universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Influence Maximization in Social Networks:

A survey on variants of the least cost influence problem"

verfasst von / submitted by

Agnes Rymarz, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2017 / Vienna 2017

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 920
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Quantitative Economics, Management and Finance
Betreut von / Supervisor:	Dr. Markus Leitner
Mitbetreut von / Co-Supervisor:	Dr. Mario Ruthmair

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich genannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.
Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Wien, Oktober 2017	
Ort, Datum	Agnes Rymarz

Abstract

The purpose of this thesis is to give an introduction to the theory of influence maximization in social networks and how this issue can be modeled mathematically. Information diffusion models have already existed before social networks became popular. In 1978 the important "Threshold model" was introduced by Granovetter.
From a marketing point of view influence maximization can be important when companies want to advertise their products via social networks. Investing in a few people at the beginning can trigger an information cascade and cause evermore people to get influenced. This viral marketing strategy builds on the word-of-mouth propaganda. The crucial question is: Who should companies address at the beginning to influence the maximum number of people? The "Target Set Selection" problem deals with this question.
An extension of the Target Set Selection problem is the Least Cost Influence Problem. The advantage of this variant is that companies can not only select people as a whole, but also target people partially which can be done e.g. by distributing discount coupons. The Least Cost Influence Problem can be summarized by the following question: which people should get how much initial partial incentives to influence a given fraction of people such that the incentives are minimized?

Zusammenfassung

Das Ziel dieser Arbeit ist es, eine Einführung in die Theorie der Maximierung von Einfluss in sozialen Netzwerken zu geben und zu zeigen, wie dieser Aspekt mathematisch modelliert werden kann. Modelle zur Informationsverbreitung hat es bereits gegeben, bevor soziale Netzwerke populär wurden. 1978 wurde eines der wichtigsten Modelle, nämlich das Threshold Model, eingeführt.
In der Marketingbranche kann Einflussmaximierung relevant sein, wenn Firmen ihre Produkte über soziale Netzwerke bewerben. In einige wenige Leute am Anfang zu investieren, kann eine Kaskade auslösen und dazu führen, dass immer mehr Leute beeinflusst werden. Diese viral-marketing-Strategie lebt von der Mundpropaganda. Die wesentliche Frage dabei ist, welche Leute am Anfang ausgewählt werden sollen, sodass am Ende die meisten Menschen beeinflusst sind. Das Target Set Selection Problem behandelt diese Frage.
Eine Erweiterung vom Target Set Selection Problem ist das Least Cost Influence Problem. Der Vorteil von dieser Variante ist, dass Firmen nicht nur Leute als Ganzes auswählen können, sondern auch nur zum Teil. Das kann zum Beispiel durch Verteilen von Rabattgutscheinen erreicht werden. Das Least Cost Influence Problem kann in einer Frage zusammengefasst werden: Welche Personen sollten am Anfang wieviel bezahlt bekommen um einen vorgegebenen Anteil an Personen zu beeinflussen, sodass die Zahlungen minimiert werden.

Acknowledgement

First of all, I want to thank my both supervisors for the great support at writing my thesis. Whenever I had a question or needed help they made effort to give me good advices. The working atmosphere was always very pleasant.
Furthermore, I want to thank my family, especially my parents, for supporting me in my education and making such a study possible. They always believe in me and my skills. Knowing to have such a strength backing is worth a lot.
Many thanks go to my friends who gave me always a good balance to my university life. It was very important for me to regain strength with my friends during the whole course of studies. They made the time of my study to an awesome era.

Contents

List of Figures

List of Algorithms

Abbreviations

TSS	Target Set Selection
LCIP	Least Cost Influence Problem
WTSS	Weighted Target Set Selection
TUM	totally unimodular
DAG	directed acyclic graph
OPOAO	Opportunistic One-Activate-One
DOAM	Deterministic One-Activate-Many
LCRB	Least Cost Rumor Blocking
LCRB-P	Least Cost Rumor Blocking under opportunistic model
LCRB-D	Least Cost Rumor Blocking under deterministic model
BFS	Breadth First Search
MCICM	Multi-Campaign Independent Cascade model
COICM	Campaign-Oblivious Independent Cascade model
EIL	eventual influence limitation
IML	Maximizing Influence while unwanted target users limited

1. Introduction

1.1 Motivation

Marketing is a very important feature for companies to advertise their products. Previously, customers have been seen as isolated individuals, but nowadays the interaction and potential influence among people play an important role. Domingos and Richardson [2001] are one of the first who considered not only the expected value from the sales to the customer itself, but also the so-called network value of customers which captures information about how much they influence other potential buyers. The reason why such a network value has to be taken into account is that people these days can influence a large number of other potential customers via social networks. In the past, the buying decision of a person mainly depended only on her/himself, but now, this decision depends on recommendations and references from family, friends and colleagues. Through social networks influencing gets easier. Social networks can be represented very well through graphs: nodes describe people and edges denote the connections between the individuals. If someone posts on a social network that he bought a certain product and is completely happy with it, all of his friends see that and may consider buying this product as well. This influence phenomenon is not only observed for products, but also for opinions, ideas, conventions and technologies. Furthermore, the spread of people one can reach via social networks is extremely large and thus social networks play an important role as a medium of information propagation. This new area of advertising, where the diffusion of innovations over a social network is crucial, is called viral marketing, because of its resemblance to the enlargement of an epidemic. Word-of-mouth advertising can be much cheaper than traditional marketing strategies since a company must only invest in the initial target set and then the process is left to its own. Furthermore, people appreciate personal advices or opinions more than those from a company directly. [Richardson and Domingos, 2002]

There are several authors who have been concerned with analyzing the dynamics of adoption in a network. In literature, this problem is referred to as the "influence maximization problem". Kempe et al. [2015] survey the fundamental models for the diffusion process

in networks such as the Threshold models and the Cascade models, which are shown in detail in Chapter 2. Whereas Domingos and Richardson [2001] model the problem in a probabilistic setting, Kempe et al. [2015] use a deterministic framework.

Many companies take advantage of knowing that people influence others and use a new viral marketing strategy to advertise their assortment. They target those individuals from whom they think they are highly influential and give them, for example, free samples. However, now the question is: who should they target? Which individuals would share the information about the product the most and can trigger an information cascade? The aim of this cascade is to cause evermore people to adopt. Since these people are part of a social network with complex relationships, this is not a trivial problem anymore and has become a central research topic. [Kempe et al., 2015]
Furthermore, companies certainly want to keep down their costs. Hence, the goal is to influence as many people as possible while minimizing the advertising costs. The decision problem of choosing the initial people who should get influenced is called the Target Set Selection (TSS) problem. There are various formal definitions and variants of the TSS problem which will be defined in Chapter 3.

Raghavan and Zhang [2015] describe a weighted version of the TSS problem where each node is associated with a weight which captures the property that different individuals need different levels of effort to adopt. The weighted version of the TSS problem is discussed shortly in Chapter 3.

In the course of the research some extensions of the TSS problem arose. An extension of the TSS problem is considered by Gunnec et al. [2013] and is called the Least Cost Influence Problem (LCIP). The advantage of this version is that companies are allowed to target people with partial incentives which means that they do not have a binary choice anymore to either target an individual or not. A company can decide either to target a person in the full amount, which can be interpreted as giving him a product for free, or target a person partially in terms of giving him a discount coupon. The advantage is that companies can maybe influence a wider spread of people with the same budget. [Gunnec et al., 2013] In Chapter 4 the LCIP is formally introduced and a way to solve the problem is presented.

Additionally, people often join not only one social network, but several networks. This is an important observations since it can be advantageous to influence a person which is part of more networks, because the person can transmit information from one network to

another network and hence reach a new range of people. Therefore, not only the influence power in one network has to be taken into account, but the problem has to be modeled in a new way. In Chapter 5 the corresponding approach of Zhang et al. [2016b] is described.

Furthermore, social networks serve not only for the diffusion of true information but rumors can spread very fast as well. Methods how these rumors can be stopped are sought, because misinformation can have serious consequences. [Fan et al., 2013] In Chapter 6, models which help to stop the spread of whisper are introduced.

1.2 Structure of the thesis

In Chapter 2 the basic models for information diffusion, such as the Threshold model and the Cascade model, are defined and explained. Some results about the approximability of the influence maximization problem for these models are summarized as well.

Chapter 3 is about the Target Set Selection (TSS) problem. First, the problem is defined and then some approximability results are summarized. The description of the Weighted Target Set Selection (WTSS) problem concludes the chapter.

The Least Cost Influence Problem (LCIP) is the topic of Chapter 4. At the beginning, the problem is defined and then some special cases where the underlying graphs are trees or complete graphs are considered. Two algorithms for special cases are explained in detail and illustrated with examples. Afterwards, it is described how the theory can be extended to general graphs.

The focus of Chapter 5 lies on the LCIP in multiple social networks. First, the problem is defined formally. Then, the core of the chapter is the study of so-called coupling schemes which aim to couple multiple networks into one network. This is advantageous since the theory for single social networks can now be applied. The coupling schemes are demonstrated with examples. At the end of the chapter, some results on the experimental work of Zhang et al. [2016b] are summarized.

In Chapter 6 the Least Cost Rumor Blocking (LCRB) problem is presented. At the beginning new diffusion models are introduced. Afterwards, the LCRB problem is defined formally. Some complexity results and possible solution approaches are explained as well. The chapter ends with the evaluation of the proposed solution approaches.

1.3 Preliminaries

In my thesis I assume basic knowledge about graph theory. The reader should be familiar with the concept of directed and undirected graphs as well as with subgraphs and the degree of a node. Relevant fundamentals can be found for example in West [2001], Bollobas [2012] and Trudeau [2013].

Since I will use mixed integer programming models I want to refer to Jünger et al. [2009], Conforti et al. [2014] and Garfinkel and Nemhauser [1972] who give a good introduction to integer programming models.

2. Basic models for the diffusion of information

To explain the way how diffusion of information works, mathematical models are needed. In this section, I will review the most important existing models, namely the Threshold model, the Cascade model, and their variants. These models build the foundation for later work, because later formulations of problems are based on such models. Kempe et al. [2003;2005] build a milestone in this research area and summarize these two conference articles in the revised work "Maximizing the spread of influence through a social network" that has been published in 2015. This chapter mainly builds upon this revised paper. Next, the influence maximization problem which asks for the initial active set with the most influence power is formally introduced. Kempe et al. [2015] define the influence maximization problem as the task to find a set of k nodes which leads to the highest expected number of influenced nodes at the end, where k is a given number. In general, the influence maximization problem is NP-hard to approximate within a factor of $n^{1-\varepsilon}$ where n is the number of nodes and $\varepsilon > 0$. [Kempe et al., 2015]

2.1 The Threshold model

Kempe et al. [2015] describe the fundamental Threshold model for networks which builds a basis for many other diffusion models. They refer to Granovetter [1978] and Schelling [1978] which were among the first who described Threshold models. Macy [1991], Valente [1995] and Young [2006] also work with related models.
The starting point of the model is a directed graph $G = (V, E)$ which represents a social network where the nodes are people and arcs the relationships between them. The nodes of this graph can either be inactive, which means that a person has not yet adopted the product or idea, or active, which says that a person is already influenced. Nodes can only switch from inactive to active but not in the other direction, which means that once a person is convinced of an innovation it can not change its opinion anymore. The goal is that as many nodes as possible become active. All models assume that the more neighbors are

active, the more likely the node itself becomes active.

2.1.1 The Linear Threshold model

In this model an incoming neighbor u of node v (i.e. $(u, v) \in E$) influences the node v with a weight $w(u, v) \in [0, 1]$. Each node v is endowed with a specific threshold $\theta(v) \in [0, 1]$, which is the boundary that has to be reached to activate node v. We can interpret the threshold as an individual level of convincement of the specific node. If the sum over the particular weights $w(i, v) \in [0, 1]$, where i runs through the set of active incoming neighbors of v, equals or exceeds the threshold, then v gets active. Formally, this is described as:

$$\sum_{i\, is\, an\, active\, neighbor\, of\, v} w(i, v) \geq \theta(v) \Rightarrow v\, gets\, active$$

This means that the overall influence from active neighbors must be larger or equal to the particular threshold to convince the person. One can now formulate a diffusion process as the following. We start with an initial active set of nodes and at every discrete time step t the active nodes stay active and other nodes may become active if their thresholds are reached. At every time step the state of the nodes are updated. The process stops if either all nodes are active or no more activations are possible. [Kempe et al., 2015]

The thresholds can either be a known deterministic value, a random value or have to be calculated as part of the optimization problem. However, for simplifying reasons, Kempe et al. [2015] assume that the thresholds $\theta(v)$ are uniformly distributed over the interval $[0, 1]$, so one can interpret the particular thresholds as the average over all possibilities. Kempe et al. [2015] show that the influence maximization problem is NP-hard for the Linear Threshold model with random thresholds, but can be approximated in a rather good way with a greedy approximation algorithm. Two further variants of the Threshold model will be discussed in the following.

2.1.2 The General Threshold model

In this model the activation procedure is the same as in the linear model, but with the only difference that the influence of the neighbors may not be linear but could depend on any monotone function f_v which assigns subsets of neighbors of node v to real numbers between 0 and 1. We need an initial condition $f_v(\varnothing) = 0$, which ensures that v cannot get active if it has no active neighbors. The node v gets active in step t if $f_v(S)$ exceeds the value of the uniformly chosen threshold $\theta(v)$ where S describes the set of active neighbors

of v at time $t-1$. The Linear Threshold model is a special case of the General Threshold model with the function $f_v(S) = min(1, \sum_{u \in S} w(u,v))$. [Kempe et al., 2015]

2.1.3 The Submodular Threshold model

The Submodular Threshold model is a special case of the General Threshold model, which requires the function $f_v(S)$ to be submodular. A function f_v is defined to be submodular if "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" [Kempe et al., 2015]. Thus, for all elements v and all pairs of sets $S \subseteq T$ we have: $f_v(S \cup v) - f_v(S) \geq f_v(T \cup v) - f_v(T)$. This special case makes sense since the influence of a particular node is lower when we add a node to a bigger set than to a smaller set. Mossel and Roch [2010] prove that there exists a polynomial-time algorithm which approximate the problem within a factor of $1 - \frac{1}{e}$ for this model.

2.2 The Cascade model

The Cascade models are based on research of interactive particle systems which are for example considered by Liggett [1985] and Durrett [1988]. Goldenberg et al. [2001a] and [2001b] use the approach of cellular automata methods to consider Cascade models in marketing settings. Easley and Kleinberg [2010] study cascading behavior in networks based on coordination games.
Kempe et al. [2015] define three types of Cascade models, namely the Independent, the General and the Decreasing Cascade model which are presented in the following.

2.2.1 The Independent Cascade model

This is the simplest among all Cascade models. As in the Threshold model we assume that nodes can either be active or inactive and once a node got active it stays active. The setting is in a discrete time frame and we start with an initial active set of nodes. An active node v has once the chance to influence an inactive neighbor w with a success probability $p_{v,w}$ which is part of the input. It is important to note that the success probability is independent from all previous activation attempts through other nodes. Either the influencing process succeeds and node w gets active, or node w stays inactive and node v can never try again to affect node w. However, other active neighbor nodes of w can try to influence w. If two nodes want to activate a certain node at the same time step, then the sequence of their attempts to influence that node is arbitrary. The process stops if no more activations

are possible. Kempe et al. [2015] show that it is NP-hard to approximate the influence maximization problem for the Independent Cascade model within a factor of $1 - \frac{1}{e}$.

2.2.2 The General Cascade model

The General Cascade model works like the Independent Cascade model with the only difference that the success probability now depends on the history. The propensity of a node to adopt an innovation may change with the number of nodes which have already tried to influence it. This is a natural extension since a person is more likely to adopt a product if more people in the surroundings already have adopted. Formally, Kempe et al. [2015] define an incremental function $p_v(u, S) \in [0, 1]$, where S describes the subset of v's neighbors that have already tried and failed to activate v, and $u \notin S$. Furthermore, the incremental function is required to be order-independent, which means that the sequence in which the other neighbors already have attempted does not change the success probability. Formally, let $S = \{u_1, u_2, \ldots, u_{|S|}\}$ be the set of neighbors which have already tried to activate v. Then, order-independent means that

$$\prod_{i=1}^{|S|}(1 - p_v(u_{\pi(i)}, \{u_{\pi(1)}, u_{\pi(2)}, \ldots, u_{\pi(i-1)}\}) = \prod_{i=1}^{|S|}(1 - p_v(u_{\varphi(i)}, \{u_{\varphi(1)}, u_{\varphi(2)}, \ldots, u_{\varphi(i-1)}\})$$

for any pairs π and φ of permutations of $\{1, \ldots, |S|\}$. If $p_v(u, S)$ is constant and does not depend on S, we obtain the Independent Cascade model. Kempe et al. [2015] show that the General Threshold model and the General Cascade model are equivalent by defining appropriate activation functions and success probabilities. However, the two distinct perspectives to look at the same random process are important for Kempe et al. [2015] for their further work to prove some statements about approximation guarantees.

2.2.3 The Decreasing Cascade model

Similar as in the Submodular Threshold model we have here also a diminishing influence when the size of the influencing set gets larger. The Decreasing Cascade model is a special case of the General Cascade model in which we require a further condition. Kempe et al. [2015] call this prerequisite the "diminishing influence condition": $p_v(u, S) \geq p_v(u, T)$ whenever $S \subseteq T$, which means that the success probability p_v is a non-increasing function with the set of neighbors who have already tried to activate v. Kempe et al. [2015] show that the Decreasing Cascade model is a special case of the Submodular Threshold model. For the Decreasing Cascade model Kempe et al. [2015] prove a similar approximation result as for the Independent Cascade model, in fact that there is a polynomial time algorithm that can approximate the problem within a factor of $1 - \frac{1}{e}$.

3. The Target Set Selection problem and its weighted version

3.1 The Target Set Selection problem

In this chapter, the Target Set Selection problem is formally defined and some results about the approximability of this problem are given. Since there is already a master's thesis from Chronowski [2014] which covers this problem and its hardness results in detail, this part will be rather short in this thesis.

3.1.1 Definition

There is not only one definition of the problem, but various authors described the TSS problem in several frameworks and with different proposals. The main features in which the different formulations vary are the underlying graphs and whether there are constraints for the target set respectively the set of activated nodes or not. A further deviation between the approaches is the threshold selection.

Chen [2009] bases his formulation of the TSS problem on the Linear Threshold model and uses deterministic threshold values, while Kempe et al. [2015] use random values. Chen [2009] formulates the problem as follows. Let $G = (V, E)$ be a directed graph, $deg(v)$ the degree of node $v \in V$ and $\theta(v) \in \mathbb{N}$ the threshold value for node $v \in V$ such that $1 \leq \theta(v) \leq deg(v)$. The process starts with all vertices inactive and we choose a subset of nodes to be active in the beginning. At every time step an inactive node v gets active if at least $\theta(v)$ of its neighbors are active. The process stops if no more activations are possible or if all nodes are active. Chen [2009] focuses on the task to find a target set of minimum size such that at the end of the process all or a certain fraction of nodes are active.

In contrast, Kempe et al. [2015] consider the problem to find a set of k initially active

nodes to maximize the influence over the network, where k is a given number. A further difference is that Kempe et al. [2015] focus on randomly chosen thresholds, different than Chen [2009] who considers the case of explicitly given deterministic thresholds.

Ackerman et al. [2010] define the TSS problem on a directed graph as the task to find an initially active target set, where its size may not exceed a given number k such that at least a given number of m nodes will be activated. Note that such a target set does not exist in every instance. Furthermore, they refer to the variant of the TSS problem defined by Chen [2009] as the minimum target set problem and to the variant by Kempe et al. [2015] as the maximum active set problem.

Nichterlein et al. [2013] formulate the TSS problem on an undirected graph $G = (V, E)$ in the following way. Given threshold function $\theta : V \to \mathbb{N}$ and an integer $k \geq 0$ is there a target set $S \subseteq V$ for G of size at most k that allows to activate all vertices in G?

3.1.2 Approximability

In the following section, some approximability results for variants of the TSS problem are given. Not to many details will be given, because the focus of this thesis lies rather on the models and not on the approximability results.

Chen [2009] provides a polylogarithmic lower bound on the approximation ratio for the TSS problem. Furthermore, he shows that the Target Set Selection problem with deterministic thresholds gets intractable if there are no simplifying assumptions. Chen [2009] shows that the problem remains NP-hard, even for the special case of a bounded bipartite graph where the thresholds are at most two.

A further case which is considered in the literature is the case of majority thresholds. This means that a node gets active if at least half of its neighbors have adopted, formally $\theta(v) = \lceil \frac{deg(v)}{2} \rceil$ for each node $v \in V$. Peleg [2002] proves that the TSS problem with majority thresholds is still NP-hard. Chen [2009] gives evidence that this special case does not admit a better approximation ratio than in the general case.

Dreyer and Roberts [2009] consider the case of constant thresholds which means that the threshold is the same for every node. They prove that the decision of choosing the target set in the TSS problem is NP-complete if the threshold value is at least three.

However, when the underlying graph is a tree, Chen [2009] provides an algorithm which

can solve the TSS problem optimally in polynomial-time.

Chiang et al. [2013] consider deterministic thresholds and require every node in the network to adopt. They analyze various types of graphs, for example block cactus graphs, chordal graphs and Hamming graphs. If the threshold value does not exceed two and the underlying graph is a chordal graph, then an optimal target set can be found in linear time.

If the underlying graph is a tree, the approximation results are the best. Knowing this fact Ben-Zwi et al. [2011] consider the treewidth of a graph which broadly speaking measures the degree of tree-likeness. A tree has treewidth one. They implement an algorithm for a graph with n vertices and a treewidth ω which runs in $n^{O(\omega)}$ time.

3.2 The Weighted Target Set Selection (WTSS) problem

Raghavan and Zhang [2015] introduce the Weighted Target Set Selection (WTSS) problem which is an extension of the TSS problem. Here, every vertex $u \in V$ is additionally endowed with a weight $c(u)$. This captures the fact that people can initially be bought with different values than their thresholds. As different people need a different level of persuasion to become an initial adopter, these weights are fitted for every user.

The WTSS and Least Cost Influence Problem (LCIP) (which will be presented in Chapter 4) are very similar, but differ in the fact that in the LCIP partial payments are allowed which are not possible in the WTSS problem. This means that in the WTSS problem either a company pays the full costs $c(u)$ or nothing. Furthermore, the WTSS problem requires a 100% adoption rate and all neighbors have equal influence on a certain node. Hence, the WTSS problem can be viewed as a special case of the LCIP. [Raghavan and Zhang, 2015]

Raghavan and Zhang [2015] propose an algorithm to solve the WTSS problem on trees similar to the dynamic programming algorithm which will be presented later to solve the LCIP on trees. This algorithm, which runs in $O(|V|)$ time, builds upon solving subproblems defined by star graphs and then use a backtracking method. Furthermore, Raghavan and Zhang [2015] extend the special case of trees to a more general setting.

Cordasco et al. [2015] also deal with the weighted version of the TSS problem. They interpret the weights $c(u)$ as costs which describe how much a user must receive to be convinced in the beginning. It is reasonable that some users have lower or higher costs

than others. For example, prominent persons or bloggers are more likely to have high costs, because they are very influential.
The problem is formulated by Cordasco et al. [2015] such that for a given network $G = (V, E)$, given thresholds $\theta(u)$ and given costs $c(u) : V \to \mathbb{N}$, the target set $S \subseteq V$ is sought for which the costs $C(S) = \sum_{u \in S} c(u)$ are minimal. The goal is that at the end the whole network is active. The influence factors (which are defined in Section 4.1) are all set to one which means that an active node influences a neighbor in the amount of one.

Cordasco et al. [2015] also propose an algorithm to solve the WTSS problem on trees. The algorithm is called $WTSS(G)$ and has similarities to the $TPI(G)$ algorithm which will be presented in Section 4.3.3. The starting point is the initial graph of the network and at each discrete time step a node is removed so that a certain function (which depends on the costs, thresholds and degrees of the nodes) is maximized. During the algorithm it can happen that a node u has a threshold value higher than its number of neighbors. Then, this node u is selected to be in the initial target set and the threshold values of its neighbors are decreased by one since the neighbors receive influence from u. Additionally, node u is removed from the graph. A further scenario that could occur is that the threshold of a node v has been decreased to zero. Then v is removed from the graph and again, its neighbors' thresholds are reduced by one. [Cordasco et al., 2015]

Algorithm $WTSS(G)$ returns for *any* graph G a target set and runs in $O(|E|log|V|)$ time. Furthermore, Cordasco et al. [2015] prove that the costs of the target set is constrained by $\sum_{u \in V} \frac{c(u)\theta(u)}{d(u)+1}$ where $c(u)$ describes the cost of node u. Cordasco et al. [2015] show that the $WTSS(G)$ solves the WTSS problem optimally if the input graph is a complete graph and whenever $c(u) \leq c(v)$ holds it also holds that $\theta(u) \leq \theta(v)$.

In experiments with different threshold settings, Cordasco et al. [2015] compare the $WTSS(G)$ algorithm with two alternative algorithms. One algorithm chooses the nodes in decreasing order with respect to the degree of the nodes and is called *DegreeInt*. The other algorithm is named *DiscountInt* and also selects the nodes in descending order of degree but additionally reduces the degree of its neighbor nodes by one.
In 17 of 18 networks the $WTSS(G)$ algorithm performs better than the others, while in only one both other algorithms return slightly better results.

4. The Least Cost Influence Problem

The Least Cost Influence Problem is an influence maximization problem where the objective is to find the minimum total amount of payments required to influence a given fraction of people in a network. It is an extension of the TSS problem and the crucial difference to the TSS problem is that here companies are allowed to target people with partial incentives. This means that they do not have the binary choice to either target a person or not, but can also target them partially. Thus companies are, for example, not only allowed to give products for free, but also to distribute discount coupons. The advantage is that instead of giving one person a product for free, the company can e.g. give away five 20%-coupons and hence reach a larger number of people. [Gunnec et al., 2013]

Gunnec et al. [2013] are one of the first who introduced the LCIP. They wrote a technical report in 2013 and then published an extended version of it in 2016. These two papers will be my main sources for this chapter.
An important prerequisite of the LCIP is the Share-Of-Choice Problem that has been considered in the literature by many authors, for example by Kohli and Krishnamurti [1989], Green and Krieger [1989] and Camm et al. [2006]. Gunnec and Raghavan [2016] introduce the LCIP as an attempt to include social network effects in the Share-Of-Choice Problem. Thereof they develop the LCIP.

Gunnec et al. [2016] define the LCIP in general and then focus on tree networks. For the special case that all neighbors have the same influence on a certain node and a 100% adoption rate is needed, they introduce two algorithms to solve the LCIP. They first describe a dynamic programming algorithm and then show that this algorithm can be viewed as a greedy algorithm. However, Gunnec et al. [2016] prove that the dynamic programming algorithm has a better worst-case running time than the greedy algorithm and that it is also applicable in the case of unequal influence. They also show that the LCIP on a tree with unequal influence from the neighbors can be reduced to the 0-1 knapsack problem and hence is NP-hard. Gunnec et al. [2016] provide a totally unimodular (TUM) integer linear programming (ILP) formulation for the LCIP on trees with equal influence and

show how this formulation can be extended to a ILP formulation for a general graph.

Cordasco et al. [2015] refer to the LCIP as an extension of the TSS problem where a company not only has the binary choice of giving someone a product for free or not, but also has the possibility to give a percentage discount. They call the problem "targeting with partial incentives" and use a target vector which determines how much incentive is given to which node. The purpose of targeting with incentives is that in the future the nodes can be easier influenced when their thresholds are reduced by these payments.
Cordasco et al. [2015] describe an algorithm to solve the LCIP for the whole network to adopt. The algorithm starts with the initial graph and in every iteration vertices are deleted by means of a certain parameter. From the remaining graph one can detect where the partial incentives are allocated. Furthermore, they concentrate on the special cases when the underlying graph is a complete graph or a tree, and show that in this two cases the target vector determined by the algorithm is optimal. In the instance of a tree, it is even possible to explicitly compute the cost of the optimal target vector, see Section 4.3.3.

A further work that has to be mentioned is "How to influence people with partial incentives" from Demaine et al. [2014]. In their work they extend the Submodular Threshold model from Kempe et al. [2015] to a fractional version where it is allowed to pay nodes with partial incentives. Demaine et al. [2014] show how the main results of the integral version can be adopted to the fractional version. Moreover, they find out that the two versions have basically the same computational complexity. However, in practice the fractional model yields almost always to a higher expected number of adopters.

4.1 Formal definition

We consider a social network as an undirected graph $G = (V, E)$ where the set of nodes $V = \{1, 2, \ldots, n\}$ represents the people in the network and the set of edges E denotes the connections between the people. Gunnec et al. [2016] use thresholds which constitute the amount of utility that has to be obtained to adopt a certain product. For vertex u the threshold is denoted by $\theta(u)$. The higher the threshold, the harder the node can be influenced. As before, a node is considered to be active if it already has adopted the product and inactive otherwise. The influence factor $d(u, v)$ describes how much an active neighbor v influences the node u. If there is no direct connection (no edge which connects them directly), then the influence factor is equal to zero. $p(u)$ describes the payment a node u gets as a tailored incentive. $\alpha \in [0, 1]$ denotes the fraction of nodes that we require to be active at the end. To formulate the problem formally, Gunnec et al. [2016] introduce

time periods, $t = 1, 2, \dots T$ and define $y_{ut} = 1$ if node u is active in time period t, and 0 otherwise. Let $N(u)$ be the set of neighbors of the node u. The process starts with all nodes inactive and the partial payments (incentives) for all nodes have to be chosen and payed. A node gets active if the sum of the total influence of the neighbors and the tailored incentive has reached its threshold. At every time step the states of the nodes are updated and the process stops if no more activations are possible. The objective is to pay a minimal amount of inducements such that at the end there are at least $\alpha|V|$ nodes which have adopted the product.
A mathematical formulation of the LCIP is shown in LCIP1.

LCIP1 [Gunnec et al., 2016]:

$$\text{Min} \sum_{u \in V} p(u) \tag{4.1}$$

$$\text{s. t. } y_{u0} = 0 \qquad \forall u \in V \tag{4.2}$$

$$p(u) + \sum_{v \in N(u)} d(u,v) \cdot y_{v(t-1)} \geq \theta(u) \cdot y_{ut} \qquad \forall u \in V, t = 2, \dots T \tag{4.3}$$

$$\sum_{u \in V} y_{uT} \geq \alpha |V| \tag{4.4}$$

$$\text{where } y_{ut} \in \{0, 1\} \text{ and } p(u) \geq 0 \qquad \forall u \in V,\ t = 1, 2, \dots T. \tag{4.5}$$

The first expression ensures that the sum over all incentives payed in the network is minimized. Constraint set (4.2) ensures that at the beginning all nodes are inactive. Constraint set (4.3) makes sure that if a node is active then the sum of incentives and the total influence for one node received from its neighbors is as least as high as its threshold. Condition (4.4) ensures that at least a given fraction α of all nodes have adopted.

4.2 Special case of tree networks

Gunnec et al. [2016] focus their work on the special case of tree networks. They show that under the additional assumption that the influence does not depend on the identity of the neighbor, the LCIP on tree networks is polynomially solvable. Furthermore, they formulate the LCIP on trees in a totally unimodular way and extend this formulation to general graphs. For the special case when the adoption rate is 100%, there are two interesting approximation results. When the influence from all neighbors is the same, the LCIP is polynomially solvable on trees. Whereas in the case of unequal influence the problem remains NP-hard. [Gunnec et al., 2016]

4.2.1 LCIP on trees with equal influence of the neighbors and 100% adoption

This section will focus on the special case of the LCIP on which the identity of the neighbors does not play a role, which means $d(u) = d(u, v),\ \forall v \in V$, 100% adoption is required, which means $\alpha = 1$, and when the underlying graph is a tree. First, star subnetworks are introduced and some properties about the problem are stated. Secondly, two algorithms from Gunnec et al. [2013] to solve the LCIP are presented. Lastly, the theory is demonstrated in two examples.

A star network has one root node and arbitrarily many leaf nodes. All leaf nodes are connected to the root node and have degree equal to one. Therefore, the degree of the root node equals the number of leaf nodes. [Gunnec et al., 2013]
To illustrate the concept of a star network consider Figures 4.1, 4.2 and 4.3. Note that the root node of a tree is not unique, but every node can serve as a root node.

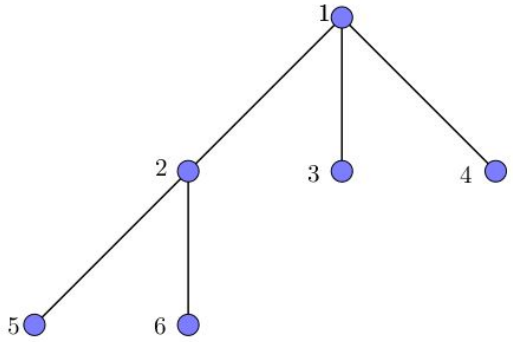

Figure 4.1: This is the initial network which we want to decompose in its star networks and node 1 is the root node.

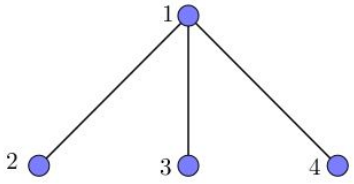

Figure 4.2: This is the first star subnetwork where root node 1 has degree 3 and the leaf nodes $2, 3$ and 4 have degrees 1.

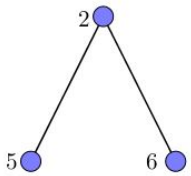

Figure 4.3: In the second star subnetwork node 2 is the root node and has degree 2 and leaf nodes 5 and 6 have degree 1.

Properties of the initial network and feasible solutions [Gunnec et al., 2013]

Property 1. In the initial network, it holds that $\theta(u) \geq d(u)$, $\forall u \in V$.

Proof. If the inequality $\theta(u) < d(u)$ holds this means that the influence which node u receives from one neighbor is sufficient for node u to adopt the product and the extra influence $\theta(u) - d(u)$ is not needed. So one can set the threshold equal to the influence factor what also means that one neighbor is sufficient. □

Property 2. For all leaf nodes it holds that $\theta(u) = d(u)$.

Proof. We assume, without loss of generality, that if all vertices in the set of neighbors $N(u)$ of node u adopt the product, then the overall influence from the neighbors will be at least as high as the threshold of node u and therefore affect u also to adopt. If this is not the case, it would mean that there must be an external payment with a value of $\theta(u) - |N(u)|d(u)$. However, this value is independent of the solution of the problem and hence one can omit it and assume that $\theta(u) \leq |N(u)|d(u)$, $\forall u \in V$. For a leaf node it holds that $|N(u)| = 1$, so it follows $\theta(u) \leq d(u)$ and we know from Property 1 $\theta(u) \geq d(u)$, so it must be the case that $\theta(u)$ equals $d(u)$. □

Property 3. If in a star network the root node adopts, then all leaf nodes will adopt too.

Proof. For each leaf node u one active neighbor is enough to adopt, because $\theta(u) = d(u)$. □

Property 4. If in a star network a leaf node is given any payments, then it gets incentives in the amount of its threshold.

Proof. Either the leaf node gets influenced from the root node and there is no need to pay incentives (see Property 3), or the leaf node does not get influenced from the root node. Since it is only connected to the root node, it would not get any other influence from the network and a payment below its threshold would not activate the leaf node. □

Property 5. The initial network can be reduced by canceling leaf nodes which have larger or equal thresholds than the root node ρ.

Proof. From Property 1 follows $\theta(\rho) \geq d(\rho)$. If a leaf node u has a larger or equal threshold than the root node ρ, then it follows that $\theta(u) \geq \theta(\rho) \geq d(\rho)$. Giving leaf node u incentives of the amount $\theta(u)$ which is greater or equal than the influence $d(\rho)$ is inefficient. □

Property 6. If a network only consists of two people, it is better to pay the one with the lower threshold.

Proof. The other person would adopt automatically from the influence. □

Property 7. An upper bound of incentive payments would be to pay all nodes except one.

Proof. Since all of its neighbors will be active, the omitted node would also adopt, because of the influence it received from the neighbors. □

Property 8. A lower bound of incentive payment could be to pay the node with the least threshold.

Proof. Paying the node with the lowest threshold could cause a cascade and activate all other nodes. □

For better understanding the LCIP, consider Figure 4.4. In this tree, each node u is associated with two numbers, its threshold $\theta(u)$ and the influence factor $d(u)$ for which $d(u) = d(u, v)$ holds for all $v \in V$. For the leaf nodes, the threshold and the influence factor is the same (Property 2). Now and in all following examples of this chapter, the value above the node describes the threshold of the node and the value below is the influence factor of the node. The purpose of the LCIP is now to find a cheapest way to get all nodes active, i.e. decide which nodes should get incentives to achieve that at the end all adopt the product and the overall incentives are as sparse as possible. A trivial solution is to pay node 1 its threshold $\theta(1) = 8$ and node 2 its threshold $\theta(2) = 6$, the other nodes would also adopt, because the influence from these two nodes are enough. This solution costs 14. However, maybe there are less expensive solutions. For example, paying node 3 its threshold $\theta(3) = 1$ leads to an activation of node 3 and decrease node 1's threshold from 8 to $\theta(1) - d(1) = 8 - 5 = 3$. Then paying 3 monetary units to node 1 causes node 1 to adopt and reduce the threshold of node 2 from 6 to $\theta(2) - d(2) = 6 - 4 = 2$. Furthermore, node 4 adopts automatically because of Property 3. Paying partial incentives in the amount of 2 to node 2 induces node 2 to adopt, and again all leaf nodes (i.e. nodes 5 and 6) adopt as well. This is a cheaper solution because the total costs are equal to 6. In

the following two algorithms it is described how to identify a cheapest solution.

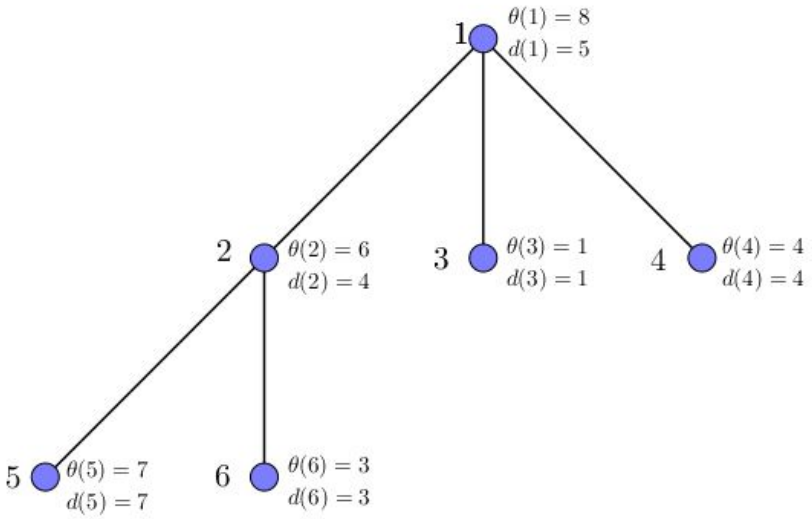

Figure 4.4: A tree network

4.2.2 Dynamic Programming Algorithm

The underlying graph is still a tree, all neighbors of a node have the same influence and the required adopting rate is still 100%. The dynamic programming algorithm proposed by Gunnec et al. [2013] divides up into three sub problems which are optimally solved separately. In this algorithm, the star subnetworks and their connection to the rest of the network, the so called "connector link", plays an important role. To solve a star we assume that influence travels through this connector link to the star network. Once a star is solved, its information is compressed into a single node and this node becomes a leaf node for the next star. When the last star is solved, a backtracking method is used to identify the nodes which have received incentives. [Gunnec et al., 2013] Before the algorithm begins, the tree is decomposed in star networks. The details are explained after the pseudocode given in Algorithm 1.

SolveStar

Gunnec et al. [2013] consider a star subnetwork and the goal is to find the cheapest solution to get all nodes in the star active. Since the leaf nodes will automatically adopt if the root node adopts, the focus is on how one can get this root node active. A trivial solution is to pay the root node incentives in the amount of its current threshold, but maybe there is

Algorithm 1: Dynamic Programming Algorithm [Gunnec et al., 2013]

1. **begin**
2. **for each** star network **do**,
3. SolveStar (see Algorithm 2)
4. CompressStar (see Algorithm 3)
5. TotalCost (see Algorithm 4)
6. **end**

a more preferable solution. Assuming the root node is influenced through the "connector link" by its influence factor $d(\rho)$, we can update its threshold to $\theta(\rho) = \theta(\rho) - d(\rho)$. The reason for doing this becomes clear below. Furthermore, we can neglect leaf nodes which have a larger threshold than the root node's influence factor and threshold. One would never pay such a leaf node because it would be cheaper to directly give incentive to the root node. So we can omit these leaf nodes and collect all other leaf nodes with thresholds smaller than the minimum of $\theta(\rho)$ and $d(\rho)$ in the set S. In the next step, we arrange the remaining leaf nodes in increasing order of their thresholds and pay them incentives in this order to reduce the root's threshold. Now, there are two cases of how much incentives should be given to the leaf nodes, if the set S is larger than $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil$. In the first case, the first $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil$ leaf nodes in S are paid and this influence is enough to convince the root node to adopt. Whereas in the second case, the first $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil - 1$ leaf nodes in S are paid and the remaining amount is paid to the root node directly. These two cases are compared and the case with the minimum cost is chosen. Other cases cannot be optimal because the remaining leaf nodes have higher thresholds than the chosen leaf nodes. If S is the empty set or is lower than $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil$, all nodes in S are paid incentives and the remaining amount, to get the root node active, is paid to the root node directly again. [Gunnec et al., 2013]

To demonstrate the difference between the two cases, consider Figure 4.5 where Case 1 is advantageous and Figure 4.6 in which Case 2 is demonstrated. Nodes which get payments are colored in pink.

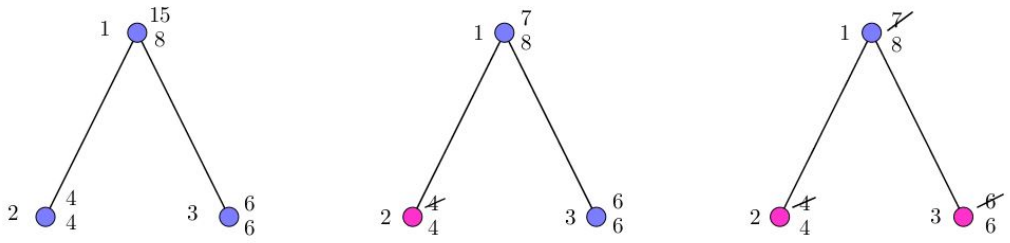

Figure 4.5: Case 1: $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil$ leaf nodes in S are paid

Case 1 (See Figure 4.5): Root node 1 has an (already updated) threshold of $\theta(1) = 15$ and an influence factor of $d(1) = 8$. For the leaf nodes it holds that $\theta(2) = d(2) = 4$ and $\theta(3) = d(3) = 6$. Since $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil = \lceil \frac{15}{8} \rceil = 2$ both leaf nodes get their thresholds payed. First, node one gets 4 monetary units, this causes the threshold of node 1 to reduce to $\theta(1) - d(1) = 15 - 8 = 7$. Then node 3 gets active by obtaining 6 monetary units. Since the influence factor of node 1 is 8, and its current threshold is only 7, it also adopts. The cost of this procedure is $4 + 6 = 10$. If instead only node 2 gets its threshold payed and root node 1 gets partial incentives in the amount of 7 monetary units, the cost would be $4 + 7 = 11$. So in this case it would be better to pay both leaf nodes and do not give any incentives to the root node.

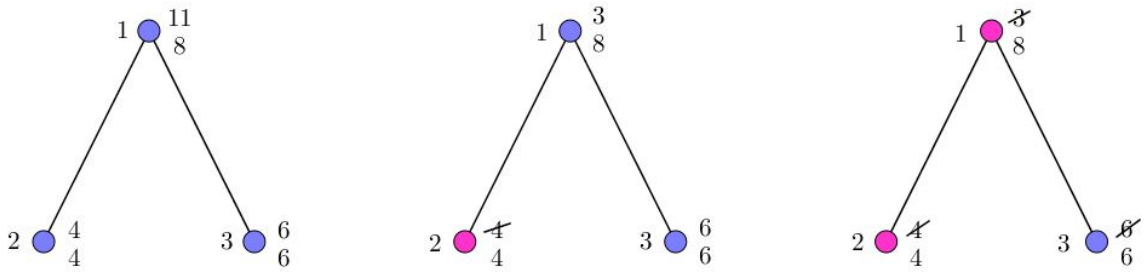

Figure 4.6: Case 2: $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil - 1$ leaf nodes in S are paid

Case 2 (See Figure 4.7): In this case, the values for the nodes are the same as above, but with the only difference that the threshold of root node 1 is $\theta(1) = 11$. The number of leaf nodes which get incentives is now $\lceil \frac{\theta(\rho)}{d(\rho)} \rceil - 1 = \lceil \frac{11}{8} \rceil - 1 = 2 - 1 = 1$. Since node 2 has a lower threshold than node 3, node 2 gets paid. Consequently, the threshold of node 1 gets updated to $11 - 8 = 3$. Now the root gets partial incentives in the amount of 3 monetary units and the effect is that the root node adopts and therefore, leaf node 3 as well. The costs to solve this star are $4 + 3 = 7$. However, if the incentive is not paid to the root node directly, but instead node 3 gets its threshold paid, the cost would be $4 + 6 = 10$. Thus, in this case it is cheaper to give incentives to the root directly.

The algorithm is not shown for each star separately with different notation, but generally for a star subnetwork. V denotes the set of nodes, L the set of leaf nodes, S^* the set of leaf nodes which have received incentives, ρ describes the root node of the star and C the cost of the star. In the array P, the given partial incentives are saved. The SolveStar algorithm is given in Algorithm 2.

Algorithm 2: SolveStar [Gunnec et al., 2013]

1. Update threshold for the root, $\overline{\theta}(\rho) = \theta(\rho) - d(\rho)$.
2. Let $S = \{u \,|\, \theta(u) < min\{\overline{\theta}(\rho),\, d(\rho)\},\, u \in L\}$.
3. Order nodes in S with respect to $\theta(u)$, $u \in S$ in ascending order
4. If $|S| \geq \lceil \frac{\overline{\theta}(\rho)}{d(\rho)} \rceil$, cost of the star, C, equals to the minimum of the following;
5. Select the first $\lceil \frac{\overline{\theta}(\rho)}{d(\rho)} \rceil$ nodes in S and give incentives equal to their threshold.
6. Select the first $\lceil \frac{\overline{\theta}(\rho)}{d(\rho)} \rceil - 1$ nodes in S and give incentives equal to their threshold.
   Pay the remaining to the root, $P[\rho] = \overline{\theta}(\rho) - (\lceil \frac{\overline{\theta}(\rho)}{d(\rho)} \rceil - 1)d(\rho)$.
7. Else cost of the star, C, equals,
8. For all nodes in S, give incentives equal to their threshold. Pay the remaining to the root, $P[\rho] = \overline{\theta}(\rho) - |S|d(\rho)$.
9. Remove the selected nodes from the set S and place them in set S^*.

CompressStar

In the SolveStar algorithm the cost of a star is calculated while the purpose of the CompressStar algorithm is to convert the star in a single node u with a new threshold $\theta'(u)$ and a new influence factor $d'(u)$. Since the compressed node becomes a leaf node in the next star, the threshold and the influence factor have the same value. To calculate these new threshold, Gunnec et al. [2013] distinguish between two cases:

1. Case: Remaining threshold at the root node.

2. Case: Threshold of a leaf node that is in S.

In the first case we have to further differentiate between two types which correspond to the two cases in the SolveStar algorithm (pay partial incentives to the root node or not). In the first type there is a partial payment to the root node, which means that the threshold for the root node is covered exactly and the new threshold of the compressed node is just the influence factor $d(\rho)$. The second type describes the case with no partial incentives given to the root node directly, which means that the influence comes only from the leaf nodes, but this influence may transcend the current threshold of the root node. When determining the new threshold, this excess influence has to be subtracted from the influence factor, i.e., $\theta'(\rho) = \overline{\theta}(\rho) + d(\rho) - |S^*|d(\rho)$. [Gunnec et al., 2013]

In the second case, the threshold of the new compressed node is simply the threshold of a leaf node in set S: Remember that at the end of the SolveStar algorithm the set S describes the nodes which have not obtained any incentives. Since there is not always a node remaining in the set S, this case is not always possible. However, sometimes this could be a cheaper way, because the threshold of a node in S is always lower than the root's influence factor. [Gunnec et al., 2013] All cases will be illustrated in the following using small examples that are based on the instances given in Figures 4.7, 4.8 and 4.9.

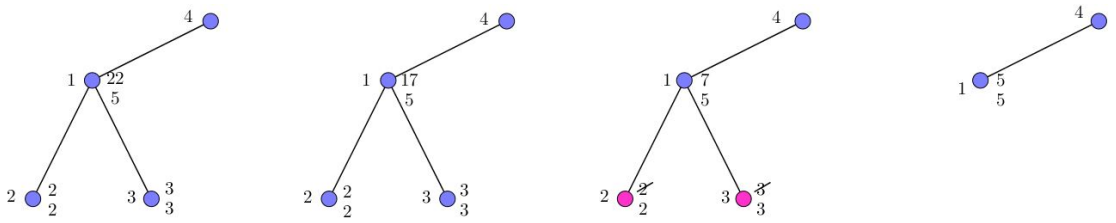

Figure 4.7: Case 1, Type 1

Case 1, Type 1 (See Figure 4.7): The nodes $1, 2$ and 3 describe a star subnetwork that is connected to node 4 (and a probably existing remaining graph) by a "connector link". Since the connector link is assumed to be realized, in the first step the threshold of node 1 is updated from $\theta(1) = 22$ to $\overline{\theta}(1) = 22 - 5 = 17$. In the next step the leaf node 2 gets paid the amount of its threshold, which has the effect that the threshold of the root node decreases by $d(1) = 5$ to $17 - 5 = 12$. Then, leaf node 3 gets incentives in the amount of 3 which reduces the threshold of node 1 again by 5. Now node 1 has a threshold of $12 - 5 = 7$. Since all leaf nodes (2 and 3) have adopted, the root node will also adopt. Hence, we only need the realized assumed connector link influence in the amount of $d(1) = 5$, so the new threshold and influence factor of the compressed node equals 5.

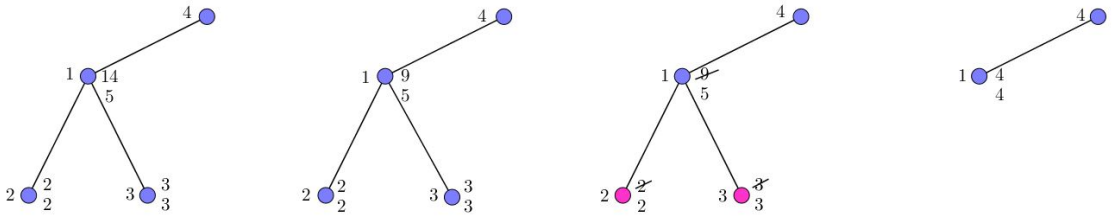

Figure 4.8: Case 1, Type 2

Case 1, Type 2 (See Figure 4.8): In this case the setting is the same as above, but now the threshold of the root node is 14 which is updated in the first step to $14 - 5 = 9$, due to the connector link. Here we have no partial payments to the root node, because the influence from the leaf nodes is enough to get the root active since $2 \cdot d(1) = 10 > 9$. Leaf node 2 gets payments in the amount of 2 which reduces the threshold of the root node to $9 - 5 = 4$. Then, leaf node 3 obtains incentives in the amount of 3 which again reduces the threshold of node 1 by 5 which causes the root node to adopt. We obtain the new threshold by applying the formula $\theta' = \overline{\theta}(\rho) + d(\rho) - |S^*|d(\rho) = 9 + 5 - 2 \cdot 5 = 4$.

Case 2 (See Figure 4.9): We have the same setting as before, but with a different threshold and influence factor for the root. The threshold of the root node is 13 and the influence

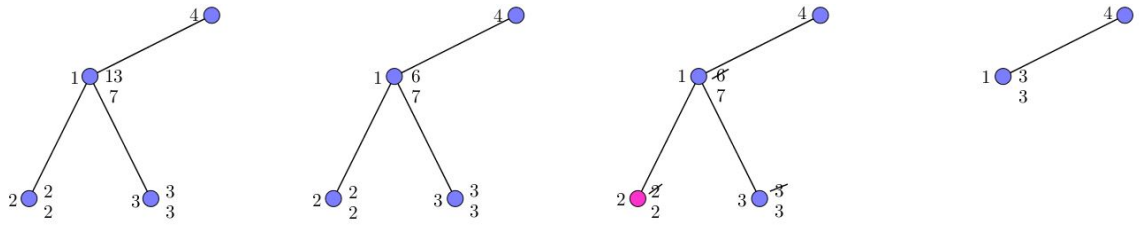

Figure 4.9: Case 2

factor is 7 which is first updated to $\overline{\theta}(1) = 13 - 7 = 6$ due to the connector link. After that, node 2 gets paid its threshold and hence gets active. Furthermore, it causes node 1 also to adopt, because the influence factor of node 1 is 7 and its current threshold is 6, so receiving influence from one node is sufficient. Now, the important point is that paying leaf node 3 its threshold $\theta(3) = 3$ is advantageous. So, the cost is $2 + 3 = 5$. Otherwise the cost would be calculated as in case 1 type 2: $\overline{\theta}(\rho) + d(\rho) - |S^*|d(\rho) = 6 + 7 - 1 \cdot 7 = 6$. Hence, it is cheaper to set the threshold of the compressed node equal to 3.

Let $B[u] = v$ be a pointer which means that the node u is represented by node v after the star in which node u occurs is already compressed. The initial condition $B[u] = u$ says that in the beginning every node points at itself. The pseudocode for the CompressStar algorithm is shown in Algorithm 3.

Algorithm 3: CompressStar [Gunnec et al., 2013]

1. New threshold $\theta' = min\{d(\rho), \overline{\theta}(\rho) + d(\rho) - |S^*|d(\rho), \min\limits_{u \in S} d(u)\}$.
2. If $\theta' = \min\limits_{u \in S} d(u)$ and $u' = arg\min\limits_{u \in S} d(u)$, then $B[\rho] = B[u']$.
3. New influence factor $d' = \theta'$.

TotalCost

With the CompressStar algorithm, stars are compressed into a single node until there is only one star left. The last star is also solved with this method, but now the threshold of the root node has not to be updated in the beginning, because there is no external influence anymore. The total cost of the LCIP is the sum of all costs for each star subnetwork. An important step is to trace back the distribution of the costs to identify the nodes which have received incentives. The set $\overline{S}$ describes the union of all sets which contain the nodes which are given incentives (S^*). Leaf nodes which are selected in the algorithm are paid in the amount of their threshold. Furthermore, if a node u is in the set $\overline{S}$ and it holds that $B[u] = v$ which means that u is represented by v then leaf node v gets paid its threshold. A node v which is not a leaf node in the initial network must be a root for some star network and hence gets paid its new compressed threshold $\theta'(v)$ if there is a

node u in the set $\overline{S}$ which points at node v. All nodes which get partial incentives can be determined by the vector P and are represented in the set $\overline{P}$ and get payments in the amount of $P[B[u]]$. [Gunnec et al., 2013]
The TotalCost algorithm is now given in Algorithm 4.

Algorithm 4: TotalCost [Gunnec et al., 2013]

1. Cost equals $\overline{C} = \sum_{all\ stars} C$.
2. Let $\overline{S} = \bigcup_{all\ stars} S^*$.
3. If $u \in \overline{S}$ and $B[u] \in L$, then pay $\theta(B[u])$ to node $B[u]$.
4. Else Pay $\theta'(B[u])$ to node $B[u]$.
5. Let $\overline{P} = \{u \mid P[u] > 0\}$ with $P[u]$ as in Algorithm 2
6. If $u \in \overline{P}$, Pay $P[u]$ to node u.

Gunnec et al. [2016] revise their dynamic programming algorithm. The SolveStar and CompressStar algorithms are combined into one algorithm which is called StarHandling. A backtracking method to identify which vertices get how much incentives is used as well. Furthermore, they prove that the algorithm solves the problem optimally in $O(|V|)$ time.

Example 4.1. *We consider the initial given graph in Figure 4.10. The progress of the algorithm is illustrated with Figure 4.11 and 4.12. First, the star with leaf nodes* $1, 2, 3$ *and* 4 *is compressed into node* 5 *, then the star with leaf nodes* 11 *and* 12 *is compressed into node 10, and then the last star is solved. The thresholds and influence factors are given in Table 4.1.*

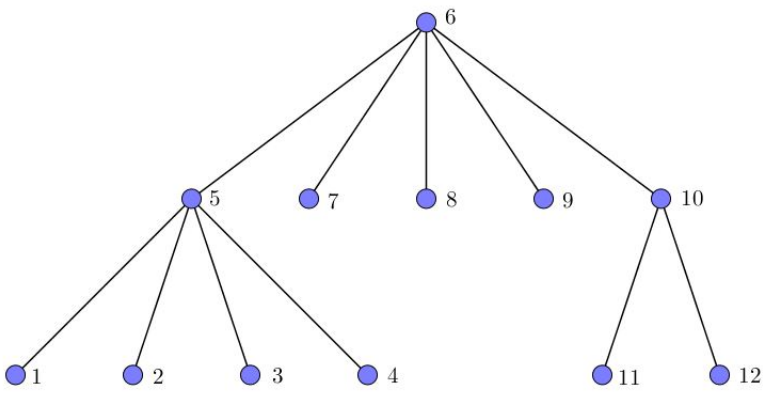

Figure 4.10: Initial given graph

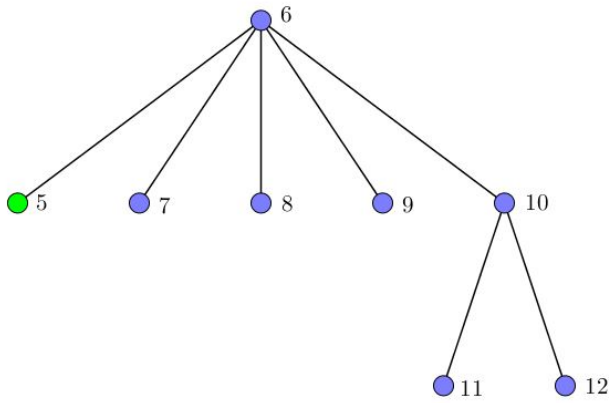

Figure 4.11: Star compressed into node 5

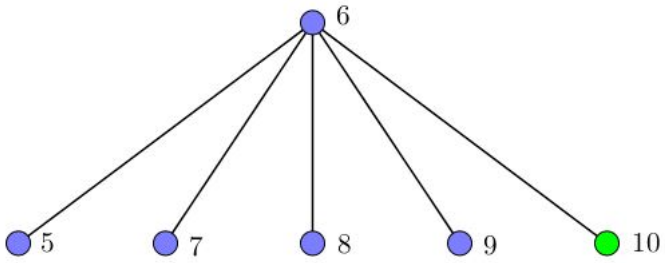

Figure 4.12: Star compressed into node 10

The numeration of the steps which are performed in the particular algorithm corresponds to the number of this particular step in the pseudocode of the algorithm. At the beginning it holds that $B[u] = u$ *and* $P[u] = 0\,, \forall u \in V$*. We start with the SolveStar algorithm for the star with leaf nodes* $1, 2, 3$ *and* 4 *and root node* 5*.*

SolveStar:
$V = \{1, 2, 3, 4, 5\}$, $L = \{1, 2, 3, 4\}$, $\rho = 5$
1. $\overline{\theta}(5) = 29 - 10 = 19$
2. $S = \{1, 2, 3, 4\}$
3. $S = \{4, 2, 3, 1\}$ *(note that the notation is not chosen well, but in the interest of the example this problem is disregarded)*
4. $|S| = 4 > \lceil \frac{19}{10} \rceil = 2$
5. $\theta(4) + \theta(2) = 6$ *vs.*
6. $\theta(4) + (\overline{\theta}(5) - (2-1)d(5)) = 2 + (19 - 1 \cdot 10) = 2 + 9 = 11$
min$\{6, 11\} = 6$*, then* $C = 6$
9. $S = \{3, 1\}$, $S^* = \{4, 2\}$

node	1	2	3	4	5	6	7	8	9	10	11	12
threshold	9	4	7	2	29	40	12	11	15	21	1	5
influence factor	9	4	7	2	10	13	12	11	15	6	1	5

Table 4.1: Threshold and influence factors for the nodes

CompressStar:

1. $\theta' = min\{d(5), \overline{\theta}(5)+d(5)-|S^{\cdot}|d(5), d(3), d(1)\} = min\{10, 19+10-2\cdot 10, 7, 9\} = 7$

2. $B[5] = B[3] = 3$

3. $d' = \theta' = \theta'(5) = d'(5) = 7$

SolveStar:

$V = \{10, 11, 12\}$, $L = \{11, 12\}$, $\rho = 10$

1. $\overline{\theta}(10) = 21 - 6 = 15$

2. $S = \{11, 12\}$

3. $S = \{11, 12\}$

7. $|S| = 2 < \lceil \frac{15}{6} \rceil = 3$

8. $P[10] = \overline{\theta}(10) - |S|d(10)$

$C = \theta(11)+\theta(12)+P[10] = \theta(11)+\theta(12)+\overline{\theta}(10)-|S|d(10) = 1+5+15-2\cdot 6 = 9$

9. $S = \emptyset$, $S^* = \{11, 12\}$

CompressStar:

1. $\theta' = min\{d(10), \overline{\theta}(10)+d(10)-|S^*|d(10)\} = min\{6, 15+6-2\cdot 6\} = min\{6, 9\} = 6$

3. $d' = \theta' = d'(10) = \theta'(10) = 6$

SolveStar: (last star)

$V = \{5, 6, 7, 8, 9, 10\}$, $L = \{5, 7, 8, 9, 10\}$, $\rho = 6$

1. $\theta(6) = 40$

2. $S = \{5, 7, 8, 10\}$

3. $S = \{10, 5, 8, 7\}$

4. $|S| = 4 \geq \lceil \frac{40}{13} \rceil = 4$

5. $\theta(10) + \theta(5) + \theta(8) + \theta(7) = 6 + 7 + 11 + 12 = 36$ *vs.*

6. *(*$P[6] = \theta(6) - (4-1) \cdot d(6)$*)*

$\theta(10) + \theta(5) + \theta(8) + P[6] = \theta(10) + \theta(5) + \theta(8) + \theta(6) - (4-1) \cdot d(6) =$

$6 + 7 + 11 + 40 - 3 \cdot 13 = 25$

min$\{36, 25\} = 25$*, then* $C = 25$

9. $S = \{7\}$, $S^* = \{10, 5, 8\}$

TotalCost:

1. $\overline{C} = \sum_{all\ stars} C = 6 + 9 + 25 = 40$
2. $\overline{S} = \bigcup_{all\ stars} S^* = \{4, 2, 11, 12, 10, 8, B[5] = 3\}$
3. *Pay leaf nodes* $4, 2, 11, 12, 8, 3$ *full incentives in the amount of* $2, 4, 1, 5, 11, 7$
4. *Pay node* 10 *(not leaf node) in the amount of* $\theta'(10) = 6$
5. $\overline{P} = \{10, 6\}$
6. *Pay nodes* 10 *and* 6 *partial incentives in the amount of* $P[10] = 3$ *and* $P[6] = 1$.

4.2.3 Greedy Algorithm

In every step of the greedy algorithm described by Gunnec et al. [2016] the node with the smallest minimum of threshold and influence factor receives payments in the amount of its threshold. If several nodes have the same minimum of influence factor or threshold, one is selected arbitrarily. Then, the thresholds of the neighbor nodes are updated which means that they are lowered by the amount of their influence factors. This could cause a neighbor node to get active as well and induce a chain reaction. Afterwards, the active nodes are eliminated and the graph consists again only of inactive nodes, and the process is repeated until all nodes are active. Example 4.2 illustrate the greedy algorithm.

Example 4.2. *In this example we consider the same situation as in Example 4.1, namely Figure 4.10 and Table 4.1. The numbers at the left stand for the iteration steps of the greedy algorithm.*

1. $\min_{u \in V} \{\theta(u), d(u)\} = d(11) = 1$
   Set of buyers $= \{11\}$
   update threshold $\theta(10) = 21 - 6 = 15$
2. $\min_{u \in V \setminus \{11\}} \{\theta(u), d(u)\} = d(4) = 2$
   Set of buyers $= \{11, 4\}$
   update threshold $\theta(5) = 29 - 10 = 19$
3. $\min_{u \in V \setminus \{11,4\}} \{\theta(u), d(u)\} = d(2) = 4$
   Set of buyers $= \{11, 4, 2\}$
   update threshold $\theta(5) = 19 - 10 = 9$
4. $\min_{u \in V \setminus \{11,4,2\}} \{\theta(u), d(u)\} = d(12) = 5$
   Set of buyers $= \{11, 4, 2, 12\}$
   update threshold $\theta(10) = 15 - 6 = 9$
5. $\min_{u \in V \setminus \{11,4,2,12\}} \{\theta(u), d(u)\} = d(10) = 6$*, but its threshold is* 9*, so node* 10 *gets a payment in the amount of* 9.
   Set of buyers $= \{11, 4, 2, 12, 10\}$

update threshold $\theta(6) = 40 - 13 = 27$

6. $\min\limits_{u \in V \setminus \{11,4,2,12,10\}} \{\theta(u), d(u)\} = d(3) = 7$
   Set of buyers $= \{11, 4, 2, 12, 10, 3\}$
   update threshold $\theta(5) = 0$
   Set of buyers $= \{11, 4, 2, 12, 10, 3, 5\}$
   update threshold $\theta(1) = 0$
   Set of buyers $= \{11, 4, 2, 12, 10, 3, 5, 1\}$
   update threshold $\theta(6) = 27 - 13 = 14$
7. $\min\limits_{u \in V \setminus \{11,4,2,12,10,3,5,1\}} \{\theta(u), d(u)\} = d(8) = 11$
   Set of buyers $= \{11, 4, 2, 12, 10, 3, 5, 1, 8\}$
   update threshold $\theta(6) = 14 - 13 = 1$
8. $\min\limits_{u \in V \setminus \{11,4,2,12,10,3,5,1,8\}} \{\theta(u), d(u)\} = \theta(6) = 1$
   Set of buyers $= \{11, 4, 2, 12, 10, 3, 5, 1, 8, 6\}$
   update threshold $\theta(7) = 0$ *and threshold* $\theta(9) = 0$
   Set of buyers $= \{11, 4, 2, 12, 10, 3, 5, 1, 8, 6, 7, 9\} = V$

Costs = $1 + 2 + 4 + 5 + 9 + 7 + 11 + 1 = 40$

Gunnec et al. [2016] prove that the greedy algorithm solves the LCIP on a tree optimally in $O(|V|log|V|)$ time. Hence, the dynamic programming algorithm has a better worst-case running time than the greedy algorithm.

4.2.4 Totally unimodular (TUM) formulation of the LCIP on trees

The formulation of the LCIP defined in Section 4.1 is based on time periods. Gunnec et al. [2016] propose a further formulation of the LCIP where they consider a propagation network with directed influence which is shown in LCIP2.

LCIP2 [Gunnec et al., 2016]:

$$\text{Min} \sum_{u \in V} p(u) \tag{4.6}$$

$$\text{s. t. } y_{uv} + y_{vu} = 1 \qquad \forall \{u, v\} \in E \tag{4.7}$$

$$p(u) + \sum_{v \in N(u)} d(u) \cdot y_{vu} \geq \theta(u) \qquad \forall u \in V \tag{4.8}$$

$$p(u) \geq 0 \qquad \forall u \in V \tag{4.9}$$

$$y_{vu} \in \{0, 1\} \qquad \forall v \in V,\ u \in N(v). \tag{4.10}$$

The objective is the same as in LCIP1 and ensures that the partial incentives paid to the nodes are minimized. The binary variable y_{uv} describes whether u influences v ($y_{uv} = 1$) or not ($y_{uv} = 0$). Constraint set (4.7) makes sure that if there is an edge which connects node u and v, either u influences v or u is influenced by v. In constraint set (4.8) it is captured that for every node in the graph the sum of the influence received from neighbors and partial payments exceeds or is equal to the node's threshold.

In the case that constraint set (4.8) is satisfied at equality, one can write $\theta(u) - \sum_{v \in N(u)} d(u) y_{vu}$ instead of $p(u)$ and so $p(u)$ would vanish and the remaining constraint set (4.7) is TUM [Gunnec et al., 2016]. However, this equality does not always hold, so Gunnec et al. [2016] distinguish between three types of influence.
Type H receives influence $d(u)$.
Type L has incoming influence of $l(u) = \theta(u) - (g(u) - 1)d(u)$.
Type Z has no incoming influence.

Let $g(u) = \lceil \frac{\theta(u)}{d(u)} \rceil$ be the number of active neighbors needed to get node u active. Now, there is a case differentiation between $g(u) \geq 2$ and $g(u) = 1$. Note that $g(u) < 1$ is impossible since $\theta(u) \geq d(u), \forall u \in V$. For Case 1 ($g(u) \geq 2$) consider Figure 4.13.

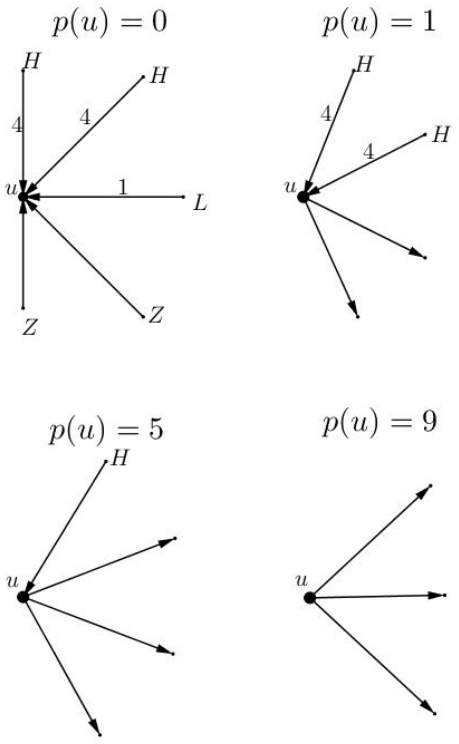

Figure 4.13: Case 1

The threshold value of u equals $\theta(u) = 9$, the influence factor is $d(u) = 4$, hence $g(u) = \lceil \frac{\theta(u)}{d(u)} \rceil = \lceil \frac{9}{4} \rceil = 3$, and $l(u) = b(u) - (g(u) - 1)d(u) = 9 - (3 - 1) \cdot 4 = 1$. If node u receives no partial payments, e.g. $p(u) = 0$, then it gets $g(u) - 1 = 2$ times the influence of type H and once the influence of type L. In the case when u gets payments in the amount of one monetary unit, e.g. $p(u) = 1$, then getting two times $d(u)$ as an incoming influence from type H is enough to convince node u to adopt. The next relevant scenario is $p(u) = 5$, in which u gets only additional influence from one arc with type H. The last case is when node u receives payments in the amount of 9 monetary units and no influence from any nodes. Generally, there are $g(u) + 1$ possible opportunities of partial payments, e.g., payments in the amount of 0 and obtaining influence of type H from $g(u) - 1$ edges and of type L from one arc, or payments in the amount of $l(u) + \lambda d(u)$ with $\lambda = 0, \dots g(u) - 1$ and influence of type H of $g(u) - 1 - \lambda$ arcs.

Figure 4.14: Case 2

The second case ($g(u) = 1$) is shown in Figure 4.14. Since the threshold value and the influence factor are both equal to 9, $g(u)$ equals 1. Furthermore, $l(u) = \theta(u) - (g(u) - 1)d(u) = \theta(u) = d(u)$. Now, there are $g(u) + 1 = 2$ possibilities. The first one is that $p(u) = 0$ and node u gets incoming influence of type L from one edge. The second scenario is $p(u) = l(u)$ and node u receives no incoming influence. In our example this corresponds to the two possibilities with $p(u) = 0$ and $p(u) = 9$.
Summarizing, if a node gets no partial payments, it receives incoming influence of type H from $g(u) - 1$ edges and influence of type L from one arc. If a node gets payments, then there is only incoming influence of type H, but none of type L. Moreover, there are not more than $g(u) - 1$ influencing arcs. With these findings, Gunnec et al. [2016] provide a further formulation for the LCIP. The key observation is that the variable y_{vu} from model LCIP2 is now decomposed into three binary variables x^H_{vu}, x^L_{vu} and x^Z_{vu} which correspond to the influence types high, low and zero. For example, $x^H_{vu} = 1$ means that node u gets influence of type H from node v. The coefficient $c^k(u)$ with $k \in \{H, L, Z\}$ describes the amount of incoming influence for node u for the corresponding types, e.g. $c^H(u) = d(u)$, $c^L(u) = l(u)$ and $c^Z(u) = 0$.

LCIP3 [Gunnec et al., 2016]:

$$\max \sum_{u \in V} \sum_{v \in N(u)} \sum_{k \in \{H,L,Z\}} c^k(u) \cdot x^k_{vu} \tag{4.11}$$

$$\text{s. t.} \sum_{k \in \{H,L,Z\}} (x^k_{uv} + x^k_{vu}) = 1 \qquad \forall \{u, v\} \in E \tag{4.12}$$

$$\sum_{v \in N(u)} x^H_{vu} \leq g(u) - 1 \qquad \forall u \in V \tag{4.13}$$

$$\sum_{v \in N(u)} x^L_{vu} \leq 1 \qquad \forall u \in V \tag{4.14}$$

$$x^k_{vu} \in \{0, 1\} \qquad \forall u \in V,\, v \in N(u),\, k \in \{H, L, Z\} \tag{4.15}$$

Remember that we want to minimize the sum of the partial payments. The payment for a particular node can be written as the difference between its threshold and its total incoming influence. This means one can write $\sum_{u \in V} p(u) = \sum_{u \in V} (\theta(u) - \sum_{v \in N(u)} \sum_{k \in \{H,L,Z\}} c^k(u) \cdot x^k_{vu})$, because the payment for a node can be replaced by the value of its threshold minus the sum of incoming influence from all active neighbors. Since $\sum_{u \in V} \theta(u)$ is constant, we can, instead of minimizing $\sum_{u \in V} p(u)$, maximize $\sum_{u \in V} \sum_{v \in N(u)} \sum_{k \in \{H,L,Z\}} c^k(u) \cdot x^k_{vu}$, which means that we maximize the overall incoming influence in the network. Exactly this objective function is described in LCIP3 in (4.11). Constraint set (4.12) makes sure that if there is a connection between two nodes either node u influences node v with exactly one type of influence or vice versa. (4.13) says that the number of influencing neighbors of type H is at most $g(u) - 1$, and (4.14) bounds the number of influencing neighbors of type L by 1.

Gunnec et al. [2016] show that the constraint matrix of LCIP3 is totally unimodular.

4.3 LCIP on more general settings

4.3.1 Approximability

Gunnec et al. [2016] show that in general the LCIP is NP-hard. When the LCIP requires a 100%-adoption rate (e.g. $\alpha = 1$) then the problem gets APX-hard and cannot be approximated within a factor of $O(2^{log^{1-\epsilon}|V|})$ for any fixed constant $\epsilon > 0$, unless $NP \subseteq DPTIME(|V|^{polylog(|V|)})$. [Gunnec et al., 2016]

4.3.2 Extension of the TUM-approach

In Section 4.2.4 the TUM formulation of the LCIP for trees has been introduced. Gunnec et al. [2016] extend this approach to purpose a LCIP formulation for general graphs. To this end, it is important to note that the influence diffusion process must be described by a directed acyclic graph. Thus, further constraints are added to the new model LCIP4 which is also applicable to general graphs.

LCIP4 [Gunnec et al., 2016]:

$$\max \sum_{u \in V} \sum_{v \in N(u)} \sum_{k \in \{H,L,Z\}} c^k(u) \cdot x^k_{vu} \tag{4.16}$$

$$\text{s.t. } (4.13), (4.14), \ (4.15) \tag{4.17}$$

$$\sum_{k \in \{H,L,Z\}} x^k_{vu} = y_{vu} \qquad \forall u \in V, \ v \in N(u) \tag{4.18}$$

$$\sum_{\{u,v\} \in C} y_{uv} \leq |C| - 1 \qquad \forall dicycle \, C \, in \, G \tag{4.19}$$

$$y_{uv} + y_{vu} = 1 \qquad \forall \{u, v\} \in E \tag{4.20}$$

$$y_{uv} \in \{0, 1\} \qquad \forall u \in V, \ v \in N(u) \tag{4.21}$$

The binary variable y_{uv} is the same as in LCIP2 and is defined to be 1 if u sends "influence" of type H, L or Z to node v and 0 otherwise. G denotes the directed graph formed by y and is a directed acyclic graph (DAG). The objective function is the same as in LCIP3 (4.11) and constraint sets (4.13), (4.14) and (4.15) are also inherited. Constraint set (4.18) links the two variables x and y. Constraint set (4.19) makes sure that the influence diffusion network is formed by a directed acyclic graph, and is called dicycle inequalities. In constraint set (4.20) it is ensured that if u and v have a connection either u sends "influence" of type H, L or Z to v or vice versa. Gunnec et al. [2016] use a branch-and-cut approach to solve the LCIP on general graphs and apply the shortest path procedure from Grötschel et al. [1985] for the separation of inequalities (4.19).

Gunnec et al. [2016] conduct some computational experiments to evaluate the performance of their branch-and-cut approach. It turns out that the solutions which are found with their approach and have an acceptable running time, are nearly optimal for large networks.

4.3.3 LCIP on general graphs

Cordasco et al. [2015] define the LCIP in a different way and refer to it as targeting with partial incentives. They introduce a targeting vector $\mathbf{s} = (s(u_1), s(u_2), ..., s(u_n))$ with $s(u) \in \mathbb{N}_0$, which indicates how much incentives are given for each node. For a given network $G = (V, E)$ and given thresholds $\theta(u) : V \to \mathbb{N}$ the task is to find a target vector $\mathbf{s}$ such that the costs $C(s) = \sum_{u \in V} s(u)$ are minimized. Different than Gunnec et al. [2016], they do not use influence factors, but assume that every vertex receives influence from an active neighbor in the amount of one. Additionally, Cordasco et al. [2015] assume that the threshold of node u is between one and the degree of node u.

To solve the LCIP with a 100% adoption rate, Cordasco et al. [2015] propose an algorithm which is called $TPI(G)$ for an input graph G. The algorithm starts with the initial graph G and at every discrete time step a node is removed from the graph such that a certain parameter is maximized. These vertices which are removed by this method do not receive partial payments. In the course of the algorithm it could be the case that a node has a higher threshold than its number of neighbors. Without partial payments it could never get active. Since a 100% adoption rate is compulsory, we need this node to adopt as well. Hence, we give it incentives so that the threshold is reduced to a value which is at most as high as the number of remaining neighbors.

The algorithm $TPI(G)$ returns for *any* graph G a target vector $\mathbf{s}$ for G. Furthermore, Cordasco et al. [2015] prove that the costs of this target vector $C(\mathbf{s}) = \sum_{u \in V} s(u)$ are constrained by $\sum_{u \in V} \frac{\theta(u)(\theta(u)+1)}{2(deg(u)+1)}$ where $deg(u)$ is the degree of node u in the initial graph.
For a complete graph or a tree, the algorithm $TPI(G)$ outputs actually an optimal target vector $\mathbf{s}$. Additionally, for a tree, the costs of the optimal target vector can be explicitly stated, e.g., $C(\mathbf{s}) = n - 1 + \sum_{u \in V} (\theta(u) - d(u))$ where n is the number of nodes of the tree. [Cordasco et al., 2015]

For the $TPI(G)$ algorithm Cordasco et al. [2015] conduct the same experiments as for the $WTSS(G)$ algorithm, but compare it with two other algorithms. One is called DegreeFrac and chooses nodes fractionally proportional to its degree. In detail, the algorithm pays on node u an payment of $s(u) = \lfloor \frac{deg(u) \cdot B}{2 \cdot |E|} \rfloor$, where B describes the budget. The second algorithm is called DiscountFrac and selects the node with maximal degree and endows it with the minimum amount which can achieve to activate this node. After paying out this node, the degree of the neighbor nodes are reduced by 1. Cordasco et al. [2015] show that the $TPI(G)$ algorithm outperforms these two algorithms.

5. The Least Cost Influence problem in multiple social networks

5.1 Introduction

Up to now, we have only considered influence maximization problems on isolated single social networks. However, many people are not only member of one social network, but join several of them. This has the effect that people could be influenced through more sources and can spread information on more platforms at the same time. Additionally, an individual can carry information from one network to another. For example, if someone reads on Facebook some information about a product and shares it on Facebook and additionally on Twitter, the information has entered a new social network.
Furthermore, it has to be taken into account that the difficulty of influencing people can vary in different social networks. These observations lead to a new problem which has to be defined and modeled. [Zhang et al., 2016b]

In this chapter the LCIP on multiple social networks will be formally defined. The key idea to solve the problem on multiple networks is to couple these social networks into one network and then use already existing methods to solve the LCIP on a single network. Therefore, so-called coupling schemes are essential. Lossless and lossy coupling schemes will be presented and illustrated with examples. At the end of the chapter the experimental work of Zhang et al. [2016b] will be summarized.

Shen et al. [2012] also consider the LCIP problem in multiple social networks. They take into account that targeting interest-matching users, which have similar interests, is advantageous. Moreover, the fact that there are not only positive but also negative relationships is considered. Shen et al. [2012] couple multiple networks into one network by combining nodes which occur in more networks into one supernode. With this approach the characteristics of the individual networks are lost.

5.2 Definition of the problem

Zhang et al. [2016b] are one of the first who define the LCIP on multiple social networks and model it in the following way.
Let k be the number of the social networks which are treated in this problem. Then every network $i \in \{1, ..., k\}$ is represented through a weighted directed graph G^i which is determined through its set of vertices $V^i = \{u_1^i, ..., u_n^i\}$ and the set of arcs E^i, where every arc is endued with a weight $w^i(v, u)$ which determines how much a node v influences node u in the i^{th} network. The threshold value for node u in the network G^i is denoted with $\theta^i(u)$. $N_u^{i-} = \{v \in V^i : (v, u) \in E\}$ and $N_u^{i+} = \{v \in V^i : (u, v) \in E\}$ represent the sets of incoming resp. outgoing neighbors of node u in the i^{th} network. $G^{1...k}$ denotes the system of all k networks and the union $U = \bigcup_{i=1}^{k} V^i$ describes the set of all nodes which are part in any of those k networks. It could also be the case that a node joins in more than one network. Zhang et al. [2016b] call these nodes overlapping users.

Remember that in the Linear Threshold model a node gets active if the total influence from its active neighbors is as least as high as its threshold. Hence, condition (5.1) is necessary to get node u active, if it is not chosen to be in the target set.

$$\sum_{v \in N(u),\, v\, is\, active} w(v, u) \geq \theta(u) \tag{5.1}$$

In the case of multiple networks, a node gets active if its threshold is reached in at least one of the networks. This means that the influence for the nodes are considered in every network separately. If a node gets enough influence to get active in one network, then this has the effect that it gets active in all networks and thus transport the information from one network to another. Condition (5.2) has to be fulfilled for at least one network i to get node u active. [Zhang et al., 2016b]

$$\sum_{v \in N_u^{i-},\, v\, is\, active} w^i(v, u) \geq \theta^i(u) \tag{5.2}$$

The general principle of the activation process is the same as in the case of a single network. We start with an initial active set, at every time step the state of the nodes is updated and the process stops if no more activations are possible. Zhang et al. [2016b] constrain the number of time steps of the diffusion to d and call this the number of propagation hops. $A^d(G^{1...k}, S)$ denotes the set of active nodes after d time steps, where S is the initial active set.

With this prerequisites, we can formally define the LCIP on multiple social networks.
Definition (Zhang et al. [2016b]): Let $G^{1...k}$ be a system of k networks where U describes the set of all users. Let d be a positive integer and $0 < \alpha \leq 1$. Then the LCIP asks for the smallest set $S \subset U$ which guarantees that at least an α fraction of users is active after d time steps, i.e.,

$$|A^d(G^{1...k}, S)| \geq \alpha|U| \tag{5.3}$$

5.3 Coupling schemes

Since there are already several investigations and approaches for the influence maximization problem in a single network, it is advantageous to find a method to convert multiple social networks into one social network. Zhang et al. [2016b] use the technique of coupling schemes. Once the multiple networks are transformed into a single social network, the existing theory for single social networks is applicable. Coupling schemes are supposed to maintain relevant information about the networks and replicate the propagation process from all particular networks.

A coupling scheme which projects multiple social networks to a single coupled network $G = (V, E)$ has to fulfill the following three important characteristics. [Zhang et al., 2016b]
1. There is a set of nodes $\mathcal{U} \subseteq V$ and a bijection $\mathcal{F}$ which maps people to vertices in the coupled network, e.g. $\mathcal{F} : U \to \mathcal{U}$. With this requirement one can identify the users in the coupled network. [1]
2. There is a time mapping function $\mathcal{T} : \mathbb{N} \to \mathbb{N}$. This condition helps to identify when a node is activated.
3. User $u \in U$ gets active at time t on $G^{1...k}$ if and only if $\mathcal{F}(u)$ gets active at time $\mathcal{T}(t)$ in G. This condition ensures that the diffusion process is the same for the set of users U than for the set of nodes $\mathcal{U}$, which means that the spread of information is retained.

These are very strong conditions and may be hard to fulfill. Therefore, the last constraint can be relaxed to that a user $u \in U$ gets active at time t on $G^{1...k}$ if $\mathcal{F}(u)$ gets active at time $\mathcal{T}(t)$ on G. This means that the activation of $u \in U$ is necessary, but not sufficient to activate $\mathcal{F}(u)$.
In the case when the third condition is relaxed then we talk about a lossy coupling scheme, because some information about the diffusion process gets lost. However, if the last

[1] Typo in the original paper

condition is satisfied the scheme is called lossless coupling scheme. [Zhang et al., 2016b] In the next sections these two coupling schemes are discussed in detail.

5.4 Lossless coupling schemes

In this section, the clique and the star coupling scheme introduced by Zhang et al. [2016b] are discussed. After that, it is shown how these schemes can be used as so-called reduced coupling schemes.
The coupling schemes are based on the Linear Threshold model (see Section 2.1.1), but they are also applicable to other diffusion models such as the Cascade model.

5.4.1 Clique coupling scheme

We consider networks $G^1, ..., G^k$ which we want to couple into one network G. First, Zhang et al. [2016b] implement dummy nodes for several people for networks which they do not join. Therefore, every user is now member of all considered networks. In the following section these dummy nodes will be omitted, because they are redundant. Furthermore, every user u has a representative vertex in network G^i which is denoted with u^i. If there is an arc between u and v in G^i then u^i and v^i are connected. Now, combining all users of all networks leads to one big network G. Edges between same users but in different networks, e.g. (u^i, u^j), are provided with a weight $w(u^i, u^j) = \theta(u^j)$. Of course, nodes which represent the same user in different networks influence each other, because if a node u^i gets active in network i then all nodes which represent the same user in other networks get active as well. However, the problem here is that the activation of these other nodes is time lagged since these other nodes are activated one step after the first representative u^i is activated. So a node u^j can influence other potential buyers one step later than u^i can spread information although they are representing the same person. To prevent this problem, Zhang et al. [2016b] implement the so called gateway vertices for all users. The gateway vertex for user u is denoted with u^0 and has the property that all representatives of u can only influence other people through this gateway vertex u^0. Therefore, all edges (u^i, v^i) are replaced by edges (u^0, v^i). Furthermore, edges between all representatives of user u and its gateway vertex, $u^0, u^1, ...u^k$, are introduced. Since there are connections between all nodes, we say they form a clique. And this clique can only influence through the gateway vertex and now has the advantage that the spread of information reaches all neighbors of any representative of user u at the same time. So the time delay has vanished. [Zhang et al., 2016b]
The extra edges between all representatives are introduced, because then the influence can travel directly from one representative to another. Otherwise the representatives would

get active one time step later through the gateway vertex. However, at the same time step their neighbors can already receive the influence also from the gateway vertex. So the extra edges are added for technical reasons.
Lastly, the thresholds and edge weights are modeled in the following way: Gateway vertices are endowed with a threshold value of 1. For a node u^i the threshold is equal to the threshold of u in network G^i, e.g., $\theta(u^i) = \theta^i(u)$.
Edges from representatives to the gateway vertex are endowed with 1. If there is a connection between user u^i and v^i in network G^i, then the edge (u^0, v^i) possesses weight $w(u^0, v^i) = w^i(u, v)$. Within a clique the edge weights are chosen in a way that they all get active if one of them is activated, e.g., $w(u^i, u^j) = \theta(u^j)$. It is easy to obtain, that by influencing in the amount of the threshold, one active neighbor within a clique is sufficient to activate the whole clique. [Zhang et al., 2016b]

To better understand the concept of the clique coupling scheme consider Example 5.1 where a very small social network is shown.

Example 5.1. *In Figure 5.1 three social networks are shown. Representatives of the same users are dyed in the same color. The nodes and their thresholds in the different networks are given, and the influence factors are specified as well.*

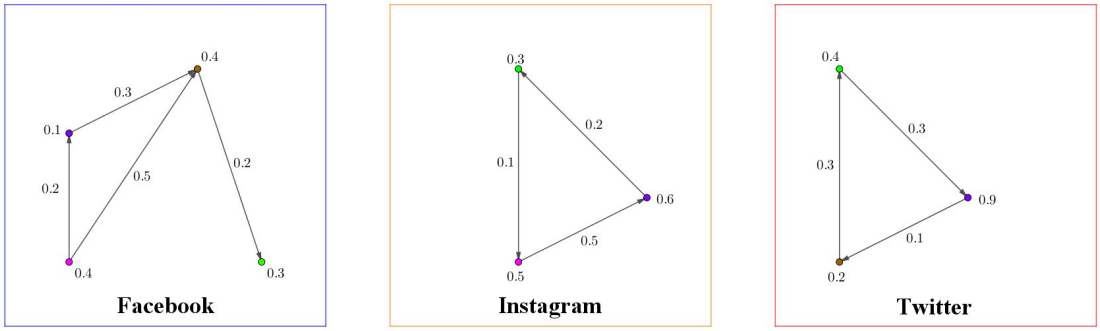

Figure 5.1: Facebook, Instagram and Twitter

In Figure 5.2 the gateway vertices are already added and the new connections between gateway and representative nodes as well as the relationships within the cliques are indicated. If the green node receives influence in the amount of 0.27 *from the brown node in Facebook, then this has the effect that the brown gateway vertex influences the green node in Facebook with a weight of* 0.27*. The influence and edge weights among the representatives of the green user and the green gateway vertex are illustrated in Figure 5.3.*

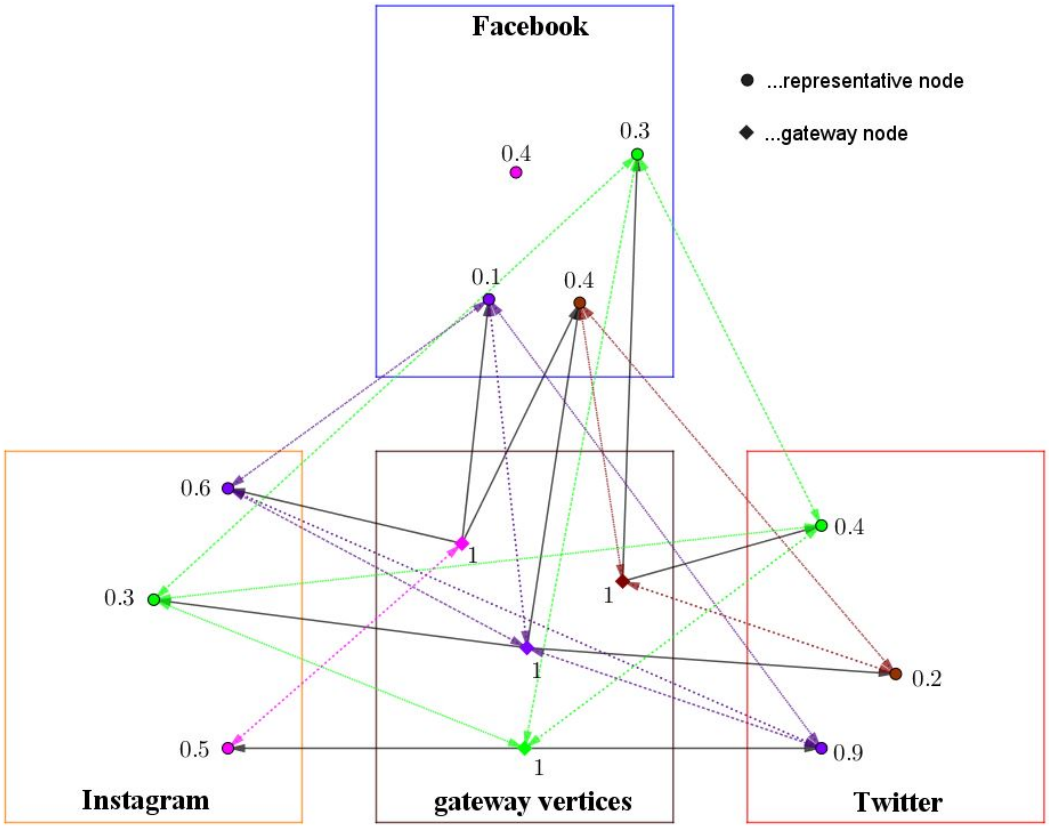

Figure 5.2: Representation with gateway vertices

Zhang et al. [2016b] state that the solution of the LCIP in multiple social networks equals to the solution in the coupled network. The only difference is that if the propagation process in the original multiple networks makes d hops, in the coupled network we need $2d$ hops, since there is an extra hop through the gateway vertices.
Further interesting observations concern the number of vertices and edges in the coupled network. Since every user u has $k+1$ associated nodes, e.g. $u^0, ..., u^k$ (dummy variables included), we have all together $(k+1)|U| = (k+1)n$ vertices, where n is the total number of nodes in the coupled network. The number of arcs equals $|E| = \sum_{i=1}^{k} |E^i| + nk(k+1)$ which is composed of the number of edges which are already in the initial networks plus the edges which are added within a clique. [Zhang et al., 2016b]

5.4.2 Star coupling scheme

Since the number of additional edges, resulting from the clique coupling scheme, is very high and increases hugely with the number of users, Zhang et al. [2016b] alternatively present the star coupling scheme. In the star coupling scheme there is another extra node for each user added namely the intermediate vertex which has its place between the gate-

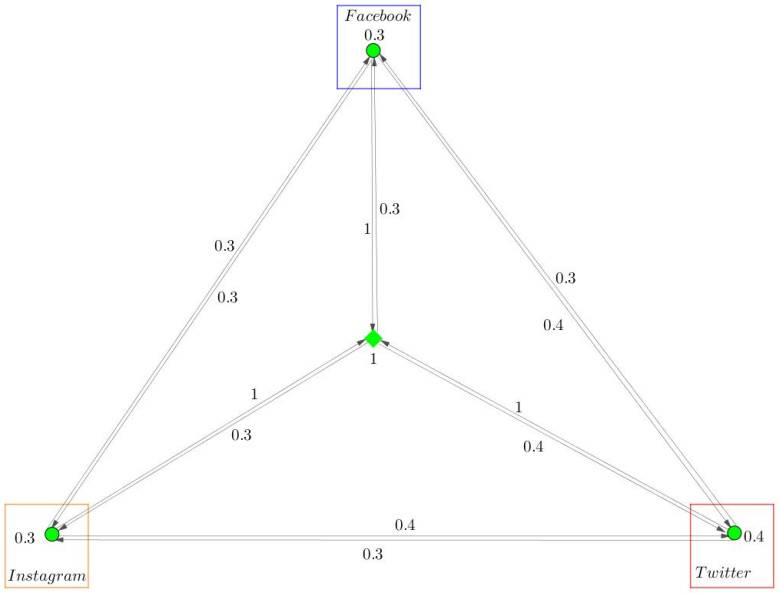

Figure 5.3: Influence among one clique

way vertex and all other vertices of what is previously called clique. Now, the term clique does not apply anymore. The advantage is that now not all representatives of one node are connected with each other but they are linked through the intermediate vertex. Furthermore, the intermediate vertex has a connection to the gateway vertex. The other nodes of the previously clique have no connection to the gateway vertex anymore.
The reason why such an intermediate node is introduced is to reduce the number of edges. We still need the gateway vertex to make sure that the influence to other users can spread only when all representatives of the activated node are active.
The intermediate vertex of user u is denoted with u^{k+1} and has a threshold $\theta(u^{k+1}) = 1$. The weights of the extra edges are modeled in the following way. The edges between intermediate and gateway vertex have weight 1, e.g., $w(u^{k+1}, u^0) = w(u^0, u^{k+1}) = 1$. The edges from the intermediate vertex to the other nodes have a weight in the amount of the threshold of the particular node, e.g., $w(u^{k+1}, u^i) = \theta(u^i)$. Whereas edges from representative nodes to the intermediate vertex have weight 1, e.g., $w(u^i, u^{k+1}) = 1$. [Zhang et al., 2016b] The idea of the star coupling scheme is illustrated in Example 5.2.

Example 5.2. *In this example we consider the same three social networks, thresholds and influence factors as in Example 5.1.*
Figure 5.1 and Figure 5.2 are also adopted for this example, but the influence among representatives of the same user is now modeled in a different way which is shown in Figure 5.4.

With the star coupling scheme the number of vertices is now $|V| = (k+2)|U| = (k+2)n$, because for every user we have 2 additional nodes, namely the gateway and the

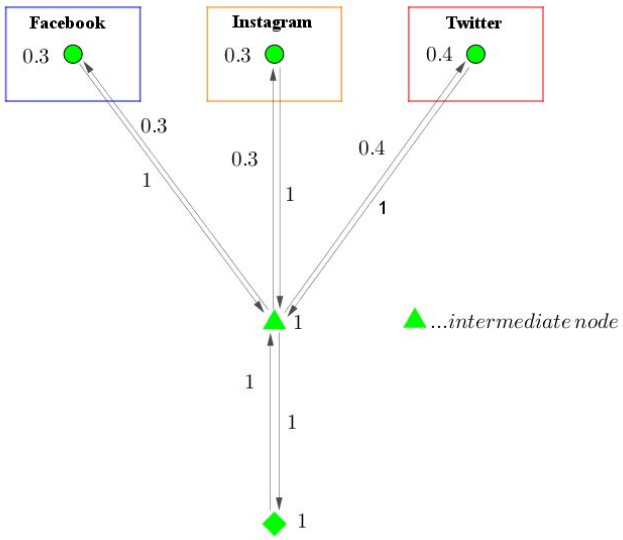

Figure 5.4: Influence in form of a star

intermediate vertex. However, the number of arcs is reduced to $|E| = \sum_{i=1}^{k} |E^i| + 2n(k+1)$ since there are no connections between all representatives but only via the intermediate vertex.

Moreover, the propagation process with d hops in the original multiple social network corresponds to the propagation process with $3d$ hops in the coupled network. [Zhang et al., 2016b]

5.4.3 Reduced coupling schemes

Since there are added dummy nodes for all networks that a certain user does not join, we have a large number of nodes. These extra vertices are added to ensure that the number of influenced nodes in the coupled network is scaled up from the number of influenced vertices in the original networks. Consider three social networks with $0.5n$, $0.7n$ and $0.4n$ users which leads in total to $1.6n$ nodes. In the clique coupling scheme we would have $(k+1)n = 4n$ vertices and in the star coupling scheme $(k+2)n = 5n$ vertices. [Zhang et al., 2016b]

They introduce weights for the nodes in the coupled network and make sure that "the total weight of active vertices is scaled from the number of active users in the original network system" [Zhang et al., 2016b]. A representative vertex is only generated for the p networks which a user joins. Furthermore, all representative vertices are associated with a weight 1 and each user vertex has weight $k - p$. Thus, the number of extra nodes is reduced to n in the clique coupling scheme and to $2n$ in the star coupling scheme. [Zhang et al., 2016b]

5.5 Lossy coupling shemes

In the previous section, coupling schemes which preserve all information about the networks, but produce a very large coupled network, are introduced. In this section, a method where some information may get lost but the size of the coupled network is kept relatively small, is discussed. Of course, we want to keep the loss of information as small as possible and find a solution for the LCIP in the coupled network which is very similar to the solution of the problem in multiple networks.
In this section, a coupling scheme based on the Linear Threshold model is considered. However, the theory of a lossy coupling scheme can be adapted for other diffusion models as well.
Zhang et al. [2016b] construct a lossy coupling scheme in the following way: First, they modify the condition to get a user u active from condition (5.4)

$$\sum_{v \in N_u^{i-},\, v\, is\, active} w^i(v,u) \geq \theta^i(u) \tag{5.4}$$

to condition (5.5) where $\alpha^1(u), ..., \alpha^k(u)$ are positive parameters.

$$\sum_{i=1}^{k} (\alpha^i(u) \sum_{v \in N_u^{i-},\, v\, is\, active} w^i(v,u)) \geq \sum_{i=1}^{k} \alpha^i(u)\theta^i(u) \tag{5.5}$$

Since we relax the condition, it could of course be the case that condition (5.4) is fulfilled but condition (5.5) not.
Next, Zhang et al. [2016b] show a method how to choose the parameters $\alpha^1(u), ..., \alpha^k(u)$. Since users are easier to influence in some networks than in other networks, Zhang et al. [2016b] define the easiness of a user u in network i as the ratio between the total influence from friends and its threshold in network i, e.g., $\epsilon^i(u) = \frac{\sum_{v \in N_u^{i-}} w^i(v,u)}{\theta^i(u)}$.
Now, this easiness values are used as parameters. Zhang et al. [2016b] construct a coupled network G with $V = \{u_1, ..., u_n\}$ and the thresholds of the vertices are given as $\theta(u) = \sum_{i=1}^{k} \alpha^i(u)\theta^i(u)$. The weight of an edge is given as $w(v,u) = \sum_{i=1}^{k} \alpha^i(u) w^i(v,u)$, with $w^i(v,u) = 0$ if there is no connection between node v and u in network i.
If a user u gets active in network G during the propagation process which starts from the same target set S in G and $G^{1...k}$ than this means that u also gets active in the network $G^{1...k}$. This implies that if an α fraction of users get active through a target set S in G than at least an α fraction gets active in $G^{1...k}$ with the same target set S. In the other direction this is not always true, because the activation condition is modified. Therefore,

the coupling scheme is called lossy, because we can not adopt all characteristics of the networks. In Example 5.3 such a matter is described.
[Zhang et al., 2016b]

Example 5.3. *We again consider the three networks from Figure 5.1 and now we want to couple them into one network using the easiness parameters.*
First, the easiness parameters for each user and network is calculated and the threshold value as well. The user colored in pink is denoted with u_p, the lilac user with u_l, the brown one with u_b and the green user with u_g.

Pink user

Facebook: $\alpha^i(u_p) = \epsilon^i(u_p) = \frac{\sum_{v \in N^{i-}_{u_p}} w^i(v,u_p)}{\theta^i(u_p)} = \frac{0}{0.4} = 0$
Instagram: $\alpha^i(u_p) = \epsilon^i(u_p) = \frac{0.1}{0.5} = 0.2$
Threshold: $\theta(u_p) = \sum_{i=1}^{k} \alpha^i(u_p)\theta^i(u_p) = 0 + 0.2 \cdot 0.5 = 0.1$

Lilac user

Facebook: $\alpha^i(u_l) = \epsilon^i(u_l) = \frac{0.2}{0.1} = 2$
Instagram: $\alpha^i(u_l) = \epsilon^i(u_l) = \frac{0.5}{0.6}$
Twitter: $\alpha^i(u_l) = \epsilon^i(u_l) = \frac{0.3}{0.9}$
Threshold: $\theta(u_l) = 2 \cdot 0.1 + \frac{0.5}{0.6} \cdot 0.6 + \frac{0.3}{0.9} \cdot 0.9 = 1$

Brown user

Facebook: $\alpha^i(u_b) = \epsilon^i(u_l) = \frac{0.3+0.5}{0.4} = \frac{0.8}{0.4} = 2$
Twitter: $\alpha^i(u_b) = \epsilon^i(u_b) = \frac{0.1}{0.2} = 0.5$
Threshold: $\theta(u_b) = 2 \cdot 0.4 + \frac{0.1}{0.2} \cdot 0.2 = 0.9$

Green user

Facebook: $\alpha^i(u_g) = \epsilon^i(u_g) = \frac{0.2}{0.3}$
Instagram: $\alpha^i(u_g) = \epsilon^i(u_g) = \frac{0.2}{0.3}$
Twitter: $\alpha^i(u_g) = \epsilon^i(u_g) = \frac{0.3}{0.4}$
Threshold: $\theta(u_g) = \frac{0.2}{0.3} \cdot 0.3 + \frac{0.2}{0.3} \cdot 0.3 + \frac{0.3}{0.4} \cdot 0.4 = 0.7$

Next, the edge weights are determined.

$w(u_g, u_p) = \sum_{i=1}^{k} \alpha^i(u_p) w^i(u_g, u_p) = 0.2 \cdot 0.1 = 0.02$
$w(u_p, u_l) = 2 \cdot 0.2 + \frac{0.5}{0.6} \cdot 0.5 = 0.817$
$w(u_l, u_b) = 2 \cdot 0.3 + \frac{0.1}{0.2} \cdot 0.1 = 0.65$
$w(u_g, u_l) = \frac{0.5}{0.6} \cdot 0.2 + \frac{0.3}{0.9} \cdot 0.3 = 0.267$
$w(u_b, u_g) = \frac{0.2}{0.3} \cdot 0.2 + \frac{0.3}{0.4} \cdot 0.3 = 0.358$
$w(u_p, u_b) = 2 \cdot 0.5 = 1$

The resulting coupled network is graphically illustrated in Figure 5.5.

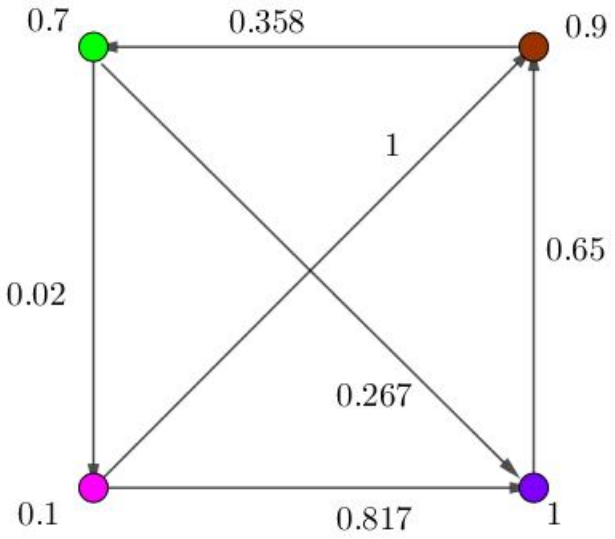

Figure 5.5: The coupled network via easiness parameters

To verify that this coupling scheme produces actually a lossy solution consider the case when $d = 1$ *and* $\alpha = 0.75$ *which means that in one step* 3 *of* 4 *users have to be influenced. In Figure 5.5 we see that this is not possible if* $|S| = 1$ *using the lossy coupling scheme. Giving the pink user incentives at the beginning leads to an activation of the brown user but no more users can also be influenced. Moreover, the green and lilac user cannot influence any other members. Whereas in the original system of networks giving the pink user incentives leads to an activation of the lilac and brown user. (Because in Facebook the threshold of the lilac user is lower than the influence from the pink node (0.1<0.2) and for the brown user as well (0.4<0.5).) Thus, we get only a feasible solution when not using the lossy coupling scheme.*

5.6 Evaluation of the coupling schemes

In this section, the results of the experimental work of Zhang et al. [2016b] are summarized.

First, Zhang et al. [2016b] propose an improved greedy algorithm and compare the solution of the algorithm to the optimal solution which is computed through a 0-1 integer linear programming problem. They generate small size networks since the IP is not applicable on large networks. The seed size describes the number of initial chosen nodes. The analysis of the data shows that the seed size, found with the heuristic, of the various cou-

pling schemes are close to the optimal seed size. Furthermore, one can observe that the seed size gets nearer to the optimal size when the number of propagation hops is increased.

Next, Zhang et al. [2016b] compare the lossy and the lossless coupling schemes. They use two different kinds of data sets. On the one side real networks data and on the other hand synthesized networks which were created through a random network model. The number of hops is set to $d = 4$ and the influenced fraction to $\alpha = 0,8$.

The evaluation of the solution quality shows that the seed size of the solution based on lossy coupling schemes is larger than of the lossless. However, a loss of information occurs only at overlapping users which are relatively sparse. Therefore, the effect on the solution quality is also not very high. Analyzing the running time yields to the conclusion that the algorithm is slower when using lossless than lossy coupling schemes. The reason for that is the large number of vertices (many of them are redundant) and edges in the lossless scheme. The impact on the running time is relatively high when the networks are large. All in all, one can conclude that lossless schemes should be applied when the focus lies on the quality of the solution and in rather small networks. Whereas, lossy coupling schemes are better when the running time is limited and the overlapping fraction is small.

Another issue which is considered by Zhang et al. [2016b] is the structure of the seed set and the influenced set. Not surprisingly, the overlapping users are chosen very frequently in the seed set although they are only a small fraction of the whole users. This is because the overlapping users can influence in more than one social network and therefore, probably can reach a larger spread of people. One can also observe that the amount of the overlapping users in the solution increases only slightly when the influenced fraction α is increased, because all good overlapping users are already taken.
Furthermore, Zhang et al. [2016b] observe that the nodes of large networks are chosen more often than nodes from small networks in the seed set and therefore, can also be found more frequently in the influenced set.
Moreover, they find out that it could be the case that influencing is easier in a particular network than in other networks. This occurs when users of one network are rarely chosen in the seed set, but appear often in the influenced set.

Lastly, the impact of additional networks is studied. Therefore, Zhang et al. [2016b] generate synthesized networks. One can observe that with increasing the number of networks the seed size decreases rapidly. Thus, the additional network speeds up the spread of the information a lot. This observation can be made with all coupling schemes.

6. Least Cost Rumor Blocking

6.1 Introduction

Since social networks serve more and more as an information platform, negative informations can spread through social networks as well. Not only the truth can spread rapidly through social networks but also rumors can diffuse very fast and can lead to serious consequences. For example, in 2009 the misinformation that the swine flu broke out appeared on Twitter [Nguyen et al., 2012]. Through the diffusion of the correct statement that the swine flu did not break out, the rumor could have been stopped.

Hence, the limitation of the diffusion of rumors is an interesting and important research topic. In the last years, a couple of authors (Fan et al. [2013], Budak et al. [2011], Pham et al. [2016], He et al. [2012], Zhang et al. [2016a]) have considered this problem and they use various approaches.

Fan et al. [2013] consider the situation when two types of information spread over the network at the same time. One type is some rumor and the second type describes the truth and can be interpreted as the protector group which wants to limit the diffusion of the rumor.

A similar approach is presented by Budak et al. [2011]. They model the fact that some rumor occurs in the network and is detected after some time and at this point a limiting campaign is started.

A different approach is considered by Pham et al. [2016]. They consider unwanted users to whom the information should not come up. Since unwanted users have an opposite opinion and can use the information for their benefits, the purpose is to prolong the activation of such users as long as possible. A company, for example, wants to hide its marketing strategy from competitive firms.

He et al. [2012] consider the scenario when a company want to block the influence from a competitor by selecting an initial target set which spreads its own information as effectively as possible. They refer to the problem as influence blocking maximization problem and use a competitive Linear Threshold model. Competitive Threshold models are also considered by Borodin et al. [2010].

He et al. [2012] prove that the objective function of the influence blocking maximization problem under the competitive Linear Threshold model is submodular and introduce a greedy algorithm. Moreover, they develop a second algorithm with a better running time which has an analogous blocking effect.
Zhang et al. [2016a] consider the detection of misinformation in social networks. Different than the other authors they utilize time constraints. They refer to the problem as Time Constrained Misinformation Detecting and find out that the problem is NP-hard. Furthermore, they show a network compression based algorithm to solve the problem.

This chapter is structured in the following way: First, diffusion models are presented. Then, the LCRB problem is defined formally. Complexity results and possible solution approaches are summarized. Lastly, results of the experimental work of Fan et al. [2013], Budak et al. [2011] and Pham et al. [2016] are summarized.

6.2 Models

Fan et al. [2013] introduce two Cascade models which describe the spread of two simultaneously starting cascades. The first cascade characterizes the rumors and is denoted by R and the second cascade P stands for the protectors. We say a node is infected when it is influenced by rumors, protected when it is influenced by protectors or inactive otherwise. The rumor originators are denoted by A_R and the initial protected nodes are described by A_P. Furthermore, the spread of information is progressive as in the Cascade models in Section 2.2 which means that once a node is infected/protected it can not change its status anymore. If a node is influenced from both cascades at the same time, then cascade P has precedence since it is assumed that people mostly believe the truth.
The models described by Fan et al. [2013] are called Opportunistic One-Activate-One (OPOAO) Model and Deterministic One-Activate-Many (DOAM) Model. They will be explained in detail in the next section.

Budak et al. [2011] describe also two Cascade models which are very similar to those of Fan et al. [2013]. The models are also progressive and the good information is preferred over the bad one. There are three possible states: inactive, influenced by cascade C (campaign) or influenced by cascade L (limiting campaign). Budak et al. [2011] call their models Multi-Campaign Independent Cascade model (MCICM) and Campaign-Oblivious Independent Cascade model (COICM). The crucial difference to the models from Fan et al. [2013] is that the cascades do not start simultaneously, but cascade C starts first and after some time the other cascade L begins to spread.

6.2.1 OPOAO Model

The diffusion process starts with an initial set of protectors A_P and an initial set of rumors A_R. At every time step the infected and protected nodes have a single chance to successfully influence an inactive neighbor. Each neighbor of a particular node u is chosen with the same probability, e.g. $\frac{1}{deg_{out}(u)}$ where $deg_{out}(u)$ describes the out-degree of node u. In the next time step again all protected and infected vertices can choose an inactive node to activate. The process runs in discrete time and ends if no more activations are possible or no more inactive nodes are available. This model captures the situation when everybody can only speak to one person at the same time. Hence, the diffusion of information is relatively slow.
[Fan et al., 2013]

6.2.2 DOAM Model

In this model we again start with initial seed sets of protectors and rumors. The important difference to the previous model is that now *every* neighbor of an infected/protected node is influenced and adopts its status in the next time step. Hence, a person can reach more people at the same time and the diffusion of the information is much faster. Again, the process stops if no more activations are possible. [Fan et al., 2013]

6.2.3 MCICM

The initial active sets of cascade C (campaign) and L (limiting campaign) are denoted with A_C resp. A_L. Cascade C is spotted with a time delay r and at that time cascade L begins to spread. At every time step a newly activated node u of cascade C and L has once the chance to influence an inactive neighbor v with a success probability $p_{C,u,v}$ resp. $p_{L,u,v}$. If both cascades reach a certain node at the same time step then cascade L has priority since "good" information is preferred. Again the process stops if no more activations are possible. [Budak et al., 2011]

6.2.4 COICM

This model is equal to the MCICM with the only difference that the success probability is independent of the campaign, e.g., $p_{u,v} = p_{C,u,v} = p_{L,u,v}$. One can interpret this as the circumstance where the quality of the information is the same for both campaigns, but the first one which arrives at the node is more likely to convince it. So the activation does not depend on which campaign tries to influence, unless both cascades try to activate at the same time then cascade L is favored. This model is applicable when two companies offer

similar products or ideas and none of them can be interpreted as a good or bad company. [Budak et al., 2011]

6.3 Problem definition

Before defining the LCRB problem in a formal way, an important observation of the network structure has to be mentioned. The nodes in the network often divide up into groups with a high density of connections within the group and sparse external connections. Since the people with many common interests form a group, it is reasonable to assume that they mostly communicate within the group about these topics. Fan et al. [2013] call such a group "community" and denote the set of disjoint communities with $C = \{C_1, C_2, ..., C_k\}$. The community where the rumor has its origin is called rumor community. The vertices which are connected with the rumor community are called R-neighbors and there is a special focus to protect these nodes, because protecting them can prevent that the rumor is transmitted to all other members of the network. Since there are only a few connections crossing communities, it is advantageous to protect these so-called bridge ends.

Fan et al. [2013] define the LCRB Problem as the task to find, for a given network with given communities, rumor originators and bridge ends, the minimum set of protector originators so that at the end there is at least an $\alpha \in [0, 1]$ fraction of bridge ends protected. They describe two variants of the problem. The first one is called Least Cost Rumor Blocking under opportunistic model (LCRB-P) and only an α fraction is required to be protected and the diffusion process is described by the OPOAO model. The second version is called Least Cost Rumor Blocking under deterministic model (LCRB-D) and covers the diffusion under the DOAM model. In this case $\alpha = 1$ is required because the information spreads out rapidly. [Fan et al., 2013]

However, Budak et al. [2011] define a slightly different problem that does not build upon community structure. The task of the problem is to minimize the number of nodes influenced by campaign C. Formally, Budak et al. [2011] refer to the problem as the eventual influence limitation (EIL) problem and the MCICM is used as propagation model. The diffusion of information for campaign C starts from the so-called adversary node n_a and is spotted with a time lag r. At the point where the spread of the information is detected, the other cascade L starts. For a given budget k the EIL problem asks for the initial active set of cascade L, A_L, which minimizes the expected number of vertices which are influenced by campaign C at the end of both processes. Budak et al. [2011] show the validity of their

results and approximations when the campaign C starts not only from one adversary node but from an initial set.
The diffusion process is defined by Budak et al. [2011] to have the high-effectiveness property as the case of $|A_C| = 1$, which means that the spread of information has one single originator, and the limiting campaign L has a strong influence power where everyone who is connected with a L-influenced node is also convinced, e.g., $p_{L,u,v} = 1$ if there is a connection between u and v, and $p_{L,u,v} = 0$ otherwise.

Pham et al. [2016] investigate a problem called Maximizing Influence while unwanted target users limited (IML) which has the Linear Threshold model from Section 2.1.1 as diffusion model. The purpose is to find an initial active set which maximizes the influence in the network such that the influence of unwanted users is under a certain threshold after a given number of propagation hops. Formally, the d-IML problem is specified as follows:

Definition (Pham et al. [2016]): Given a social network represented by a directed graph $G = (V, E, w)$ and a Linear Threshold model. Let $T = \{t_1, t_2, ..., t_p\}$ be the set of unwanted users and d the number of propagation hops. Let $\delta_d(S)$ be the total number of users that have been influenced by the set of S after d hops. The goal of the d-IML problem is to choose the set of seed users $S \subseteq V$ which has at most size k that maximizes $\delta_d(S)$ such that the total influence unwanted users receive is bounded by a parameter of leakage τ_i, i.e.: $\sum_{u_i \in N^a(t_i)} t_i < \tau_i$, where $N^a(t_i)$ describes the active neighbors of node t_i. [1]

6.4 Complexity results and approach of possible solutions

The solution approaches of the LCRB-P and LCRB-D problems divide up into two steps. The first one is to figure out the bridge ends which is done with the Breadth First Search (BFS)-method. The second stage is to determine the protectors. [Fan et al., 2013]
For the LCRB-P problem Fan et al. [2013] introduce a greedy algorithm with an approximation ratio of $1 - \frac{1}{e}$ for selecting the initial protectors. The algorithm chooses the vertex which has the highest marginal gain in protecting the bridge ends. Furthermore, they show that the expected influence function of protectors is submodular because of the diminishing return condition, similar as in Section 2.2.3.
For the LCRB-D problem Fan et al. [2013] propose an algorithm with a $O(\ln n)$-approximation ratio where n is the number of bridge end nodes.

[1] Slightly simplified the definition by Pham et al. [2016]

Budak et al. [2011] prove some interesting properties about the EIL problem. First, the EIL problem is NP-hard even when the diffusion process has the high-effectiveness property. Secondly, under the MCICM the EIL problem is not submodular in general, but with the high-effectiveness property it is. Lastly, the EIL problem under the COICM is submodular in general.
For the case when the EIL problem is submodular, Budak et al. [2011] propose an algorithm which solves the problem with an approximation ratio of $1 - \frac{1}{e}$. However, in large social networks this method is too expensive and thus Budak et al. [2011] develop three heuristics. The first one is called degree centrality. Nodes which have a high degree centrality, which means that they have many neighbors and thus can influence a lot of people, are chosen as seed nodes for the limiting campaign. The second heuristic is based on early infectees. Here, the initial nodes which are expected to be already influenced at time r are chosen. The last heuristic is called largest infectees. In this method, the nodes which are expected to influence the most people are chosen in the initial active set of cascade L.

For the d-IML problem Pham et al. [2016] prove that the influence function $\delta_d(.)$ is submodular for the Linear Threshold model. Moreover, they show that the d-IML problem is NP-hard and can not be approximated in polynomial time within a ratio of $1 - \frac{1}{e}$.
Furthermore, Pham et al. [2016] introduce a heuristic algorithm which is based on a heuristic function using not only the marginal gain of a user but also the fitness. Additionally, they formulate the d-IML problem as $0 - 1$ Integer Linear Programming problem.

6.5 Evaluations

In this section the experiments of Fan et al. [2013], Budak et al. [2011] and Pham et al. [2016] are summarized.

Fan et al. [2013] execute experiments on two real-world networks. They compare the algorithms introduced for the OPOAO as well as for the DOAM model with two heuristics, namely MaxDegree and Proximity. In the MaxDegree algorithm the vertices are selected as protectors with respect to the node degree in descending order. Whereas in the Proximity algorithm the nodes which are direct neighbors of rumors are selected as protectors.

For the OPOAO model the number of infected nodes of each algorithm is compared. The research from Fan et al. [2013] leads to the result that the greedy algorithm performs better than the two heuristics when the propagation hops are around 9 or higher. Whereas in the case of less propagation hops the latter two heuristics are slightly better.

Under the DOAM model the number of selected protectors as well as the number of influenced nodes are considered. In both categories the greedy algorithm proposed by Fan et al. [2013] performs better than the two heuristics except when the number of rumor originators is very small. Especially when the network is large and has a high density, the greedy algorithm is performing very well.

Budak et al. [2011] base their experiments on four regional networks from Facebook. The performance of the greedy algorithm as well as the behavior of the three heuristics are studied.
Under the MCICM with the high-effectiveness property and a time delay about 20% which means that the ratio of the delay of cascade L to the duration of cascade C is 0.2 all four methods exert well. When the time delay is raised to 50%, which means that the campaign L is started later than before, one can observe that the performance of all methods is quickly reduced. Increasing the delay to 70% has the effect that all methods have a very bad performance.
Under the COICM the largest infectees and degree centrality heuristics perform similar to the greedy method whereas the early infectees heuristic runs poorer. Since here no high-effectiveness property is assumed, the results are accordingly inferior to the findings considered under the other model.
When the MCICM is considered without the high-effectiveness property, then the greedy algorithm is too costly and therefore, the heuristics are taken into account. Again, the largest infectees and degree centrality heuristics perform similar and the early infectees worse. [Budak et al., 2011]

Pham et al. [2016] analyze three real world data sets. They compare the number of activated users when varying the size of the seed set and the number of propagation hops, respectively. One can observe that the heuristic algorithm performs best for all three data sets.
Furthermore, with increasing the number of initial active users resp. the number of propagation hops the quality difference between the heuristic algorithm and other methods increases as well. [Pham et al., 2016]

7. Conclusion

We have seen that Threshold models and Cascade models are often used to model diffusion processes in social networks. The TSS problem exists in many variants and authors describe it in various settings.

One extension of the TSS problem is considered in detail in Chapter 4, namely the LCIP which has the advantage of partial payments. For the special case of trees and a 100% adoption rate and equal influence of neighbors, two algorithms are introduced which solve the problem optimally. A TUM formulation of the problem on trees helps to formulate the problem in general.
For the case that the whole network is required to adopt, an algorithm is described which solves the LCIP on general graphs. For complete graphs and trees it solves the problem actually optimally.
A possible research direction could be to consider the case if not a 100% adoption rate is needed or the time is limited. [Gunnec et al., 2016]

A further extension is to consider the problem not only on one isolated network, but for multiple networks. Since people often join more than one social network this diversification is reasonable. Coupling schemes are introduced which bundle multiple social networks into one network. On the one hand, there are lossless coupling schemes which preserve all informations about the networks, but produce a rather big network. On the other hand, lossy coupling schemes are introduced which retain not exactly all properties of the networks, but have the advantage that the coupled network is quite small. Overall, the coupling schemes provide solutions near to the optimal solution.
Zhang et al. [2016b] want to concentrate in their further work on the problem when not every network uses the same diffusion model.

Since not only positive information can spread through social networks, but rumors as well, the LCRB problem is introduced. The purpose is to limit the diffusion of the rumor and to spread the truth. As underlying model the Cascade model is used.

It turns out that the LCRB-P is NP-hard but can be approximated with a ratio of $1 - \frac{1}{e}$. For the LCRB-D there is an algorithm with an approximation ratio of $O(ln\, n)$. Furthermore, the EIL problem is NP-hard. However, if it is submodular then it can be approximated with a ratio of $1 - \frac{1}{e}$.
Fan et al. [2013] suggest to consider the LCRB problem under diffusion models without the submodularity property.

Since the research area is relatively new there is much space for various research directions. New variants of models or problems could be considered or the existing algorithms and heuristics could be improved.

Bibliography

E. Ackerman, O. Ben-Zwi, and G. Wolfovitz. Combinatorial model and bounds for target set selection. *Theoretical Computer Science*, 411(44-46):4017–4022, 2010.

O. Ben-Zwi, D. Hermelin, D. Lokshtanov, and I. Newman. Treewidth governs the complexity of target set selection. *Discrete Optimization*, 8(1):87–96, 2011.

B. Bollobas. *Graph theory: an introductory course*, volume 63. Springer Science & Business Media, 2012.

A. Borodin, Y. Filmus, and J. Oren. Threshold models for competitive influence in social networks. In *Internet and Network Economics, volume 6484 of Lecture Notes in Computer Science*, pages 539–550. Springer, 2010.

C. Budak, D. Agrawal, and A. El Abbadi. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th International Conference on World Wide Web*, pages 665–674. ACM, 2011.

J. D. Camm, J. J. Cochran, D. J. Curry, and S. Kannan. Conjoint optimization: An exact branch-and-bound algorithm for the share-of-choice problem. *Management Science*, 52(3):435–447, 2006.

N. Chen. On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415, 2009.

C.-Y. Chiang, L.-H. Huang, B.-J. Li, J. Wu, and H.-G. Yeh. Some results on the target set selection problem. *Journal of Combinatorial Optimization*, 25(4):702–715, 2013.

M. Chronowski. Modeling the influence in social networks: A survey on the target set selection problem. Master's thesis, University of Vienna, 2014.

M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer programming*. Springer, 2014.

G. Cordasco, L. Gargano, A. A. Rescigno, and U. Vaccaro. Optimizing spread of influence in social networks via partial incentives. In *International Colloquium on Structural Information and Communication Complexity*, pages 119–134. Springer, 2015.

E. D. Demaine, M. Hajiaghayi, H. Mahini, D. L. Malec, S. Raghavan, A. Sawant, and M. Zadimoghadam. How to influence people with partial incentives. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 937–948. ACM, 2014.

P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66. ACM, 2001.

P. A. Dreyer and F. S. Roberts. Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion. *Discrete Applied Mathematics*, 157(7):1615–1627, 2009.

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

D. Easley and J. Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

L. Fan, Z. Lu, W. Wu, B. Thuraisingham, H. Ma, and Y. Bi. Least cost rumor blocking in social networks. In *2013 IEEE 33rd International Conference on Distributed Computing Systems (ICDCS)*, pages 540–549. IEEE, 2013.

R. S. Garfinkel and G. L. Nemhauser. *Integer programming*, volume 4. Wiley New York, 1972.

J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001a.

J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 2001:1, 2001b.

M. Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.

P. E. Green and A. M. Krieger. Recent contributions to optimal product positioning and buyer segmentation. *European Journal of Operational Research*, 41(2):127–141, 1989.

M. Grötschel, M. Jünger, and G. Reinelt. On the acyclic subgraph polytope. *Mathematical Programming*, 33(1):28–42, 1985.

D. Gunnec and S. Raghavan. Integrating social network effects in the share-of-choice problem. *Decision Sciences*, 2016.

D. Gunnec, S. Raghavan, and R. Zhang. The least cost influence problem. Technical report, University of Maryland, College Park, 2013.

D. Gunnec, S. Raghavan, and R. Zhang. Tailored incentives and least cost influence maximization on social networks. Technical report, University of Maryland, College Park, 2016.

X. He, G. Song, W. Chen, and Q. Jiang. Influence blocking maximization in social networks under the competitive linear threshold model. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 463–474. SIAM, 2012.

M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey. *50 years of integer programming 1958-2008: From the early years to the state-of-the-art*. Springer Science & Business Media, 2009.

D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.

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

D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.

R. Kohli and R. Krishnamurti. Optimal product design using conjoint analysis: Computational complexity and algorithms. *European Journal of Operational Research*, 40(2): 186–195, 1989.

T. M. Liggett. *Interacting particle systems*. Springer-Verlag, New York, 1985.

M. W. Macy. Chains of cooperation: Threshold effects in collective action. *American Sociological Review*, pages 730–747, 1991.

E. Mossel and S. Roch. Submodularity of influence in social networks: From local to global. *SIAM Journal on Computing*, 39(6):2176–2188, 2010.

N. P. Nguyen, G. Yan, M. T. Thai, and S. Eidenbenz. Containment of misinformation spread in online social networks. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 213–222. ACM, 2012.

A. Nichterlein, R. Niedermeier, J. Uhlmann, and M. Weller. On tractable cases of target set selection. *Social Network Analysis and Mining*, 3(2):233–256, 2013.

D. Peleg. Local majorities, coalitions and monopolies in graphs: a review. *Theoretical Computer Science*, 282(2):231–257, 2002.

C. V. Pham, M. T. Thai, D. Ha, D. Q. Ngo, and H. X. Hoang. Time-critical viral marketing strategy with the competition on online social networks. In *International Conference on Computational Social Networks*, pages 111–122. Springer, 2016.

S. Raghavan and R. Zhang. Weighted target set selection on social networks. Technical report, University of Maryland, 2015.

M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 61–70. ACM, 2002.

T. Schelling. *Micromotives and Macrobehavior*. New York, Norton, 1978.

Y. Shen, T. N. Dinh, H. Zhang, and M. T. Thai. Interest-matching information propagation in multiple online social networks. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1824–1828. ACM, 2012.

R. J. Trudeau. *Introduction to graph theory*. Courier Corporation, 2013.

T. W. Valente. *Network models of the diffusion of innovations.* Cresskill New Jersey Hampton Press, 1995.

D. B. West. *Introduction to graph theory*, volume 2. Prentice Hall Upper Saddle River, 2001.

H. P. Young. The diffusion of innovations in social networks. *The economy as an evolving complex system III: Current perspectives and future directions*, 267, 2006.

H. Zhang, A. Kuhnle, H. Zhang, and M. T. Thai. Detecting misinformation in online social networks before it is too late. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 541–548. IEEE, 2016a.

H. Zhang, D. T. Nguyen, H. Zhang, and M. T. Thai. Least cost influence maximization across multiple social networks. *IEEE/ACM Transactions on Networking (TON)*, 24 (2):929–939, 2016b.