algorithm is able to highly increase the number of activated nodes and that it outperforms several baselines.

The last, more general, problem we investigate is BIM-SL: we want to find both a set of seeds and a set of edges, incident to these seeds, such that the number of nodes

influenced at the end of the process is maximized and the cost of the selected element does not exceed the given budget. We prove that the BIM-SL problem cannot be approximated within a factor greater than $1 - \frac{1}{e}$, unless $P = NP$. Therefore, we focus on approximation algorithms. We first assume that the edge costs are all greater that a given constant c_{min} and we propose two approximation algorithms that guarantee approximation factors of $1 - \frac{1}{e^{\frac{c_{\min}}{1+c_{\min}}}}$ and $\left(1 - \frac{1}{e}\right) c_{\min}$, respectively. Then, we attack the challenging case of our problem that consists in finding a good approximation even when the cost function includes small values. We propose an algorithm that guarantees a constant approximation ratio of about 0.0878.

The results presented in this thesis made a progress in the study of influence maximization and point out several new challenges. In the following we list several open problems worthy of further investigation.

The first open problem, directly related to this thesis, is the study of our problems in a generalization of ICM, which is the Decreasing Cascade model [41]. In this model the probability of a node u to influence v is non-increasing as a function of the set of nodes that have previously tried to influence v. DCM generalizes the Independent Cascade Model by taking into account the effect of previous attempts to activate a node on its chances of being activated in the future, hence it is no longer independent. The decreasing aspect is the assumption that the more neighbours try to influence v, the less likely it will become active.

Other research directions that deserve further investigation include the study of IM-L, BIM-L and BIM-SL on different information diffusion models such as LTM or the Triggering Model [41].

Next, we plan to analyse a minimization version of our problems where we allow the deletion of edges incident to seeds and the deletion of seeds. This setting is interesting in the field of epidemiology in order to limit the diffusion of a virus or to block the contamination of a disease.

Another interesting future work is to improve the running time of the algorithms used for IM-L and BIM-L, and to design new techniques to obtain a fast algorithm for the BIM-SL problem in order to evaluate the solution quality and the performances on real-world networks.

The main open problem regard BIM-SL is the gap between the lower bound on approximation of $1-\frac{1}{e} \approx 0.6321$ and the constant approximation of 0.0878. To close this gap, we aim at improving the algorithm in Chapter 6 by devising a better approximation algorithm for RBIM-L problem, which directly implies a better approximation for the BIM-SL problem.

In this thesis, we assumed that each node knows the topology of the graph, i.e. the graph is public. It would be interesting to study the case where we have a public graph and, moreover, each node in the public graph has an associated private graph. This representation of graphs is called private-public model [16] and it is useful to depict social networks where the nodes are the users. In this scenario, the public graph is

visible to everyone and the private graph of each node is visible only to the user. From each node, the graph is just a union of its private graph and the public graph.

Finally, it would be challenging to study the budgeted seed selection and link addition problem in time evolving graphs since networks, nowadays, are dynamically changing in their topologies.

Bibliography

[1] Lada A. Adamic and Eytan Adar. "Friends and Neighbors on the Web". In: *SOCIAL NETWORKS* 25 (2001), pp. 211–230.

[2] *Arnetminer*. `http://arnetminer.org`. Accessed: 2015-01-15.

[3] C. Asavathiratham, S. Roy, B. Lesieutre, and G. Verghese. "The influence model". In: *IEEE Control Systems* 21.6 (2001), pp. 52–64.

[4] Lars Backstrom and Jure Leskovec. "Supervised random walks: predicting and recommending links in social networks". In: *Proceedings of the Forth International Conference on Web Search and Web Data Mining (WSDM)*. ACM, 2011, pp. 635–644.

[5] Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. "Everyone's an Influencer: Quantifying Influence on Twitter". In: *Proc. of the 4th ACM International Conference on Web Search and Data Mining*. WSDM11. Hong Kong, China: ACM, 2011, pp. 65–74. ISBN: 978-1-4503-0493-1.

[6] Albert-László Barabási and Eric Bonabeau. "Scale-free networks". In: *Sci. Am.* 288.5 (2003), pp. 50–59.

[7] Frank M Bass. "A new product growth for model consumer durables". In: *Management science* 15.5 (1969), pp. 215–227.

[8] Elisabetta Bergamini, Pierluigi Crescenzi, Gianlorenzo D'Angelo, Henning Meyerhenke, Lorenzo Severini, and Yllka Velaj. "Improving the betweenness centrality of a node by adding links". In: *arXiv preprint arXiv:1702.05284* (2017).

[9] Paolo Boldi and Sebastiano Vigna. "Axioms for centrality". In: *Internet Mathematics* 10.3–4 (2014), pp. 222–262.

[10] Béla Bollobás, Christian Borgs, Jennifer Chayes, and Oliver Riordan. "Directed scale-free graphs". In: *Proceedings of the 14th annual ACM-SIAM symposium on Discrete algorithms (SODA)*. SIAM. 2003, pp. 132–139.

[11] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. "Maximizing Social Influence in Nearly Optimal Time". In: *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '14. Society for Industrial and Applied Mathematics, 2014, pp. 946–957. ISBN: 978-1-611973-38-9. URL: `http://dl.acm.org/citation.cfm?id=2634074.2634144`.

[12] Vineet Chaoji, Sayan Ranu, Rajeev Rastogi, and Rushi Bhatt. "Recommendations to boost content spread in social networks". In: *Proceedings of the 21st World Wide Web Conference 2012 (WWW12)*. ACM, 2012, pp. 529–538.

[13] Wei Chen, Chi Wang, and Yajun Wang. "Scalable Influence Maximization for Prevalent Viral Marketing in Large-scale Social Networks". In: *Proc. of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD10. 2010.

[14] Wei Chen, Yajun Wang, and Siyu Yang. "Efficient Influence Maximization in Social Networks". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. ACM, 2009, pp. 199–208. ISBN: 978-1-60558-495-9. DOI: 10.1145/1557019.1557047. URL: http://doi.acm.org/10.1145/1557019.1557047.

[15] Wei Chen, Yifei Yuan, and Li Zhang. "Scalable Influence Maximization in Social Networks Under the Linear Threshold Model". In: *Proceedings of the 2010 IEEE International Conference on Data Mining*. ICDM '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 88–97.

[16] Flavio Chierichetti, Alessandro Epasto, Ravi Kumar, Silvio Lattanzi, and Vahab Mirrokni. "Efficient Algorithms for Public-Private Social Networks". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. Sydney, NSW, Australia: ACM, 2015, pp. 139–148.

[17] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, Alessandro Panconesi, and Prabhakar Raghavan. "Models for the compressible web". In: *Proceedings of the 50th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2009, pp. 331–340.

[18] Edith Cohen. "Size-Estimation Framework with Applications to Transitive Closure and Reachability". In: *J. Comput. Syst. Sci.* 55.3 (Dec. 1997), pp. 441–453. ISSN: 0022-0000. DOI: 10.1006/jcss.1997.1534. URL: http://dx.doi.org/10.1006/jcss.1997.1534.

[19] Edith Cohen and Haim Kaplan. "Summarizing Data Using Bottom-k Sketches". In: *Proceedings of the Twenty-sixth Annual ACM Symposium on Principles of Distributed Computing*. PODC '07. ACM, 2007, pp. 225–234. ISBN: 978-1-59593-616-5. DOI: 10.1145/1281100.1281133. URL: http://doi.acm.org/10.1145/1281100.1281133.

[20] Edith Cohen, Daniel Delling, Thomas Pajor, and Renato 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*. CIKM '14. ACM, 2014, pp. 629–638. ISBN: 978-1-4503-2598-1. DOI: 10.1145/2661829.2662077. URL: http://doi.acm.org/10.1145/2661829.2662077.

[21] James S. Coleman, Elihu Katz, and Herbert Menzel. *Medical innovation: A diffusion study*. The Bobbs-Merrill Company, 1966.

[22] Gennaro Cordasco, Luisa Gargano, Manuel Lafond, Lata Narayanan, Adele A. Rescigno, Ugo Vaccaro, and Kangkang Wu. "Whom to befriend to influence people". In: *CoRR* abs/1611.08687 (2016). URL: `http://arxiv.org/abs/1611.08687`.

[23] Pierluigi Crescenzi, Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Greedily improving our own centrality in a network". In: *International Symposium on Experimental Algorithms*. Springer, Cham. 2015, pp. 43–55.

[24] Pierluigi Crescenzi, Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Greedily Improving Our Own Closeness Centrality in a Network". In: *ACM Trans. Knowl. Discov. Data* 11.1 (2016), 9:1–9:32.

[25] Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Influence Maximization in the Independent Cascade Model". In: *Proceedings of the 17th Italian Conference on Theoretical Computer Science, Lecce, Italy, September 7-9, 2016.* 2016, pp. 269–274.

[26] Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "On the maximum betweenness improvement problem". In: *Electronic Notes in Theoretical Computer Science* 322 (2016), pp. 153–168.

[27] Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Recommending links through influence maximization". In: *arXiv preprint* (2017). URL: `http://arxiv.org/abs/1706.04368`.

[28] Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Selectiong nodes and buying links to maximize the information diffusion ona network". In: *42st International Symposium on Mathematical Foundations of Computer Science (MFCS 2017)*. Leibniz International Proceedings in Informatics (LIPIcs). Aalborg, Denmark, 2017, 75:1–75:14.

[29] Gianlorenzo D'Angelo, Lorenzo Severini, and Yllka Velaj. "Selectiong nodes and buying links to maximize the information diffusion ona network". In: *arXiv preprint* (2017). URL: `http://arxiv.org/abs/1706.04368`.

[30] Pedro Domingos and Matt Richardson. "Mining the Network Value of Customers". In: *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD01. ACM, 2001, pp. 57–66.

[31] P. Erdős and A. Rényi. "On random graphs I". In: *Publicationes Mathematicae* 6 (1959), pp. 290–297.

[32] Uriel Feige. "A Threshold of Ln N for Approximating Set Cover". In: *Journal of the ACM* 45.4 (1998).

[33] Jacob Goldenberg, Barak Libai, and Eitan Muller. "Talk of the network: A complex systems look at the underlying process of word-of-mouth". In: *Marketing letters* 12.3 (2001), pp. 211–223.

[34] Jacob Goldenberg, Barak Libai, and Eitan Muller. "Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata". In: *Academy of Marketing Science Review* 2001.9 (2001), p. 1.

[35] Amit Goyal, Wei Lu, and Laks V. S. Lakshmanan. "SIMPATH: An Efficient Algorithm for Influence Maximization Under the Linear Threshold Model". In: *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*. ICDM '11. IEEE Computer Society, 2011, pp. 211–220. ISBN: 978-0-7695-4408-3. DOI: 10.1109/ICDM.2011.132. URL: http://dx.doi.org/10.1109/ICDM.2011.132.

[36] Amit Goyal, Wei Lu, and Laks V.S. Lakshmanan. "CELF++: Optimizing the Greedy Algorithm for Influence Maximization in Social Networks". In: *Proceedings of the 20th International Conference Companion on World Wide Web*. WWW '11. ACM, 2011, pp. 47–48. ISBN: 978-1-4503-0637-9. DOI: 10.1145/1963192.1963217. URL: http://doi.acm.org/10.1145/1963192.1963217.

[37] Mark Granovetter. "Threshold models of collective behavior". In: *American journal of sociology* 83.6 (1978), pp. 1420–1443.

[38] Paul Jaccard. "Étude comparative de la distribution florale dans une portion des Alpes et des Jura". In: *Bulletin del la Société Vaudoise des Sciences Naturelles* 37 (1901), pp. 547–579.

[39] Kyomin Jung, Wooram Heo, and Wei Chen. "IRIE: Scalable and Robust Influence Maximization in Social Networks". In: *Proceedings of the 2012 IEEE 12th International Conference on Data Mining*. ICDM '12. IEEE Computer Society, 2012, pp. 918–923. ISBN: 978-0-7695-4905-7. DOI: 10.1109/ICDM.2012.79. URL: http://dx.doi.org/10.1109/ICDM.2012.79.

[40] David Kempe, Jon Kleinberg, and Éva 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*. KDD03. ACM, 2003, pp. 137–146.

[41] David Kempe, Jon Kleinberg, and Éva Tardos. "Maximizing the Spread of Influence through a Social Network". In: *Theory of Computing* 11.4 (2015), pp. 105–147.

[42] Elias Boutros Khalil, Bistra Dilkina, and Le Song. "Scalable Diffusion-aware Optimization of Network Topology". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD14. ACM, 2014, pp. 1226–1235.

[43] Samir Khuller, Anna Moss, and Joseph Naor. "The Budgeted Maximum Coverage Problem". In: *Inf. Process. Lett.* 70.1 (1999), pp. 39–45.

[44] Masahiro Kimura, Kazumi Saito, and Hiroshi Motoda. "Blocking Links to Minimize Contamination Spread in a Social Network". In: *ACM Trans. Knowl. Discov. Data* 3.2 (2009), 9:1–9:23.

[45] Masahiro Kimura, Kazumi Saito, and Hiroshi Motoda. "Solving the Contamination Minimization Problem on Networks for the Linear Threshold Model". In: *Proc. of the 10th Pacific Rim International Conference on Artificial Intelligence*. PRICA08. Springer Berlin Heidelberg, 2008, pp. 977–984.

[46] Chris J Kuhlman, Gaurav Tuli, Samarth Swarup, Madhav V Marathe, and SS Ravi. "Blocking Simple and Complex Contagion By Edge Removal". In: *IEEE International Conference on Data Mining*. ICDM13. IEEE. 2013.

[47] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins, and Eli Upfal. "Stochastic models for the web graph". In: *Proceedings ot the 41st Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2000, pp. 57–65.

[48] Jérôme Kunegis. "KONECT - The Koblenz Network Collection". In: *Proceedings of the 1st International Web Observatory Workshop (WOW)*. 2013, pp. 1343–1350.

[49] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. "Graph Evolution: Densification and Shrinking Diameters". In: *ACM Trans. Knowl. Discov. Data* 1.1 (Mar. 2007). ISSN: 1556-4681. DOI: 10.1145/1217299.1217301. URL: http://doi.acm.org/10.1145/1217299.1217301.

[50] Jure Leskovec and Andrej Krevl. *SNAP Datasets: Stanford Large Network Dataset Collection*. http://snap.stanford.edu/data. June 2014.

[51] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. "Cost-effective Outbreak Detection in Networks". In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '07. ACM, 2007, pp. 420–429. ISBN: 978-1-59593-609-7. DOI: 10.1145/1281192.1281239. URL: http://doi.acm.org/10.1145/1281192.1281239.

[52] Dong Li, Zhiming Xu, Sheng Li, Xin Sun, Anika Gupta, and Katia P. Sycara. "Link recommendation for promoting information diffusion in social networks". In: *22nd International World Wide Web Conference (WWW), Companion Volume*. ACM, 2013, pp. 185–186.

[53] Zhepeng (Lionel) Li, Xiao Fang, and Olivia R. Liu Sheng. "A Survey of Link Recommendation for Social Networks: Methods, Theoretical Foundations, and Future Research Directions". In: *CoRR* abs/1511.01868 (2015). URL: http://arxiv.org/abs/1511.01868.

[54] David Liben-Nowell and Jon M. Kleinberg. "The link prediction problem for social networks". In: *Proceedings of the 12th ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 2003, pp. 556–559.

[55] Linyuan Lü and Tao Zhou. "Link prediction in complex networks: A survey". In: *Physica A: Statistical Mechanics and its Applications* 390.6 (2011), pp. 1150 – 1170.

[56] Vijay Mahajan, Eitan Müller, and Frank M Bass. "New Product Diffusion Models in Marketing: A Review and Directions for Research". In: *Journal of Marketing* 54 (1990), pp. 1–26.

[57] Lauren Meyers. "Contact network epidemiology: Bond percolation applied to infectious disease prediction and control". In: *Bulletin of the American Mathematical Society* 44.1 (2007), pp. 63–86.

[58] Stanley Milgram. "The small world problem". In: *Psychology today* 2.1 (1967), pp. 60–67.

[59] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. "An analysis of approximations for maximizing submodular set functions–I". In: *Mathematical Programming* 14.1 (1978), pp. 265–294.

[60] M. E. J. Newman. "Clustering and preferential attachment in growing networks". In: *Physical Review E* 64.2 (2001), p. 025102.

[61] M. E. J. Newman. "Spread of epidemic disease on networks". In: *Phys. Rev. E* 66 (1 2002), p. 016128.

[62] M. E. J. Newman. "The structure and function of complex networks". In: *SIAM review* 45.2 (2003), pp. 167–256.

[63] H. Nguyen and R. Zheng. "On Budgeted Influence Maximization in Social Networks". In: *IEEE Journal on Selected Areas in Communications* 31.6 (2013), pp. 1084–1094. ISSN: 0733-8716.

[64] Manos Papagelis. "Refining Social Graph Connectivity via Shortcut Edge Addition". In: *ACM Trans. Knowl. Discov. Data* 10.2 (2015), p. 12.

[65] Nikos Parotsidis, Evaggelia Pitoura, and Panayiotis Tsaparas. "Centrality-Aware Link Recommendations". In: *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2016, pp. 503–512.

[66] Matthew Richardson and Pedro Domingos. "Mining Knowledge-sharing Sites for Viral Marketing". In: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD02. ACM, 2002, pp. 61–70.

[67] E. Rogers. *Diffusion of innovations*. Free Press, 1995.

[68] Thomas C Schelling. *Micromotives and macrobehavior*. Norton & Company, 2006.

[69] Daniel Sheldon, Bistra N. Dilkina, Adam N. Elmachtoub, Ryan Finseth, Ashish Sabharwal, Jon Conrad, Carla P. Gomes, David B. Shmoys, William Allen, Ole Amundsen, and William Vaughan. "Maximizing the Spread of Cascades Using Network Design". In: *CoRR* abs/1203.3514 (2012). URL: `http://arxiv.org/abs/1203.3514`.

[70] Youze Tang, Yanchen Shi, and Xiaokui Xiao. "Influence Maximization in Near-Linear Time: A Martingale Approach". In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. SIGMOD '15. ACM, 2015, pp. 1539–1554. ISBN: 978-1-4503-2758-9. DOI: 10.1145/2723372.2723734. URL: http://doi.acm.org/10.1145/2723372.2723734.

[71] Youze Tang, Xiaokui Xiao, and Yanchen Shi. "Influence Maximization: Near-optimal Time Complexity Meets Practical Efficiency". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. SIGMOD '14. ACM, 2014, pp. 75–86. ISBN: 978-1-4503-2376-5. DOI: 10.1145/2588555.2593670. URL: http://doi.acm.org/10.1145/2588555.2593670.

[72] Jeffrey Travers and Stanley Milgram. "An Experimental Study of the Small World Problem". In: *Sociometry* 32 (1969), pp. 425–443.

[73] T. Valente. *Network models of the diffusion of innovations*. Hampton Press, 1995.

[74] D.P. Williamson and D.B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[75] Xiaojian Wu, Daniel Sheldon, and Shlomo Zilberstein. "Efficient Algorithms to Optimize Diffusion Processes under the Independent Cascade Model". In: *NIPS Workshop on Networks in the Social and Information Sciences*. Montreal, Quebec, Canada, 2015.

[76] Zhi Yu, Can Wang, Jiajun Bu, Xin Wang, Yue Wu, and Chun Chen. "Friend recommendation with content spread enhancement in social networks". In: *Inf. Sci.* 309 (2015), pp. 102–118.

[77] Y. Zhang, A. Adiga, A. Vullikanti, and B. A. Prakash. "Controlling Propagation at Group Scale on Networks". In: *IEEE International Conference on Data Mining*. ICDM15. 2015, pp. 619–628.