Extended Generalized Blockmodeling

Micheli Knechtel Lessa

THÈSE DE DOCTORAT D'AVIGNON UNIVERSITÉ

École Doctorale N° 536
Sciences et Agronomie

Spécialité / Discipline de doctorat :
Informatique

Laboratoire de Mathématiques

Présentée par
Micheli Knechtel Lessa

Extended Generalized Blockmodeling

Soutenue publiquement Janvier 2021 devant un jury composé de :

Mme.	Flavia C. Bernardini	MCF/HDR, IC, Universidade Federal Fluminense	Rapporteur
M.	Thierry Mautor	MCF/HDR, LD, Université de Versailles	Rapporteur
M.	Eric Sanjuan	MCF/HDR, LIA, Université d'Avignon	Examinateur
M.	Daniel Gouron	MCF, LMA, Université d'Avignon	Examinateur
M.	Serigne Gueye	MCF/HDR, LIA, Université d'Avignon	Directeur de thèse

DOCTORATE THESIS OF AVIGNON UNIVERSITY

Doctoral School N° 536
Sciences and Agronomy

Specialty / Discipline of the doctorate:
Computer science

Mathematics Laboratory

Presented by
Micheli Knechtel Lessa

Extended Generalized Blockmodeling

Publicly defended January 2021 before a jury composed of :

Mme.	Flavia C. Bernardini	MCF/HDR, IC, Universidade Federal Fluminense	Rapporteur
M.	Thierry Mautor	MCF/HDR, LD, Université de Versailles	Rapporteur
M.	Eric Sanjuan	MCF/HDR, LIA, Université d'Avignon	Examinateur
M.	Daniel Gouron	MCF, LMA, Université d'Avignon	Examinateur
M.	Serigne Gueye	MCF/HDR, LIA, Université d'Avignon	Directeur de thèse

Abstract

Blockmodeling is a set of techniques initially designed to analyse social networks but whose practical interest becomes larger, as we will see further in this thesis for terminology graphs.

One of the goals of blockmodeling is to reduce a large, potentially incoherent network to a smaller comprehensible structure that can be interpreted more readily.

There is great interest in capturing the cluster structure of a network in terms of equivalences, blocks and partitions. Up to now, most blockmodeling methods are focused in fitting the network structure to only one type of structure pattern.

However, there are a variety of social networking applications in which it is interesting to consider more than one type of pattern simultaneously, so that the structure of the network can take the form of several indicators for underlying relationships.

The research question is, how to deal with the situations where an analyst has several relations types of relations for a set of actors. Thus, we propose an optimization model, which we call the extended generalized blockmodeling. The main objective of extended generalized blockmodeling is to find the partition size and the set of patterns that has the best representation of the network structure. The extended generalized blockmodeling expands the possibilities of the framework, making it possible to analyze networks without any prior knowledge about them.

The extended generalized block modeling belongs to the class of highly combinatorial problems, the exact method is only suitable for small networks, so the second question is how to make this approach viable for medium and large networks.

Therefore, we propose the first non-exact approach to generalized extended block modeling, based on the VNS algorithm as an alternative for medium-sized networks. Even though the results found by the heuristic may not be the best of all the solutions to the problem, the experiments show that it converges to a satisfactory result in a not prohibitively long time.

The third question, which we address in this thesis, is the extended generalized blockmodeling a suitable approach in the field of bibliometrics and Natural Language processing. To do so, we analyse the network of terms concerning terrorism research.

For all these questions, we demonstrate the numerical results, based on artificial

and real datasets benchmarks. These results allow the exploration of the application opportunities of the extended generalized block modeling as well as it's limitations.

Key words: social networks, blockmodeling, terminology graphs, heuristics, VNS Heuristic.

Résumé

Le blockmodeling est un ensemble de techniques initialement conçues pour analyser les réseaux sociaux mais dont l'intérêt pratique devient plus grand, comme nous le verrons plus loin dans cette thèse pour les graphes terminologiques.

L'un des objectifs du blockmodeling est de réduire un grand réseau potentiellement incohérent en une structure compréhensible plus petite qui peut être interprétée plus facilement.

Il y a un grand intérêt à capturer la structure de cluster d'un réseau en termes d'équivalences, de blocs et de partitions. Jusqu'à présent, la plupart des méthodes de modélisation par blocs visent à adapter la structure du réseau à un seul type de modèle de structure.

La question de recherche est de savoir comment gérer les situations où un analyste a plusieurs types de relations pour un ensemble d'acteurs. Ainsi, nous proposons un modèle d'optimisation, que nous appelons le extended generalized blockmodeling. Le principal objectif de extended generalized blockmodeling est de trouver la taille de la partition et l'ensemble de modèles qui a la meilleure représentation de la structure du réseau.

Le extended generalized blockmodeling élargit les possibilités du framework, permettant d'analyser les réseaux sans aucune connaissance préalable à leur sujet.

Le extended generalized blockmodeling appartient à la classe des problèmes hautement combinatoires, la méthode exacte ne convient que pour les petits réseaux, donc la deuxième question est de savoir comment rendre cette approche viable pour les réseaux moyens et grands.

Par conséquent, nous proposons la première approche non exacte pour le extended generalized blockmodeling, basée sur l'algorithme VNS comme alternative pour les réseaux de taille moyenne. Même si les résultats trouvés par l'heuristique ne sont peut-être pas la meilleure de toutes les solutions au problème, les expériences montrent qu'elle converge vers un résultat satisfaisant dans un temps qui n'est pas prohibitif.

La troisième question, que nous abordons dans cette thèse, est le extended generalized blockmodeling, une approche appropriée dans le domaine de la bibliométrie et du traitement du langage naturel. Pour ce faire, nous analysons le réseau de termes concernant la recherche sur le terrorisme.

Pour toutes ces questions, nous démontrons les résultats numériques, basés sur des benchmarks de jeux de données artificiels et réels. Ces résultats permettent d'explorer les opportunités d'application de la modélisation de bloc généralisée étendue ainsi que ses limites.

Mots clés: réseaux sociaux, blockmodeling, graphes terminologiques, heuristique, VNS heuristique

Acknowledgements

This thesis would not have been possible without the support of a large number of fantastic people. I'm definitely lucky to be surrounded by so many good people.

First and foremost I would like to thank my thesis director Serigne Guey for providing guidance and feedback throughout my PhD.

I am thankful to Flávia Cristina Bernardine and Thierry Mautor for accepting to be reporters for my thesis.

I would also like to thank to Eric Sanjuan and Daniel Gourion for all valuable comments, scientific insights and advices.

Besides, my best thanks go to my husband Matthieu Romedenne and to my three best friends: Lynda Achour, Rym Smai and Alena Melnikava who never gave up on me and were always there to help me in any situation.

To conclude, I cannot forget to thank my family and friends for all the unconditional support in this five years of intense academic work.

Avignon, 22th of November 2019

Contents

Chapter 1

Introduction

More and more social networks become accessible and, even for small networks, the quantity of data that they carry can be quite huge. There is an increasing need of transforming this data into something more comprehensive which can allow social analysts to theorize and analyze data in order to extract information [19]. Blockmodeling is a framework to describe a social network as a small structure, which can be easily interpreted, by clustering units regarding some kind of equivalence [34].

The obtained structure is a reduced graph (whose nodes are the clusters), which is represented as a relational matrix, called image matrix. It can be specified or obtained as process result. The main idea of equivalence is to find predefined patterns of relations among actors of the network, where every predefined pattern also know as ideal block, has a social meaning.

The conventional part of the blockmodeling framework is designed to deal with two type of equivalences: structural equivalence and regular equivalence (formally defined in chapter 2). Structural equivalence is very strict [19] and does not correspond to most of the real relationships. Regular equivalence is an attempted to make it more flexible and is considered as a weak property.

The notion of equivalence has been merged into a more general framework where the relations between the clusters (the image matrix) must be as close as possible to ideal blocks. For instance, the complete ideal block, that is a block where all the entries are 1, correspond to the structural equivalence.

The generalized blockmodeling expand the possibilities of the framework, but also require some previous knowledge, as the size of the partition and a pre-definition of the ideal models. Extended generalized block-modeling, introduced here, makes it possible to analyze networks without any prior knowledge about them.

1.1 Motivation and Objectives

Blockmodeling was a tool originally designed to help social psychologists and sociologists get informations about the social groups, such as: membership, arrangement, and articulation of the members, by arranging social relational data into pre-defined structural layout, also called ideal block. Despite its origin, blockmodeling has also become a useful tool for discerning the structural groups of any network.

In general, blockmodeling provides a general framework for fitting data by using an explicit criterion function capturing the difference between structure pattern and network. However, despite this generality, there remain some open problems:

1. For small networks, be they real or artificial networks, these structures are relatively straightforward to discern, for an example take a look in figure 1.1. However, once networks are large or complex, identifying their structure becomes a difficult task [19].

2. To date, and to the best of our knowledge, all blockmodeling methods have focused on optimization approaches that require prior knowledge about the network, we can cite some of the most popular works [18], [39], [10] and [9] etc. In addition, current optimization techniques can deal only with networks with few nodes.

The objective of this work is to describe a new optimitzation approach that does not require any information about the network, and also analyze the strengths and weaknesses of the proposed method in the application field.

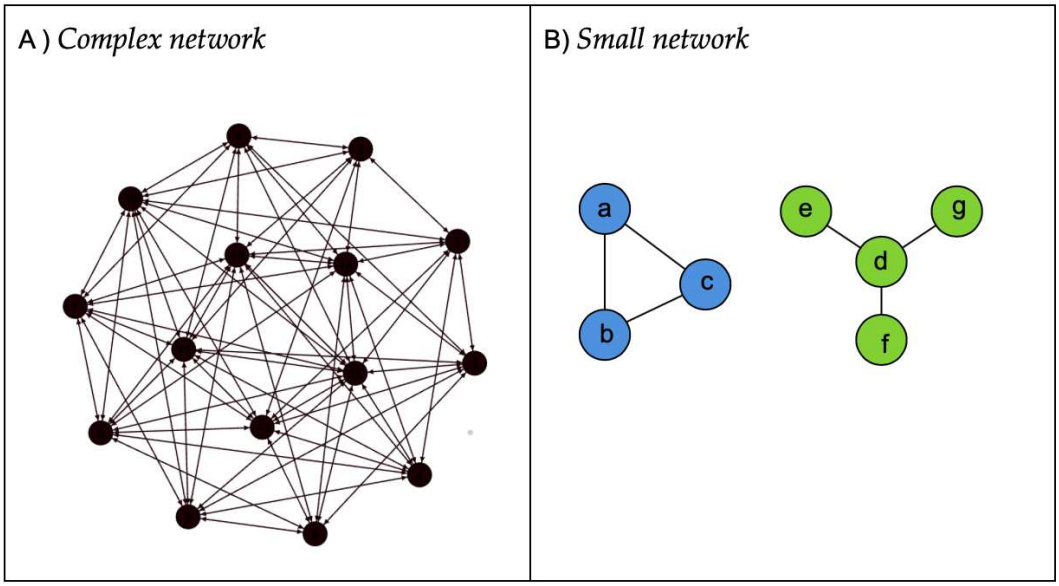


Figure 1.1: *Complex versus small networks*

1.2 Contributions of the Thesis

This thesis begins by investigating, in the first chapters, combinatorial optimisation techniques to address the blockmodeling problem without any prior information. The methodologies proposed and tested in the first part are then used to approach clustering problems on particular graphs called "terminology graphs". To our knowledge, applying blockmodeling concepts to analyse this graph have been never investigated.

The thesis contributions is this twofold in the optimization point of view:

1. We give a formalisation of the extended generalised blockmodeling, that according to our studies, is the first and most complete approach: it allows to analyse networks without any a prior knowledge about them. The other models require at least to know the size of the partition (i.e. the number of sub-sets that the partition will contain) and a pre-definition of the ideal models.
2. For the extend generalised block-modeling, we proposed the first integer linear programming model. To improve the model, we applied different optimisation techniques, to reduce the computational difficulty. But after all improvements, the model remains limited to small networks (no more than 20 nodes).
3. To solve larger instances, we designed heuristic algorithm, using variable neighborhood search techniques. Computational experiments showed that the VNS method provides very good (often proven optimal) solutions computed in short computing times.
4. Through an analysis of a terminological graph, we show, in another chapter, that generalised extended blockmodeling provides a flexible and effective method for analysing networks of terms, in particular to analyse thematic evolution through time.

1.3 Organization of the document

The document is organised as follows. Up to now, in this first chapter, an overview and structure of the thesis have been presented. The new section below will fix the notations used in this dissertation.

In the Chapter 2, basic notions of network analysis and blockmodeling are introduced. The notion of "equivalence" as the central concept in blockmodeling is reviewed. Lastly, we present the formalisation of extended generalised blockmodeling.

Chapter 3 presents the 0-1 linear programming model proposed for the problem. The decision variables, criterion function, required input, general and block type constrains are presented in detail. We also discusses the conceptual changes into blockmodeling framework, when changing some general constraints in the model. Further more in this chapter, we present the techniques applied to reduce the computational complexity and also an analysis of experimental results obtained and limitations.

Chapter 4, presents a variable neighbourhood search (VNS) heuristic algorithm specially designed for extended generalised blockmodeling. A literature review followed by comparative analysis between existent similar techniques are presented. Later on, in this chapter, the proposed (VNS) algorithm is described in details. The last section, of this chapter, presents preliminary results.

Chapter 5 concerns the application and evaluation of the ideas presented in chapter 3 and 4 in the field of terminology graph clustering. The chapter begins with a state-of-the-art on terminology graphs and relevant works done to decompose such types of graphs. Then the chapter advances to an analysis of the strengths and weaknesses of using extended generalised blockmodeling for these graphs. Finally, chapter 6 provides the conclusion of the results, discusses encountered difficulties and outlines future research directions.

1.4 Basic definitions

In this section, we formally define the basic definitions for the blockmodeling framework. We also introduce the notation, that we will use throughout this thesis.

1.5 Introduction to networks

Networks can be found everywhere, wherever we find sets of elements in which some pairs of elements are, in some sense, connected with each other, we can form a network. Elements can be people, but may also be groups, organizations, nation states, web sites, or scholarly publications etc. For each of these, finite number of relationship between pairs of elements, can be established. The most common examples are networks among people, organizations, countries etc. A formal definition of networks was already provided in section 1.5.2.

Since the 1970s, the empirical study of networks has played a central role in social science, and many of the mathematical and statistical tools used for studying networks have been first developed in sociology [36], but can be also applied in other fields.

These are certainly exciting times for the research community, besides the large group of networking applications that can be found nowadays, the techniques are normally general enought, to be broadly used for different types of applications.

1.5.1 Types of networks

In general, networks can be classified into different types according to several criteria. The most common types are based in the set of possible values that edge relations can take, as an example we can cite: binary, signed and valued networks. A binary network simply distinguishes whether there is a connection between pairs of elements. In a

signed network, each edge has a positive or negative sign, while in a valued network we measure connection between pairs.

For example « Andrea follows Rodrigo » on Twitter is a example of a binary relationship, « Max likes Alex's comments » on Facebook is signed relationship it encoded a positive sign, while « Anna retweeted 4 tweets from Andrea » is valued relationship, it quantifies edges relation. Another way to classify networks is by the nature of relationship, Facebook friends and LinkedIn connections require mutual confirmation, those are undirected network types. Following on Twitter and LiveJournal is Asymmetric relationship even if a follow-back tie can exist, it's directed network.

Another criterion used to classify networks is based on the number of different sets, if all elements come from the same set the network is referred as one-mode, if there are two sets then it is two-mode and so on. As one-mode network, we can take a league of baseball, there is only one set of elements, the players. Corporations employ people networks, these are composed of two different set of elements: corporations and people, so we can say it's two-mode network. Most of blockmodeling techniques deal with one-mode and two-mode networks, more than two is quite rare. This dissertation focuses on one-mode undirected binary networks.

1.5.2 Network

A network is defined as $G = (V, R)$ where $V = (v_1, \ldots, v_n)$ are the set of nodes and R set of edges. An edge is a pair (i, j) of nodes $i, j \in V$ that can be viewed in practice as a relation between i and j $R : (V \times V)$ can be represented by an adjacency matrix $S = S(G)$ whose rows and columns are indexed by the vertices of G, and $S_{kl} = 1$ if vertex v_k is connected to vertex v_l, zero otherwise.

1.5.3 Cluster and Clustering

A *cluster* c, as showed in table 1.3, is a set of nodes which share structural characteristics" defined in terms of their relations in R.

Sharing "structural characteristics means the way that the nodes composing the cluster are connected. If, for instance, we are interested in finding in a social network fully connected community, the "structural characteristic" of identified clusters will be the fact that all nodes are (almost) connected to all others. Many other characteristics, inside the clusters and between them may be drawn. These characteristics are at the root of the notion "ideal blocks" introduced latter.

A set of cluster forms a *clustering* $C = \{c_1, c_2, \ldots, c_n\}$ which is partition of the set $V : \bigcup_{k=1}^{n} c_k = V$, if $c_k \cap c_l = \emptyset$ and $\forall\, ki \neq l$. The clustering C also divide the edge relation $R : (V \times V)$ into blocks: $R(c_k, c_l) = E \cap c_k \times c_l$.

1.5.4 Block and ideal block

Let c_k and c_l be two clusters. A block is a sub-matrix of S whose lines are indexed by c_k components, and columns by c_l ones.

	a	b	c	b	e	f	g	h	i	j	l	m	n
a	0	0	1	1	1	0	0	0	0	0	0	0	0
b	1	0	1	0	0	0	0	0	0	0	1	0	0
c	1	0	0	1	0	0	0	0	0	0	1	0	0
d	1	1	1	0	0	0	0	0	0	0	0	0	0
e	1	0	1	1	0	0	0	0	0	0	0	1	1
f	0	0	0	0	1	0	0	0	0	0	0	0	0
g	0	1	0	0	0	0	0	1	1	0	0	0	0
h	0	1	0	0	0	0	1	0	1	0	0	0	0
i	0	1	0	0	0	0	1	1	0	0	0	0	0
j	1	0	0	0	0	1	0	0	0	0	0	0	0
l	1	1	0	0	0	0	0	0	0	1	0	0	0
m	1	0	0	0	0	1	0	0	0	1	0	0	0
n	0	0	0	0	1	1	0	0	0	0	0	0	0

Table 1.1: *Adjacency matrix S*

		c_1						c_2		c_3			c_4	
		a	b	e	f	j	l	m	n	g	h	i	c	d
c_1	a	0	0	1	0	0	0	0	0	0	0	0	1	1
	b	1	0	0	0	0	1	0	0	0	0	0	1	0
	e	1	0	0	0	0	0	0	0	0	0	0	1	1
	f	0	0	1	0	0	0	1	1	0	0	0	0	0
	j	1	0	0	1	0	0	0	0	0	0	0	0	0
	l	1	1	0	0	1	0	0	0	0	0	0	0	0
c_2	m	1	0	0	1	1	0	0	0	0	0	0	0	0
	n	0	0	1	1	0	0	0	0	0	0	0	0	0
c_3	g	0	1	0	0	0	0	0	0	0	1	1	0	0
	h	0	1	0	0	0	0	0	0	1	0	1	0	0
	i	0	1	0	0	0	0	0	0	1	1	0	0	0
c_4	c	1	0	0	0	0	1	0	0	0	0	0	0	1
	d	1	1	0	0	0	0	0	0	0	0	0	1	0

Table 1.2: *Clustering S into* $(k = 4)$

		c_1					
		a	b	e	f	j	l
c_3	g	0	1	0	0	0	0
	h	0	1	0	0	0	0
	i	0	1	0	0	0	0

Table 1.3: *block* (c_3, c_1)

In the case where $c_k = c_l$ the block is called diagonal, and non-diagonal otherwise. A block represents the relationships between c_k and c_l units. The way that clusters are connected (or not) gives meaningful informations about their relationships. Suppose, for instance, that $c_k \neq c_l$ and that all elements of c_k, are in relation with all elements

of c_l, and vice-et-versa: all elements of c_k, are in relation with all elements of c_l. If the network represents a social network, and arcs friend relationships, one may say that c_k and c_l individuals have the same friends. Similarly, if $c_k = c_l$, with each individual connected to each others, then we can say that the cluster c_k forms a community of friends. Notice in this example that the single important property is the fact that each member of the cluster c_k share the same relations with the other cluster c_l. The number of elements in c_k or c_l do not play any role. This property of c_k members is called "structural equivalence" and will be defined more formally later as well as with the notion of regular equivalence.

In graph terminology, the graphs induced by the relationships example are called complete bipartite graph (when $c_k \neq c_l$) and clique when $(c_k = c_l)$. Moreover, if we consider these relationships in terms of matrix values (block) we obtain rectangular or square block entirely composed of 1 values. The state-of-the-art in blockmodeling called it a "complete block". Many other block types have been considered in the literaturee defining a set of block called "ideal blocks". The corresponding exhaustive list is given below. In table 1.1 ideal blocks are given in matrix forms. In table 1.2 each form are described, accompanied by a practical interpretation in terms of social networks. Notice that a similar interpretation will be given in the case of terminology graphs (chapter 5). Figure 1.3 then gives some graphical illustrations.

Formally the set of ideal blocks will be noticed β. It is very important to keep in mind that β defined type of relations between clusters whatever is the number of components of each cluster.

When we say, for instance, that we want to find clusters in such a way that all non-diagonal blocks are ideally null, it means that we want clusters with no relation between them. In our model presented later we will distinguished diagonal to non-diagonal blocks, which give us flexibility to decide what ideal blocks, among β, we seek to find in the diagonal and off-diagonal part. For any block (c_k, c_l) to find, we notice $\beta(c_k, c_l) \subseteq \beta$ the set of ideal blocks associated to it.

	l		
k	1	1	1
	1	1	1
	1	1	1
	1	1	1

complete

	l		
k	0	0	0
	1	1	1
	0	0	1
	1	0	1

row-dominant

	l		
k	1	0	1
	1	0	0
	1	0	1
	1	1	0

column-dominant

	l		
k	1	0	0
	1	0	1
	0	1	1
	0	0	1

regular

	l		
k	1	0	1
	1	0	0
	1	0	1
	0	0	1

row-regular

	l		
k	1	1	0
	0	0	0
	1	0	1
	0	0	1

column-regular

	l		
k	0	0	0
	0	0	0
	0	0	0
	0	0	0

null

	l			
k	0	0	0	1
	0	0	0	1
	0	1	0	0

row-functional

	l		
k	0	0	1
	0	1	0
	0	0	0
	1	0	0

column-functional

Table 1.4: *Block type adjacency matrix examples*

Block type	Description
Null	fully covered by 0
Complete	fully covered by 1
Regular	one covered rows and columns
Row regular	each row is one covered
Column regular	each column is one covered
Row dominant	there is a row filled with 1
Column dominant	there is a column filled with 1
Row functional	there is only one 1 in each row
Column functional	there is only one 1 in each column

Table 1.5: *Types of Connections*

1.6 Measures of network

In this section, we formally define some metrics used by social network analysis, presented in Chapter 2. Measures are a common method used to understand networks and their nodes. Node centrality concepts and measures help determine the importance of a node in a network [2].

Various centrality measures have been proposed over the years, the three most popular concepts in (SNA) are: closeness, degree centrality and betweenness. Besides centrality measures, we wil also mathematically define in this section: network density,

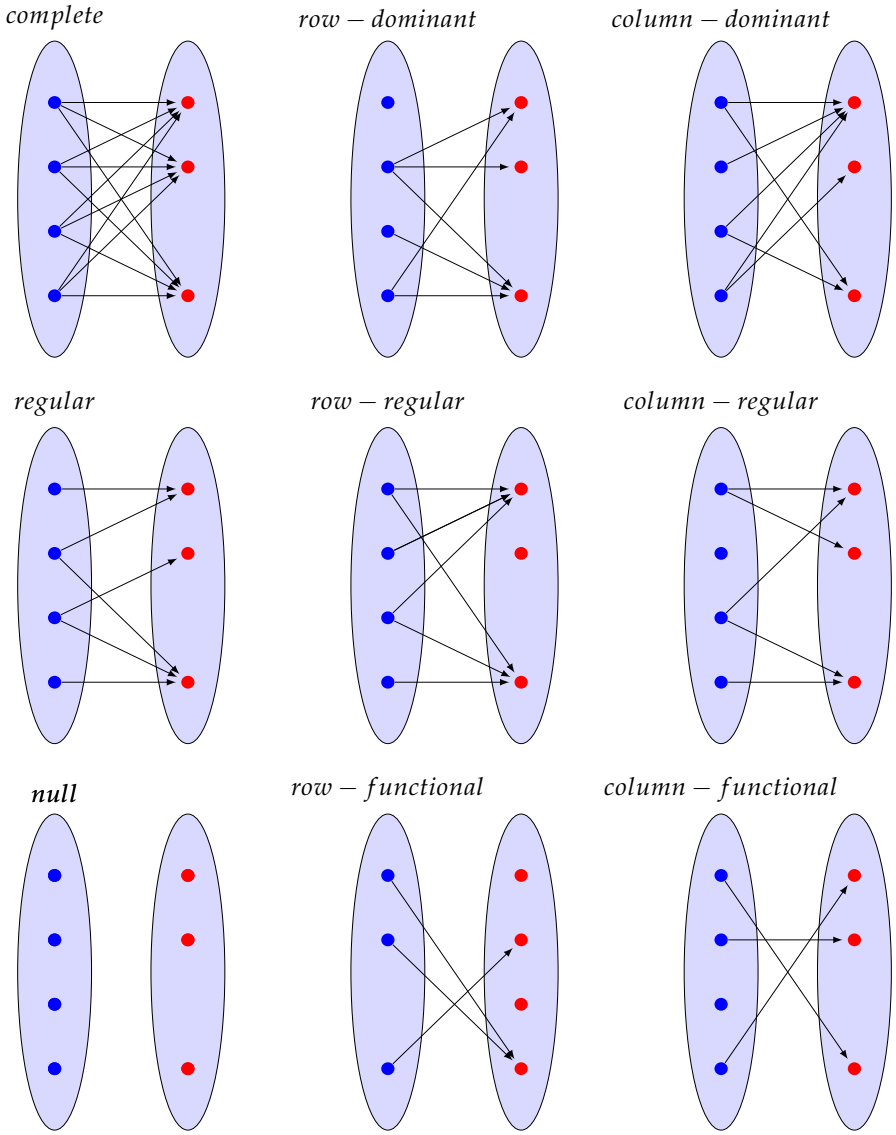


Figure 1.2: *Type of graph*

clustering coefficient, geodesic distance and diameter, that are also measures widely used to analyse social networks.

1.6.1 Closeness centrality

For a graph G with n number of nodes with $n = |V|$, the closeness centrality of a node v_i is represented mathematically as:

$$closeness(v_j) = \frac{n-1}{\sum_{i=1}^{n}(v_i, v_j)}$$

where $d(v_i, v_x)$ denotes the geodesic distance between (shortest paths) linking node v_i to node v_j.

1.6.2 Degree centrality

The easiest way of measuring a node's centrality is by counting the number of other nodes connected directly to this node, is a measure indicating the importance of node in the network.

$$degree(v_j) = \sum_{i=1}^{n} l(v_i, v_j)$$

where n is the number of nodes in the network and $l(v_i, v_j) = 1$ if and only if node i and j are connected, 0 otherwise.

1.6.3 Betweenness centrality

Regarding Alizera et al.(2012)[4] betweenness centrality measures the fraction of all shortest paths that pass through a given node or in simple terms it quantifies the number of times a node acts as a bridge along the shortest path between two other nodes. It can be expressed as:

$$betweenness(v_l) = \sum_{i \neq 1}^{n} \frac{g_{ij}(v_l)}{g_{ij}}; i \neq j \neq l$$

where g_{ij} is the geodesic distance (shortest paths) connecting v_i to node v_j and $g_{ij}(v_l)$ is the geodesic distance connecting v_i to node v_j that contains v_l.

1.6.4 Network density

Network density describes the portion of the possible connections, is a measure of the proportion of existent number of edges in the network divided by the possible maximal number of edges. For undirected simple graphs, the network density can be defined as:

$$density = \frac{|R|}{n(n-1)})$$

where R is the set of edges and $n = |V|$.

1.6.5 Clustering coefficient

The clustering coefficient of a graph G is the average over the clustering coefficients of its nodes. Let's, consider the following three nodes: v_i, v_j and v_l with mutual relations, relations between v_i and v_j as well as between v_i and v_l, it is supposed to represent the

likeliness that v_j and v_l are also related. The clustering coefficient for a graph can be expressed as:

$$clustering\ coefficient\ = \frac{3 \times number\ of\ triangles\ of\ node}{number\ of\ triples\ of\ node}$$

1.6.6 Geodesic distance

For a graph G the number of edges of the shortest path that connects the nodes v_i and v_i is called geodesic distance and denoted by $d(v_i, v_j)$.

1.6.7 Diameter distance

The graph diameter of a graph is the length $max_{(v_i,v_j)}d(v_i, v_j)$ of the longest shortest path between any two graph nodes (v_i, v_j).

Chapter 2

State-of-the-art in social network analysis and blockmodeling

Networks are data structure already formally defined in 1.5.2. Network analysis is the general name given to certain number of techniques and methodological tools for analyzing networks. Blockmodeling is a set of techniques initially designed to analyse social networks but whose practical interest becomes larger as we will see for terminology graphs.

This chapter begins with a state-of-the-art section on social network analysis motivations, contributions, and tools will be highlighted in Section 2.1. As one of one this tool in which this thesis is focused, blockmodeling concepts and current techniques will be reviewed in Section 2.5. In particular, the concept of equivalence will be introduced in section 2.3. The definitions, strengths and weaknesses of structural and regular equivalence are showed in subsections 2.3.1 and 2.3.2. Indirect and direct blockmodeling methodologies are reviewed. Generalised blockmodeling concepts are discussed in 2.6 and finally a new approach will be suggested in section 2.7.

2.1 Social network analysis

Social network analysis is a methodology for studying the connections and behavior of individuals within social groups [15].

Actors are usually people, but they can also be animals, organizations or nations. It can be applied to various problems in different disciplines, not limited to social networks: a reason why many authors use also the term «network analysis». Since this work is not limited to social networks, we use also the general terms as network analysis.

Network analysis concentrate its attention on how nodes are connected with each other and how these structure can be studied and analyzed, rather than treating indi-

viduals (persons, organizations, states) as discrete units of analysis, it focuses on how the structure of ties affects individuals and their relationships [31]. It looks to see the extent to which the structure and composition of ties affect the norms.

For social network analysis, "norms" is the concept that tells individuals how they should behave and what they should expect of others. It reduces the uncertainty and help to define "how things should be" in a particular group behaviors patterns imposed by group that are considered acceptable on a group [38].

According to Hogan [22] the roots of social network analysis are found in the mathematical study of graph theory and empirical studies of social psychology.

In figure 2.1 we display a taxonomy of graph theory approaches for social networking analysis research, based on the study done by Chelmis et al.(2011)[13].

The taxonomy divides SNA techniques into five area: random walks, temporal graphs, visualization, metrics and network structure.

Random walks concern to methods used to understand how information is spreading across the network. A temporal graph is a graph in which connections between vertices are active at specific times, and such temporal information leads to completely new patterns and knowledge that are not present in a non-temporal graph [26]. Visualization methods focus its efforts in how to understand network data and display the analysis result.

Metrics measure network properties. It is a useful tool for understanding the behavior and relationships in a network, so it has a special attention in this section. There are several types of metrics, the most common and often important are: centrality, diameter, geodesic distance, density and cluster coefficient.

Centrality concepts seeks to quantify the influence of every node in the network, several types have been proposed in the literature, leading to many different definitions. The most popular in SNA are: degree centrality, betweenness centrality and closeness centrality [20].

These metrics raise important information about the network and are formally defined in chapter 1, below we present the intuitive description.

- Closeness centrality calculates the shortest paths between all nodes, assigning to each node a score based on its sum of shortest paths that indicates the capacity of a node to be reached.
- Degree centrality gives an importance score based on the number of connections held by each node, it's usefull information for finding popular individuals.
- Betweenness centrality measures the number of times a node lies on the shortest path between other nodes, it shows which nodes act as "bridges", it's crucial information to identify nodes who influence the flow around a system.
- Clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.

- Geodesic distance measures the shortest path between two nodes.
- Diameter, on the other hand measures the distance of nodes in a network and is defined as the maximum geodesic distance between any pair of nodes.

Following taxonomy another classification that deserves attention is network structure. The network structure focuses its study on the network topology and its characteristics, it incorporates community detection techniques.

Communities structures are groups of vertices which share common properties within the graph. They are said to be present in a network, if it can be divided into sets of nodes, such that the connection between nodes of network follows a pattern.

As states by Farrag et al.(2007), detecting communities in networks is one of the fundamental approaches that provide a solution for networks disciplines where systems are often represented as graphs.

According Fortunato et al.(2012)[23] there has been a major effort in the last years to devise algorithms capable of extracting a complete information about the community structure of graphs. Community-detection facilitate informative visualization of social networks, and can be considered a summary of the whole network [21].

A number of community-detection algorithms and methods based in graph theory have been proposed in literature, it can be sub-divided into four categories:

1. Node-centered: refers to the methods in which each node of a group must satisfy certain properties, for example we can mention the algorithm to discover the maximum clicks.
2. Group-centered: these are methods that requires the total group to meet an explicit condition, for example, the group density is greater than or equal to a certain threshold, and remove the nodes with a grade below the typical average degree.
3. Network-centric: are the methods in which the goal is to partition the network nodes into disjoint sets. As examples, we can mention the following approaches: clustering supported vertex similarity using Jaccard similarity and Cosine similarit, Latent space models supported k-means clustering, blockmodeling approximation based on exchangeable graph models, spectral clustering are using minimum cut problem that the number of edges between the two sets is reduced and so on.
4. Hierarchy-centric: refers to the methods that construct a hierarchical structure of communities, as examples, we have divisive hierarchical grouping and agglomerative hierarchical grouping.

In this dissertation we focus our interest into non-overlapping blockmodeling techniques, based on the relation between community members. As we could see in this section, blockmodeling is one of many other tools conceived for network analysis that has its roots in graph theory and it's the focus of this thesis. It's presented in this chapter.

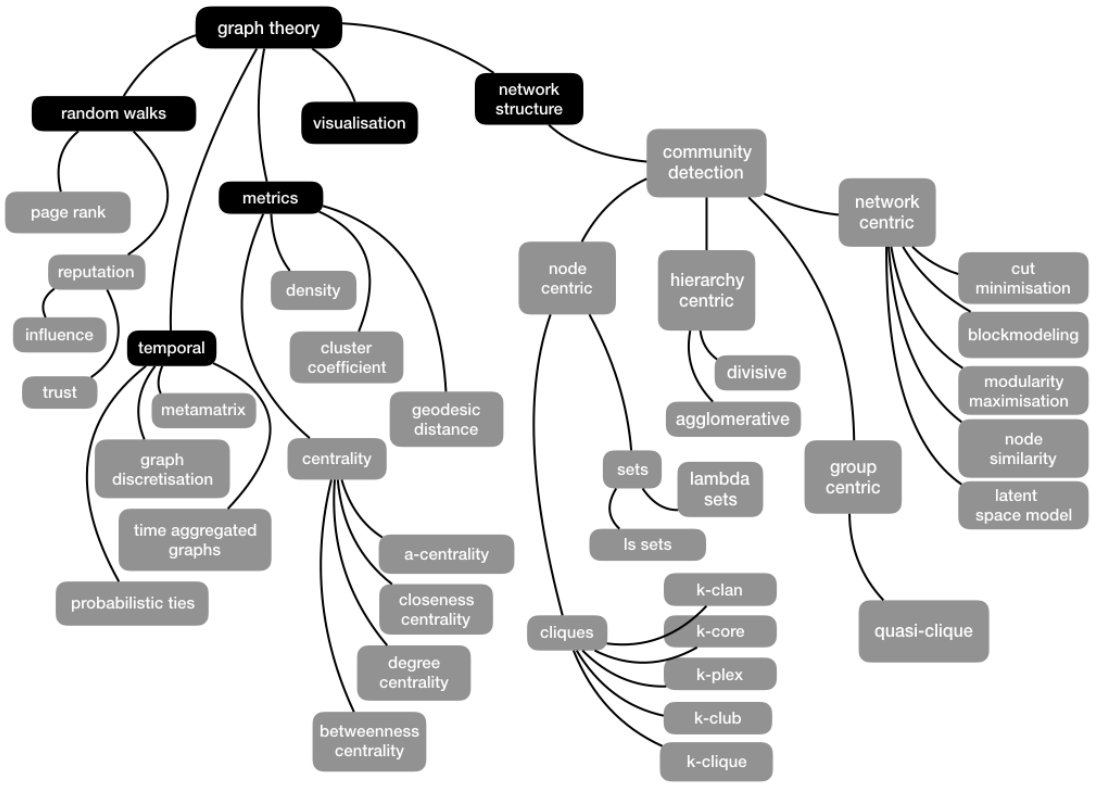


Figure 2.1: *A taxonomy of graph theory approaches in Social Networking Analysis Research*

2.2 Roles and positions

Positions and roles are key concepts in social network analysis. According to Wasserman and Faust (1994) [45] several SNA techniques to describe the structural properties of the network are concerned with the dual notions of social position and social role.

In social network analysis, these terms translate into procedures for analyzing structural similarities and patterns of node relationships. Those notions of position and role is defined in terms of equivalence (presented in next section 2.3) of node with respect to some formal mathematical property.

Structural equivalence, discussed in section 2.3.1, is one the set of property for defining position. Actors that are structurally equivalent are in identical "positions" with regard to all other actors, and they should be exactly substitutable in order to be considered to be in the same position.

Regular equivalence is the set of mathematical properties used to capture the social notion of role. Actors are said to be regularly equivalent if they have the similary relation with members of other sets. Hanneman (2005)[24] state that regular equivalence captures the idea of the "role" that an actor plays with respect to occupants of other

"roles" in a structure. An example is given in section 2.3.2 to illustrate and help to grasp the concept.

2.3 Equivalences for binary networks

An equivalence relation, in a mathematical point of view, is a binary relation defined in a given set of elements $(x, y, z, \ldots)$, and respecting a given set of properties. If the relation is noticed, let's say $\mathcal{R}$, the properties to respect are precisely: the reflexivity $(x\mathcal{R}x)$, the symetry $(x\mathcal{R}y \Longleftrightarrow y\mathcal{R}x)$ and the transitivity $(x\mathcal{R}y$ and $y\mathcal{R}z \Longrightarrow x\mathcal{R}z)$. The equivalence relation divides the set into partitions called equivalence classes. Each component in a equivalence class are then considered to be equivalent according to the equivalence relation.

To illustrate this standard concept, let us consider the classical equivalence relation in graph theory, defined on the node set (V) of a directed graph $G = (V, R)$. In this relation, two nodes x and y will be equivalent (i.e $x\mathcal{R}y$) iff there exist a directed path from x to y and another one from y to x. It can be shown that $\mathcal{R}$ is an equivalence relation. As we already know from basic graph theory concepts, the equivalence classes corresponding to this relation defined the strongly connected components of G (see []). Each component representing a set of nodes which are possible to reach (following a directed path) from any other node of the class.

If we look at clustering problems in the light of equivalence relation, one may interpret clusters as set of nodes sharing properties that makes them equivalent. Then clusters can be viewed as equivalence classes of an equivalence relation to define. As two different equivalence relations may lead to (possibly) two clusterings, one may remark that "good", "bad" or "optimal" clustering does not make sense if the properties defining the classes we want to compute is not given.

There is no "ideal" clustering unless the word "ideal" is clearly defined. We will see that in our case "ideal" will be related to the type of blocks (or subgraphs) we would like to see in the clustering. It is important also to see that defining only the equivalence relation is not sufficient to find the corresponding classes.

Implementing an efficient computational procedure is necessary in practice to derive the classes. For instance, to find strongly connected components, polynomial time algorithms based on depth (or breadth)-first-search algorithm are usually used. Thus another relation should be accompanied to another associated algorithm with which the corresponding clusters can be generated.

The researches on social network graphs and blockmodeling are at the origin of other equivalence relations such as : structural and regular equivalences.

2.3.1 Structural equivalence

Structural equivalence concept has been defined by Lorrain and White (1971). Before giving its formal definition, we introduce below the intuitions and motivations that support it. Intuitively, two nodes (individuals) will be structurally equivalent iff they are connected to the rest of the network in « identical » ways. « identical » means that the two nodes are linked to exactly the same nodes. The Wasserman and Faust Network is used as the following example [45].

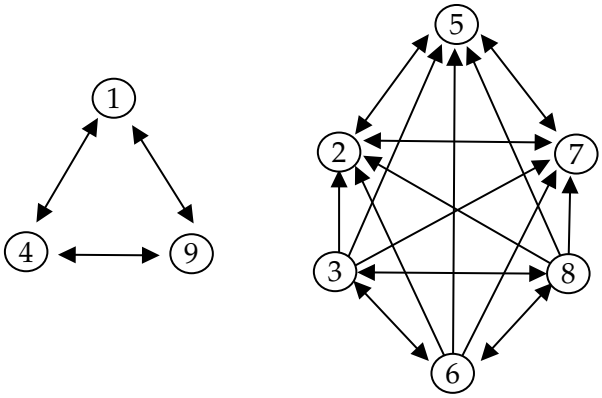


Figure 2.2: *Wasserman and Faust Network*

Classes built from structural equivalence are groups of individuals having the same connection to the same neighbours. We can also use the term "position" to refer to a collection of nodes that are structurally equivalent because they are in identical "positions" in the structure of the network. More formally the structural equivalence can be mathematically defined as follows:

If $G = (V, R)$ [1] *and* $\equiv$ *is an equivalence relation on* V *then* $\equiv$ *is a structural equivalence if and only if for all* $a, b, c \in V$ *such that* $a \neq c$ *and* $b \neq c$, $a \equiv b$ *implies:*

1. aRb *if and only if* bRa;
2. aRc *if and only if* bRc;
3. cRa *if and only if* cRb;
4. aRa *if and only if* bRb;

Structural equivalence clusters can be traduced in terms of blocks. To understand this, let us consider our previous example. The adjacency matrix of the graph is given below and we indicate in the line and column indices the clusters respecting structural equivalence.

Note that the blocks in table 2.2 are either null or complete. According to the definition of structural equivalence this is not a surprise as shown in the following result due to Batagelj, Ferligoj, and Doreian, 1992.

[1] V is used instead of P in the original definition to keep a uniform notation in the dissertation.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	0	0	0	1	0	0	0	0	1
2	0	0	0	0	1	0	1	0	0
3	1	0	0	1	0	1	0	1	1
4	1	0	0	0	0	0	0	0	1
5	0	1	0	0	0	0	1	0	0
6	1	0	1	1	0	0	0	1	1
7	0	1	0	0	1	0	0	0	0
8	1	0	1	1	0	1	0	0	1
9	1	0	0	1	0	0	0	0	0

Table 2.1: *Adjacency matrix*

		c_1			c_2			c_3		
		6	3	8	4	1	9	2	5	7
c_1	6	0	1	1	0	0	0	1	1	1
	3	1	0	1	0	0	0	1	1	1
	8	1	1	0	0	0	0	1	1	1
c_2	4	0	0	0	0	1	1	0	0	0
	1	0	0	0	1	0	1	0	0	0
	9	0	0	0	1	1	0	0	0	0
c_3	2	0	0	0	0	0	0	0	1	1
	5	0	0	0	0	0	0	1	0	1
	7	0	0	0	0	0	0	1	1	0

Table 2.2: *Permuted and partitioned adjacency matrix*

Theorem 2.3.1. *There are are four possible blocks respecting structural equivalence : null (all 0) block and complete (all 1) block; and for diagonal blocks also null block with 1's on the diagonal and complete block with 0's on the diagonal.*

0	1	1	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1

Table 2.3: *Examples of structural blocks*

Proof. see Batagelj, Ferligoj, and Doreian, 1992. □

These kind of blocks will belong to a set of "ideal" blocks in our generalized block-modeling problem.

2.3.2 Regular equivalence

Regular equivalence has been introduced by White and Reitz, 1983. Two nodes (individuals) will be regularly equivalent iff they are equivalent connected to equivalent others. Formally, if $G = (V, R)$ [2]and $\equiv$ is an equivalence relation on V then $\equiv$ is a regular equivalence if and only if for all $a, b, c \equiv V$, $a \equiv b$ implies:

1. aRc implies there exists $d \in V$ such that bRd and $d \equiv c$;
2. cRa implies there exists $d \in V$ such that dRb and $d \equiv c$.

The formal definition means that if a is regularly equivalent to c then if a is in relation with a node b then there must exist an individual d equivalent to b with which c is in relation. The concept is actually more easy to grasp intuitively than formally. As a practical example, let us consider the wife's class [24], composed only of married womens with children, and the husband's class, composed of married men with children, and finally the kids class, composed of children.

The wives, are considered *equivalent* because each one has a certain pattern of ties with a husband and child. Those women that belongs to the wife's class do not have ties to the same husband and child. Which means, they are not *structurally equivalent*, they have a different husband and child. They won't be automorphically [3] equivalent. But they are similar because they have the same relationship with members who belong to the same subset.

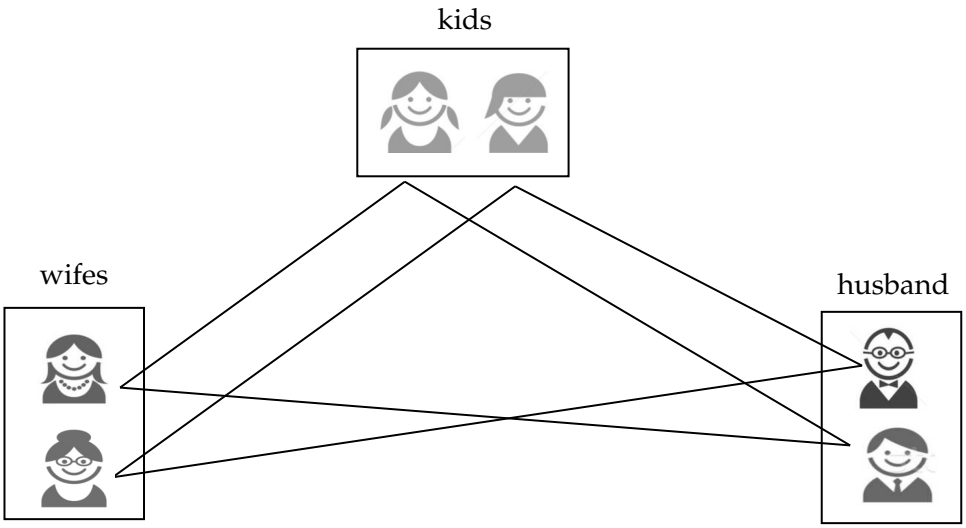


Figure 2.3: *Regular equivalence example*

Regarding Hanneman et al.(2005) [24], even if regular equivalence is a weak type of equivalence, due to a flexible definition, the concepts and methods used to identify and describe regular sets of equivalence are a good mathematical representation of the

[2] V is used instead of P in the original definition to keep a uniform notation in the dissertation.
[3] Automorphisms constitute the "symmetries" of a graph

sociological concept of role, centerpiece of most sociological theories, that strongly contribute to the development of networks analyses. Let's analyse the relationship between classes (groups).

Figure 2.4 displays three versions of Everett Network (Borgatti and Everett 1989), graph a) represent nodes and their connections, no classification is made, therefore all nodes have the same color. Graph b) is clustered according structural equivalence, in this scenario there are six colorings, meaning, six equivalent classes. Nodes of the same color belong to the same group, and have the same position in the network, as displayed in the image, they are connected in the exact same way. Graph c), is clustered regarding regular equivalence, thus the graph was divided into three equivalence classes.

As we can notice in figure 2.4, the idea is to bring out some main features of the network by clustering the nodes into categories of nodes depending on the type of equivalence, so the big question is, what are meaningful types to use ?

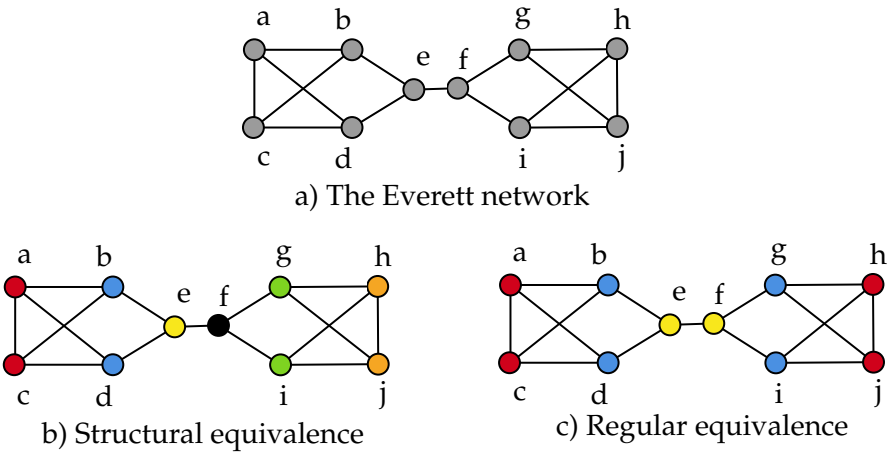


Figure 2.4: *Clustering by different type of equivalences*

Regardless of the equivalence relation considered we pointed out before that finding the corresponding classes need efficient computational procedures. Two types of procedures classified as direct or indirect by Batagelj et al. (1992) have been proposed in the litterature. We review them in the section below.

2.3.3 The usual steps for solving clustering problems

Doreian et al.(2005) [19]provide a list of usual steps for solving a clustering problem, the steps are:

1. Select the set of units V.
2. Measure the appropriate variables for the given problem. Variables can be measured with different scale types, they should be standardized.

3. Choose an appropriate dissimilarity between units for the given problem and types of variables used.
4. Define an appropriate criterion function to evaluate the selected type of clustering.
5. Define an algorithm for the given clustering problem.
6. Determine the clustering(s) that optimize(s) the chosen criterion function with the selected algorithm. An approximate solution may be necessary if there is no exact algorithm solution or if an excessive amount of computing time is needed to obtain an exact solution.

The first two steps do not require further discussion, the units and the type of relations must be defined before the analysis begins.

Regarding the choice of an appropriate dissimilarity, the third step of Doreian's list, it should be done carefully respecting the mathematical properties.

There are a large number of dissimilarities, therefore several studies in the literature concerning the choice of dissimilarity. For more information please check: Sokal and Sneath (1963), Clifford and Stephenson (1975), Everitt (1974), Gordon (1981), Lorr (1983), and Hubalek (1982).

Regarding step four, a criterion function can be constructed indirectly by means of appropriately defined (dis)similarity measures or directly by use of network data, and it's discussed in the next section. For the steps 5 and 6, we reserve chapter 3 and 4.

2.4 Indirect and directed approach

There are several blockmodeling procedures, Batagelj et al. (1992) proposed to classify into direct and indirect approaches.

1. Indirect approach: reduction to the standard data analysis problems (cluster analysis, multidimensional scaling) by determining a dissimilarity matrix between units which is compatible with the selected type of equivalence;
2. Direct approach: construction of a criterion function $P(C)$ which measures the fit of the clustering C to the network data, and solving the corresponding optimization problem.

2.4.1 Indirect approach

Indirect cluster approach

The objective of indirect cluster approach is to find, in a given network, a set of homogeneous or well separated clusters defined according to some measured variables.

It can be represented in general terms as the optimization problem which seeks for an partion C^*, such that:

$$P(C^*) = \min_{C \in \Phi} P(C) \tag{2.1}$$

where C is a partition of V, Φ is the set of all feasible partitions, and $P : \Phi \longrightarrow \mathbb{R}$ is a criterion function. The conventional blockmodeling is based on a criterion function constructed indirectly; in this case, Ward's (1963) [33] criterion function is the most used and is formulate as the problem of finding a partitions with k clusters, and P is defined as:

$$P(C) = \sum_{c \in C} \sum_{v \in c} d(f(v), t_c) \tag{2.2}$$

where $f(v) = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_n\}$ is the vector with the description values associate to the node v and t_c is the center of the cluster c and can be express as:

$$t_c = (\overline{\mathcal{V}}_{1c}, \overline{\mathcal{V}}_{2c}, ..., \overline{\mathcal{V}}_{nc}) \tag{2.3}$$

and $\overline{\mathcal{V}_{ic}}$ is the average of the variables $\mathcal{V}_i$, $i = 1, \ldots, n$, for the units from the cluster c, expressed as:

$$\overline{\mathcal{V}_{ic}} = \frac{\sum_{v \in c} \mathcal{V}_i(v)}{|c_i|} \tag{2.4}$$

and d is the euclidean distance.

As an example let's consider the set of five units $V = \{a, b, c, d, c\}$ that have measurements based on two variables $(\mathcal{V}_1, \mathcal{V}_2)$, as display in table 2.4.

	a	b	c	d	e
$\mathcal{V}_1$	1	2	3	5	5
$\mathcal{V}_2$	1	3	2	3	5

Table 2.4: *Variable measurements*

By using Ward's criterion function 2.2, where $t_c = (\overline{\mathcal{V}}_1, \overline{\mathcal{V}}_2)$ is the center of the cluster c and the dissimilarity d is the euclidean distance, we can caculate all possible partitions in to two clusters 2.5, as the following example:

$$\sum_{v \in c_1} d(f(a), t_{c_1}) = d((1,1) - (1,1)) = 0 \tag{2.5}$$

$$\sum_{v \in c_2} d(f(b), t_{c_2}) = d((2,3) - (\frac{2+3+5+5}{4}, \frac{3+2+3+5}{4})) = 1,77 \tag{2.6}$$

$$\sum_{v \in c_2} d(f(c), t_{c_2}) = d((3,2) - (\frac{2+3+5+5}{4}, \frac{3+2+3+5}{4})) = 1,46 \tag{2.7}$$

$$\sum_{v \in c_2} d(f(d), t_{c_2}) = d((5,3) - (\frac{2+3+5+5}{4}, \frac{3+2+3+5}{4})) = 1,27 \tag{2.8}$$

$$\sum_{v \in c_2} d(f(d), t_{c_2}) = d((5,5) - (\frac{2+3+5+5}{4}, \frac{3+2+3+5}{4})) = 2,15 \tag{2.9}$$

$$P(C_1) = 0 + 1,77 + 1,46 + 1,27 + 2,15 = 6,65 \tag{2.10}$$

the lowest value for $P(C_{15}) = 5.41$ and best partition $C^* = \{\{a,b,c\},\{d,e\}\}$

C	c_1	c_2	t_{c_1}	t_{c_2}	P(C)
1	a	bcde	(1.0, 1.0)	(3.75, 3.25)	6.65
2	b	acde	(2.0, 3.0)	(3.50, 2.75)	8.18
3	c	abde	(3.0, 2.0)	(3.25, 3.00)	8.67
4	d	abce	(5.0, 3.0)	(2.75, 2.75)	7.24
5	e	abcd	(5.0, 5.0)	(2.75, 2.25)	5.94
6	ab	cde	(1.5, 2.0)	(4.33, 3.33)	6.66
7	ac	bde	(2.0, 1.5)	(4.00, 3.67)	7.21
8	ad	bce	(3.0, 2.0)	(3.33, 3.33)	9.58
9	ae	bcd	(3.0, 3.0)	(3.33, 2.67)	9.48
10	bc	ade	(2.5, 2.5)	(3.67, 3.00)	8.48
11	bd	ace	(3.5, 3.0)	(3.00, 2.67)	9.34
12	be	acd	(3.5, 4.0)	(3.00, 2.00)	8.08
13	cd	abe	(4.0, 2.5)	(2.67, 3.00)	8.58
14	ce	abd	(4.0, 3.5)	(2.67, 2.33)	9.11
15	de	abc	(5.0, 4.0)	(2.00, 2.00)	5.41

Table 2.5: *All partitions and Values of the Criterion Function*

Indirect blockmodeling approach

The indirect blockmodeling approach is the one where a given partitioning of a network is done by defining a (dis)similarity measure between pairs of units first and then, on this basis, an appropriate clustering criterion function. For the indirect approach, some standard steps, as shown in figure 2.5, have to be considered. The process consists, first, of defining a node property[4] as description of the units. As an example of such node properties, we can cite the degree and distribution of triads.

[4] node properties are also similarity measure

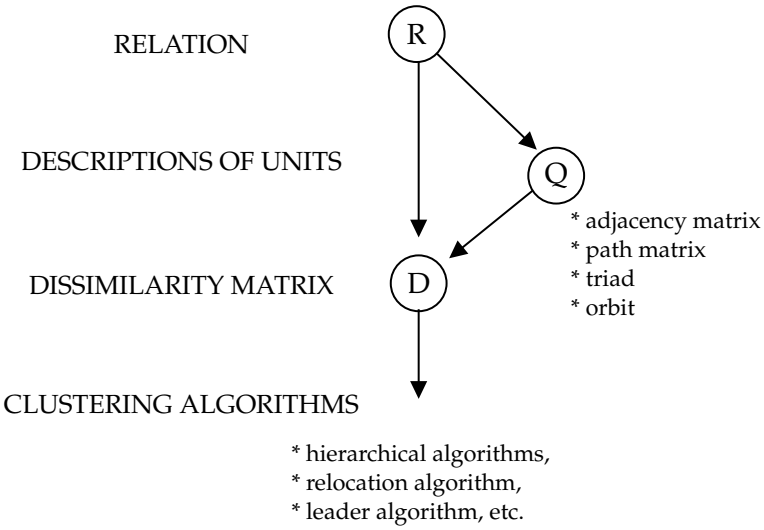


Figure 2.5: *Indirect blockmodeling schema*

The second step is to ensure that the selected description matches with the chosen type of equivalence. This mathematical exercise can be done for structural equivalence; Therefore, for other types of equivalence, it's quite complex to verify.

Batagelj (1991)[6] states that a property is considered a structural equivalent, if it depends only on the v unit and the relation R, without relying on the label of units. For instance, degree is a structural property and distribution of triads is a counterexample, to see the proof, please check out [6].

The next step is to calculate the matrix of dissimilarities based on description, it's an important step that must be done carefully, since the selected dissimilarity measure must also be compatible with the chosen equivalence. Some examples of dissimilarities are: manhattan distance, euclidean distance and Jaccard dissimilarity. The same mathematical exercise done for a unit property, should be done for dissimilarities, but now working with a pair of units, an example of can be find in Doreian et al.(2005:181)[19].

Once the description and the dissimilarity measure is defined, there are different efficient clustering algorithms that can be applied, such as hierarchical algorithms, the reallocation algorithm, and so on. For an application example of indirect method, let's consider Knoke directed information network, example given by [24].

The intuition here is based on structural equivalence. If we try to identify the nodes that are most similar to others, intuitively by looking at a graph in 2.6, we can see that nodes 2,5, and 7 are structurally similar since they have reciprocal ties with each other and almost everyone else.

The unit description is based in the degree properties, so when looking across the rows we can count out-degrees, and when looking down the columns, we can count in-degrees. Based in the degree similarities we can see which are the central nodes and which are the isolated ones.

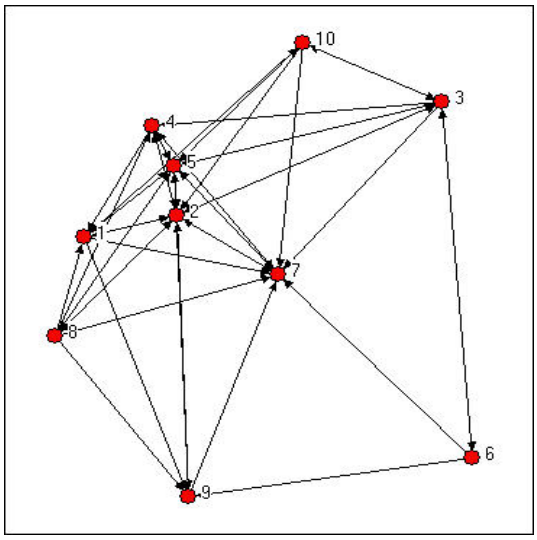


Figure 2.6: *Knoke directed information network*

Two nodes are said to be structurally equivalent if they have the same patterns of ties with other nodes, this means their scores in rows and columns is similar.

	1 Coun	2 Comm	3 Educ	4 Indu	5 Mayr	6 WRO	7 News	8 UWay	9 Welf	10 West
1 Coun	—	1	0	0	1	0	1	0	1	0
2 Comm	1	—	1	1	1	0	1	1	1	0
3 Educ	0	1	—	1	1	1	1	0	0	1
4 Indu	1	1	0	—	1	0	1	0	0	0
5 Mayr	1	1	1	1	—	0	1	1	1	1
6 WRO	0	0	1	0	0	—	1	0	1	0
7 News	0	1	0	1	1	0	—	0	0	0
8 UWay	1	1	0	1	1	0	1	—	1	0
9 Welf	0	1	0	0	1	0	1	0	—	0
10 West	1	1	1	0	1	0	1	0	0	—

Table 2.6: *Adjacency matrix for Knoke information network*

We can see the similarity of the nodes if we expand the matrix as showed in 2.6 by adding the row vector just after the column vector for each, like that each node can be represented as single column, as can be see in table 2.7.

The chosen dissimilarity measure is the Euclidean distance. The Euclidean distance between two vectors is equal to the square root of the sum of the squared differences between them.

Using the Ward's (1963) criterion function with the agglomerative algorithm, we have as a result the dendrogram in figure 2.8.

Another useful approach to understanding the bases of similarity and difference among sets of structurally equivalent actors is the block model, and a summary based on it called the image matrix. Because we working with raw adjacency matrix, the adjacencies can be turned into a valued measure of dissimilarity by calculating geodesic

1 Coun	2 Comm	3 Educ	4 Indu	5 Mayr	6 WRO	7 News	8 UWay	9 Welf	10 West
—	1	0	1	1	0	0	1	0	1
1	—	1	1	1	0	1	1	1	1
0	1	—	0	1	1	0	0	0	1
0	1	1	—	1	0	1	1	0	0
1	1	1	1	—	0	1	1	1	1
0	0	1	0	0	—	0	0	0	0
1	1	1	1	1	1	—	1	1	1
0	1	0	0	1	0	0	—	0	0
1	1	0	0	1	1	0	1	—	0
0	0	1	0	1	0	0	0	0	—
—	1	0	0	1	0	1	0	1	0
1	—	1	1	1	0	1	1	1	0
0	1	—	1	1	1	1	0	0	1
1	1	0	—	1	0	1	0	0	0
1	1	1	1	—	0	1	1	1	1
0	0	1	0	0	—	1	0	1	0
0	1	0	1	1	0	—	0	0	0
1	1	0	1	1	0	1	—	1	0
0	1	0	0	1	0	1	0	—	0
1	1	1	0	1	0	1	0	0	—

Table 2.7: *Concatenated row and column adjacencies for Knoke information network*

	1	2	3	4	5	6	7	8	9	10
1	0	2	2	1	2	2	1	1	0	1
2	2	0	2	2	1	2	2	1	2	2
3	2	2	0	2	2	2	1	2	2	2
4	1	2	2	0	2	2	1	1	1	1
5	2	1	2	2	0	2	2	1	2	2
6	2	2	2	2	2	0	2	2	2	2
7	1	2	1	1	2	2	0	1	1	2
8	1	1	2	1	1	2	1	0	1	2
9	0	2	2	1	2	2	1	1	0	1
10	1	2	2	1	2	2	2	2	1	0

Figure 2.7: *Euclidian distances in sending for Knoke information network*

distances, take a look at 2.9.

Even though it was widely used in the 1980s, according to Doreian et al. (2005: 183)[19], there are at least three serious problems in the indirect approach:

1. make sure that a (dis)similarity measures is compatible with selected equivalence.
2. Although this can be done for structural equivalence, it seems unlikely that such

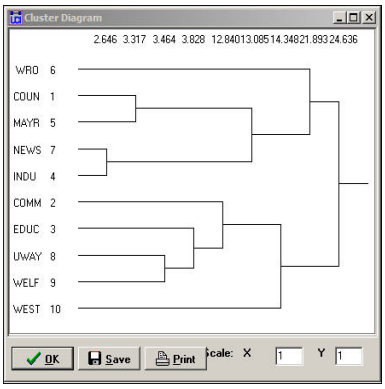


Figure 2.8: *Dendogram of structural equivalence data*

Measure: Euclidean Distance
Diagonal: Treat as Missing Values
Use geodesics? YES
Input dataset: C:\Documents and Settings\hanneman\My Documen

Structural Equivalence Matrix

	1 COUN	2 COMM	3 EDUC	4 INDU	5 MAYR	6 WRO	7 NEWS	8 UWAY	9 WELF	10 WEST
1	0.000	18.466	24.739	13.000	3.317	22.068	13.191	24.860	23.917	29.086
2	18.466	0.000	12.689	17.889	13.038	24.125	12.450	13.454	12.610	20.372
3	24.739	12.689	0.000	25.000	21.517	27.514	21.863	4.000	3.742	12.570
4	13.000	17.889	25.000	0.000	13.038	22.136	2.646	25.239	24.228	29.614
5	3.317	13.038	21.517	13.038	0.000	22.405	13.077	21.932	20.761	26.814
6	22.068	24.125	27.514	22.136	22.405	0.000	21.471	28.125	27.295	23.345
7	13.191	12.450	21.863	2.646	13.077	21.471	0.000	21.954	21.024	26.533
8	24.860	13.454	4.000	25.239	21.932	28.125	21.954	0.000	3.464	12.570
9	23.917	12.610	3.742	24.228	20.761	27.295	21.024	3.464	0.000	13.115
10	29.086	20.372	12.570	29.614	26.814	23.345	26.533	12.570	13.115	0.000

HIERARCHICAL CLUSTERING OF EQUIVALENCE MATRIX

Level	WRO 6	COUN 1	MAYR 5	NEWS 7	INDU 4	COMM 2	EDUC 3	UWAY 8	WELF 9	WEST 10
2.646	.	.	.	XXX	.	.	.	.	.	.
3.317	.	XXX	XXX	.	.	.	.	.	.	.
3.464	.	XXX	XXX	.	.	XXX	.			
3.828	.	XXX	XXX	.	XXXXX	.				
12.840	.	XXX	XXX	XXXXXXX	.					
13.085	.	XXXXXXX	XXXXXXX	.						
14.348	.	XXXXXXX	XXXXXXXXX							
21.893	XXXXXXXXX	XXXXXXXXX								
24.636	XXXXXXXXXXXXXXXXXX									

Figure 2.9: *Profile similarity of geodesic distances of rows and columns of Knoke information network*

a measure can be specified for regular equivalence.

3. if other equivalence types are specified, it will be necessary to construct compatible (dis)similarity measures for them also.

2.4.2 Direct approach

The direct approach to blockmodeling was first introduced in two papers in the journal Social Networks 14 (Batagelj et al., 1992b; Batagelj et al., 1992a). The first one (Batagelj et al., 1992b) introduced direct methods for structural equivalence and compared them with indirect methods, while the second one (Batagelj et al.,1992a) introduced methods for regular equivalence.

While indirect approaches require the use of (dis)similarity measures for all pairs of units and a compatibility study between selected measures and equivalence, the direct approach is more general.

The standard steps for the direct approach are illustrated in figure 2.10. In addition, since direct methods incorporate a criterion function into the procedure of searching for the optimal partition, we can be certain that if we check all possible partitions that the optimal partitions will be found. As being pointed out in the early comparisons (Batagelj et al., 1992b; Batagelj et al., 1992a), no such guarantee is offered by indirect methods.

The direct approach is the one where the criterion function is the center stage and it reflects directly the selected equivalence, it captures the discrepancy measure, between the partition and concern equivalence type. The procedure proposed in this work belongs to the family of algorithms that falls into the category of direct approaches, we present the following formal definition.

Assuming that a network is defined as $G = (V, R)$, where V is a set of units/nodes and R. Denoting β as the set of all equivalence relations of a selected type (e.g., regular or structural equivalence) over G, so every equivalence relation B on V defines a partion C. If we are able to construct a criterion function $P(C)$ with the following properties:

1. $P(C) \geq 0$
2. $P(C) = 0 \Leftrightarrow C \equiv B \Leftrightarrow B \in \beta$

By property 1, the minimal value of $P(C)$ is zero, by property 2, $P(C)$ is equal to zero, if and only if C corresponds to a clustering C associated to equivalence relation B.

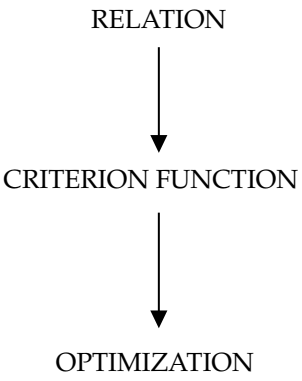


Figure 2.10: *Direct blockmodeling schema*

2.5 Conventional blockmodeling

The conventional blockmodeling is characterized by restricted attention to structural and regular equivalence; use of indirect and direct approach, with little or no attention to how well an established blockmodel fits the network data that have been blockmodeled.

According to Doreian, most of the previous empirical work on blockmodeling was done using the conventional model.

The conventional blockmodeling characterizes the fundamental ideas of blockmodeling and, even if it has restrictions, it is the starting point for broader concepts.

2.6 Generalized blockmodeling

In a chronological order of development, first came structural equivalence proposed by Lorrain and White (1971), which imposes very stringent properties. The term regular equivalence was first proposed by White and Reitz (1983). It's an attempt to relax structural equivalence, therefore, that is considered as a weak type of equivalence, because is too flexible and can fit to almost everything. Later on, searching for general mathematical theory, Everett and Borgatti (1994), published a paper where they mention a variety of other types of equivalences which are special cases of structural and regular equivalences. As mentioned in section 2.5, these early blockmodeling studies devoted to structural and regular equivalence are known in the literature as conventional block models, and are characterized by:

1. an indirect approach is used; does not deal with data directly, it needs to be transformed in some way and then clustered.
2. attention is strictly restricted to structural and regular equivalence, and induce block types are: null blocks, complete blocks, and regular (one covered for both rows and columns) blocks.

3. has little or no attention on how well the network fits to the blockmodels

Doreian et al.(1994) in their paper *"Partitioning networks based on generalized concepts of equivalence"* observe that different types of equivalence generally lead to different types of blocks. They also argue that an appropriate generalization of the idea of equivalence is one in which each block, from a particular image matrix, is free to conform to a different form of equivalence. So, instead of developing new equivalence types and deriving the set of allowed blocks, they propose to start with a set of permitted ideal blocks knowing that partitions established in terms of this new blocks will represent new equivalences. Generalized block modeling is characterized as a direct approach, where no transformation is required in the network data. Morever, a much broader set of block types is used to define equivalence instead of adding different equivalences in the definition. The last difference between conventional and generalized blockmodeling comes from the possibilities of pre-specification of block types (not only the allowed block types, but also their positions). In the generalized blockmodeling the criterion function is the center stage; its going evaluates a partition based on set of permitted blocks (ideal blocks). Let's formalize generalized blockmodeling as optimization problem, starting by repeating some definitions and notations:

- A network is defined as $N = (V, R)$, where V is a set of nodes $= \{v_1, v_2, \ldots, n\}$ and R a set of edges that can also be considered as a binary matrix $S : (v_i, v_j) \in R \Leftrightarrow S_{ij} = 1$.
- $C = \{c_1, c_2, \ldots, c_n\}$ is partition of the set V. Φ is a set of all feasible partitions. A partition C also partitions the relation R into blocks. Each such block consists of units belonging to clusters c_i and c_j and all arcs leading from cluster c_i to cluster c_j. If $i = j$, the block $R(c_i, c_j i)$ is called a diagonal block.
- Let $\beta(c_i, c_j)$ denote a set of all ideal blocks corresponding to block(c_i, c_j).

Then the global error of clustering C can be expressed as:

$$P(C) = \sum_{c_i, c_j \in C} \min_{B \in \beta(c_i, c_j)} (R(c_i, c_j), B) \tag{2.11}$$

where the term $d(R(c_i, c_j), B)$ measures the difference (error) between the block $R(c_i, c_j)$ and the ideal block B. As said before in subsection **??**, the are nine block types in the literature well defined: null, complete, regular, row-regular, column regular, column-dominant, row-dominant, row-functional and column-functional. The block type inconsistencies are computed as defined in table 2.8. The quantities used in the expressions for deviations have the following meaning:

- s_t total block sum = number of 1s in a block
- n_r cardinality i - number of rows in a block,
- n_c cardinality j - number of columns in a block,
- p_r number of non-null rows in a block,
- p_c number of non-null columns in a block,

- m_r maximal row sum,
- m_c maximal column sum,
- s_d diagonal block sum = number of 1s on a diagonal,
- d diagonal error = $\min(s_d, n_r - s_d)$
- Throughout the number of elements in a block is $n_r n_c$

Block type	Inconsistences $(R(c_i, c_j), B)$	Image matrix position
Null	s_t	off-diagonal
	$s_t + d - s_d$	diagonal
Complete	$n_r n_c - s_t$	off-diagonal
	$n_r n_c - s_t + d + 2s_d - n_r$	diagonal
Row-dominant	$(n_c - m_r - 1)\, n_r$	diagonal, $s_d = 0$otherwise
	$(n_c - m_r)\, n_r$	
Col-dominant	$(n_c - m_c - 1)\, n_r$	diagonal, $s_d = 0$otherwise
	$(n_c - m_c)\, n_r$	
Row-regular	$(n_r - p_r)\, n_c$	
Col-regular	$(n_c - p_c)\, n_r$	
Regular	$(n_r - p_r)\, n_c + (n_r - p_r)\, n_r$	
Row-functional	$s_t - p_r + (n_r - p_r)\, n_c$	
Col-functional	$s_t - p_c + (n_r - p_c)\, n_r$	

Table 2.8: *Deviations Measures for Types of Blocks*

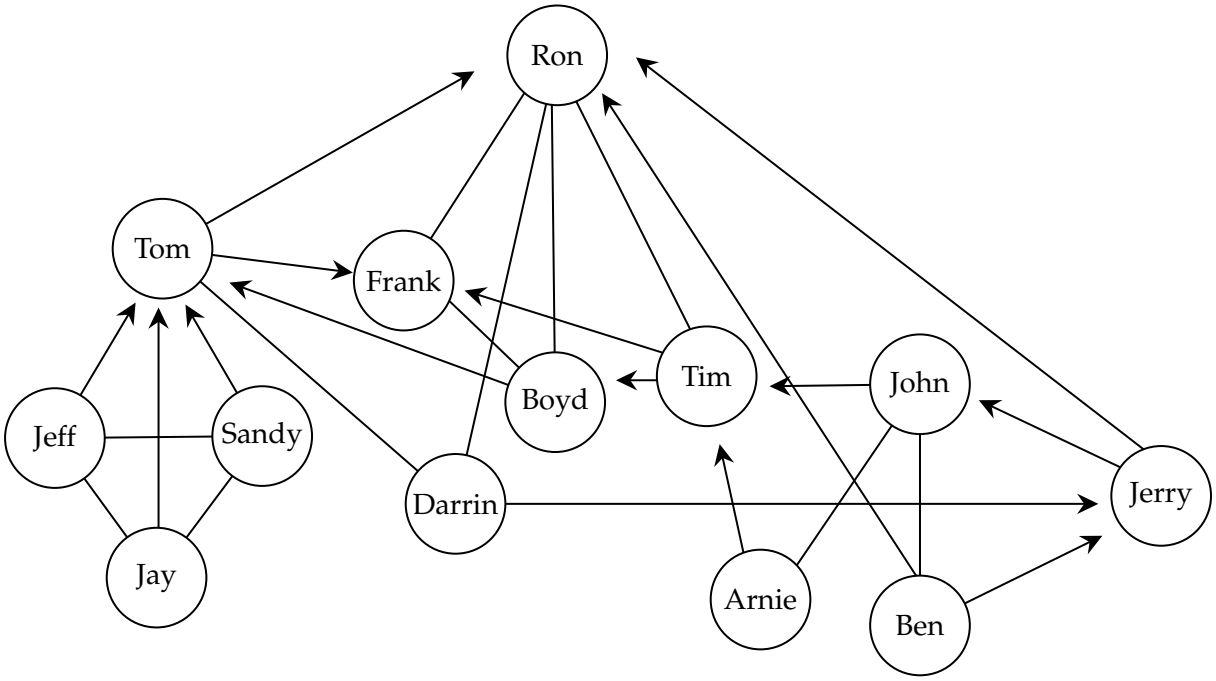


Figure 2.11: *Baseball team - Diagram*

To better illustrate the generalized blockmodeling thinking process, let's consider as

a practical example a baseball team network. The network consists of 13 boys, and each boy was invited to name his three best friends. The vertices are the boys and the edges represent the relationship "my best friend". The presence of a 1 indicates a best friend relationship between two boys and 0 represents the absence, more details in Table 2.9

		1	2	3	4	5	6	7	8	9	10	11	12	13
Ron	1	0	0	1	1	1	0	0	0	0	0	0	0	0
Tom	2	1	0	1	0	0	0	0	0	0	0	1	0	0
Frank	3	1	0	0	1	0	0	0	0	0	0	1	0	0
Boyd	4	1	1	1	0	0	0	0	0	0	0	0	0	0
Tim	5	1	0	1	1	0	0	0	0	0	0	0	1	1
John	6	0	0	0	0	1	0	0	0	0	0	0	0	0
Jeff	7	0	1	0	0	0	0	0	1	1	0	0	0	0
Jay	8	0	1	0	0	0	0	1	0	1	0	0	0	0
Sandy	9	0	1	0	0	0	0	1	1	0	0	0	0	0
Jerry	10	1	0	0	0	0	1	0	0	0	0	0	0	0
Darrin	11	1	1	0	0	0	0	0	0	0	1	0	0	0
Ben	12	1	0	0	0	0	1	0	0	0	1	0	0	0
Arnie	13	0	0	0	0	1	1	0	0	0	0	0	0	0

Table 2.9: *Baseball team - Adjacency matrix*

In Figure 2.11, Jeff's arrow towards Tom shows that Jeff likes Tom enough to rank him among his three best friends. Tom does not see Jeff in the same way. The same thing happened between Ben and Jerry, Ben has lines in one direction but not in the other. Boyd and Frank have a reciprocal connection to Ron. The rest of the figure can be read in the same fashion. All information concerning this relation is contained in the diagram.

By restricting our attention to structural equivalence connections, and assuming there are four clusters, and applying the direct approach, the best and unique partition is:

			1	3	4	5	2	6	10	11	12	13	7	8	9
c_1	Ron	1	0	1	1	1	0	0	0	0	0	0	0	0	0
	Frank	3	1	0	1	0	0	0	0	1	0	0	0	0	0
	Boyd	4	1	1	0	0	1	0	0	0	0	0	0	0	0
	Tim	5	1	1	1	0	0	0	0	0	0	0	0	0	0
c_2	Tom	2	1	1	0	0	0	0	0	1	0	0	0	0	0
c_3	John	6	0	0	0	1	0	0	0	0	1	1	0	0	0
	Jerry	10	1	0	0	0	0	1	0	0	0	0	0	0	0
	Darrin	11	1	0	0	0	1	0	1	0	0	0	0	0	0
	Ben	12	1	0	0	0	0	1	1	0	0	0	0	0	0
	Arnie	13	0	0	0	1	0	1	0	0	0	0	0	0	0
c_4	Jeff	7	0	0	0	0	1	0	0	0	0	0	0	1	1
	Jay	8	0	0	0	0	1	0	0	0	0	0	1	0	1
	Sandy	9	0	0	0	0	1	0	0	0	0	0	1	1	0

Table 2.10: *Baseball team partition with* $(k = 4)$ *based in structural equivalence*

and the closest ideal blockmodel pattern:

As result of direct approach, the criterion function return the number of inconsistencies, by block, of the partition, for this particular case, there are 20 inconsistencies associated with the partition, as displayed in Table 2.12.

The ideal blockmodel associated to the diagonal block (c_1, c_1) is complete with two

	c_1	c_2	c_3	c_4
c_1	complete	null	null	null
c_2	null	null	null	null
c_3	null	null	null	null
c_4	null	complete	null	complete

Table 2.11: *Baseball team image matrix for partition with* $(k = 4)$ *based in structural equivalence*

	c_1	c_2	c_3	c_4
c_1	2	1	1	0
c_2	2	0	1	0
c_3	5	1	7	0
c_4	0	0	0	0

Table 2.12: *Structural equivalent partition - Deviation error*

inconsistencies, because there is no connection from Frank to Tim, nor from Boyd to Tim. The block (c_1, c_2), has one inconsistency, Boyd has identified Tom as best friend in what otherwise would be a null block.

The same thing hapens with the block (c_1, c_3), where the connection between Frank and Darrin, is seen as an inconsistency, and so on.

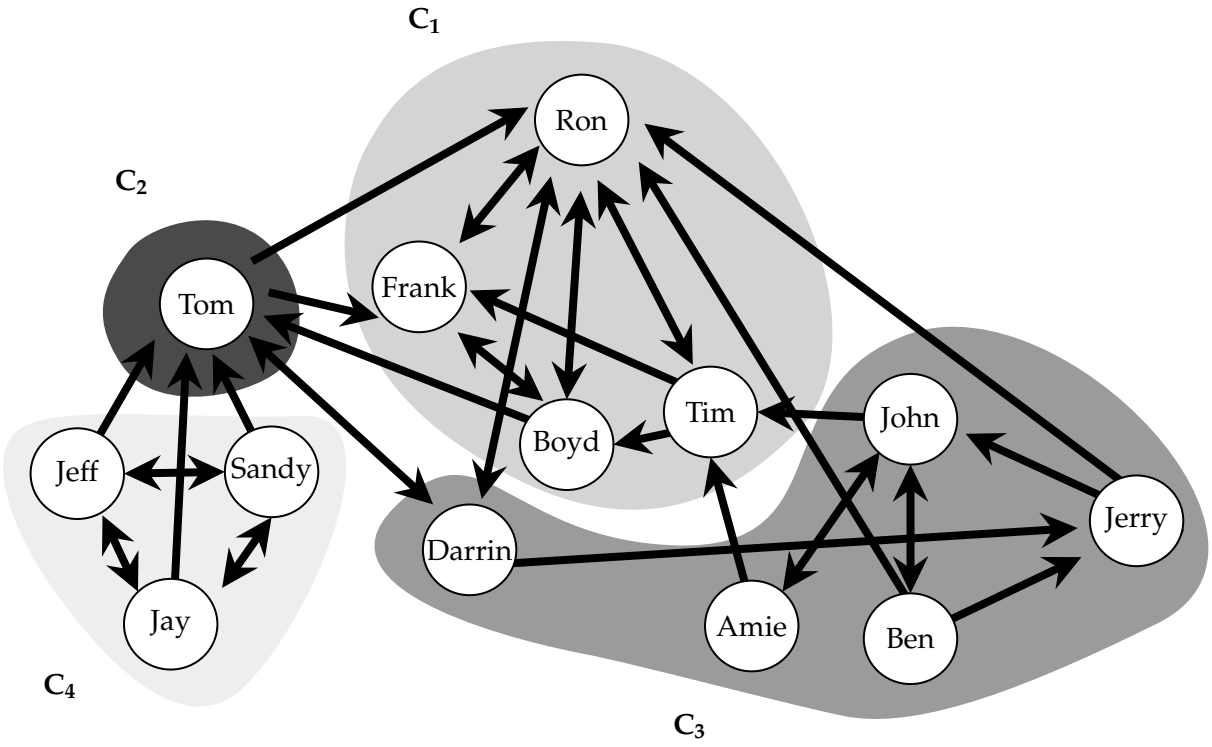


Figure 2.12: *Structural equivalent partition - Diagram*

The bigger number of inconsistency is concentrated in two blocks: (c_1, c_3) and (c_3, c_3), it can be verified in Table 2.12, it suggests taking a closer look at the blockmodeling con-

nections pattern and maybe partition size.

By using a relaxed the set of permitted block types and keeping the partition size as four, we have as best unique partition:

			1	2	5	6	10	11	12	13	7	8	9	3	4
c_1	Ron	1	0	0	1	0	0	0	0	0	0	0	0	1	1
	Tom	2	1	0	0	0	0	1	0	0	0	0	0	1	0
	Tim	5	1	0	0	0	0	0	0	0	0	0	0	1	1
	John	6	0	0	1	0	0	0	1	1	0	0	0	0	0
	Jerry	10	1	0	0	1	0	0	0	0	0	0	0	0	0
	Darrin	11	1	1	0	0	1	0	0	0	0	0	0	0	0
c_2	Ben	12	1	0	0	1	1	0	0	0	0	0	0	0	0
	Arnie	13	0	0	1	1	0	0	0	0	0	0	0	0	0
c_3	Jeff	7	0	1	0	0	0	0	0	0	0	1	1	0	0
	Jay	8	0	1	0	0	0	0	0	0	1	0	1	0	0
	Sandy	9	0	1	0	0	0	0	0	0	1	1	0	0	0
c_4	Frank	3	1	0	0	0	0	1	0	0	0	0	0	0	1
	Boyd	4	1	1	0	0	0	0	0	0	0	0	0	1	0

Table 2.13: *Partition with* $(k = 4)$ *based on generalized blockmodeling*

and connection pattern can be summarized as:

	c_1	c_2	c_3	c_4
c_1	regular	row-dominant	null	row-dominant
c_2	col-dominant	null	null	null
c_3	col-dominant	null	complete	null
c_4	col-dominant	null	null	complete

Table 2.14: *Baseball team image matrix for partition with* $(k = 4)$ *using generalized blockmodeling*

The error distribution across the blocks is equals zero, there are no inconsistencies:

	c_1	c_2	c_3	c_4
c_1	0	0	0	0
c_2	0	0	0	0
c_3	0	0	0	0
c_4	0	0	0	0

Table 2.15: *Generalized blockmodeling partition - Deviation error*

The partition is in exact accordance with the given set of ideal blocks types, it's a clear improvement. Figure 2.13 displays the network with the partition produced by generalized approach, that has an interesting interpretation.

To help the interpretation, the clusters are labeled as follows: c_1 corresponds to the cluster (Ron, Tom, Tim, John, Jerry, and Darrin); the cluster with (Ben and Arnie) is label as c_2; c_3 corresponds to cluster composed of Jeff, Jay, and Sandy; and c_4 is the labeled for the cluster (Frank and Boyd). Cluster c_3 is made up of actors that are between themselves structurally equivalent: internally they are a maximal complete subgraph. Externally they all send a tie to Tom, that belongs to cluster c_1, meaning that Tom is the element bridge between blocks c_3 and c_1. The nodes in c_3 are not connected to other clusters, the blocks: (c_3, c_2) and (c_3, c_4) are all null.

The internal structure of cluster c_1 is in accordance to the regular block type. As John is connected to Ben and Arnie, the block (c_1, c_2) is row-dominant. Similarly, Tim is connected to both Boyd and Frank, making the block (c_1, c_4) also row-dominant.

The block (c_2, c_1) is column-dominant, with Ben and Arnie connected to John. Since the nodes in c_2, are not connected to the other clusters, except c_1, the blocks (c_2, c_4) and (c_2, c_3) are all null. Because Frank and Boyd both cite Ron as best friend, the (c_4, c_1) is column-dominant. As these two nodes also have a reciprocal connection, the internal structure of the cluster is complete.

Finally, no connections between nodes from c_4 to c_2 or either c_3.

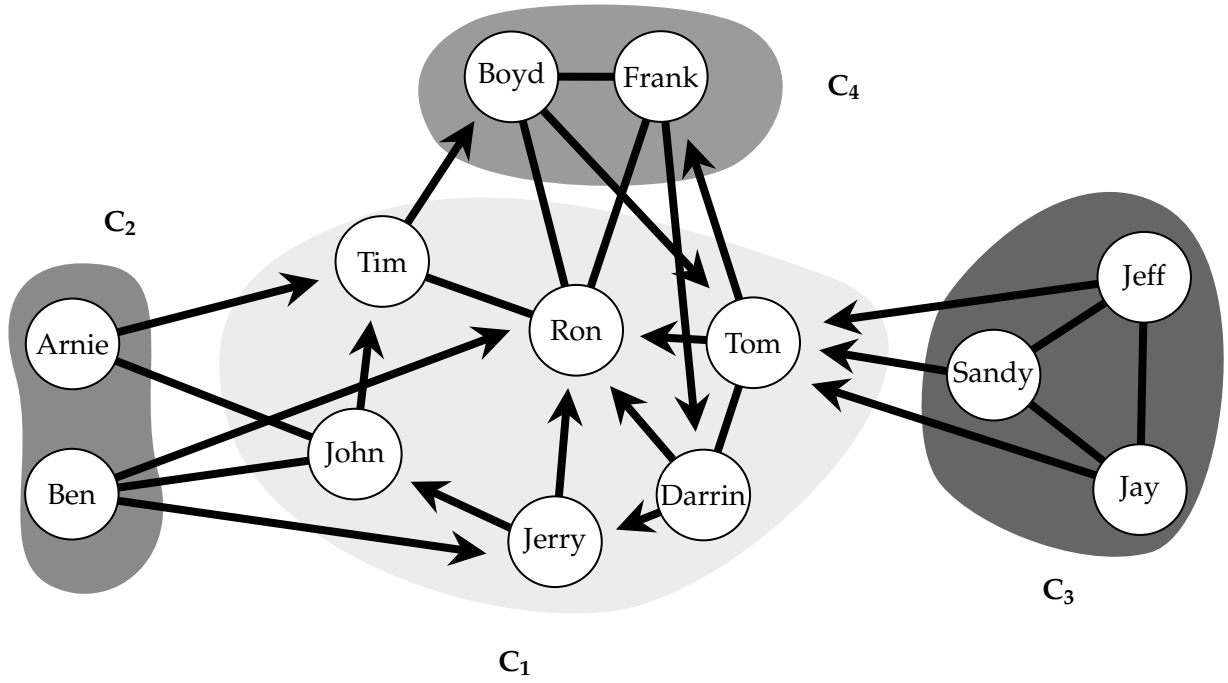


Figure 2.13: *Generalized blockmodeling - Diagram*

2.7 Extended generalized blockmodeling

As being mentioned earlier in this chapter, the conventional part of the block-modeling framework is designed to deal with two type of equivalences: structural equivalence and regular equivalence.

Structural equivalence is very strict and does not correspond to most of the real relationships. Regular equivalence is an attempt to make it more flexible and is considered as a weak property.

The notion of equivalence has been merged into a more general framework where the relations between the clusters (the image matrix) must be as close as possible to ideal

blocks. For instance, the complete ideal block, that is a block where all the entries are 1, correspond to the structural equivalence. The generalized block-modeling expands the possibilities of the framework, but also requires some previous knowledge, as the size of the partition and a pre-definition of the ideal models.

The extended generalized blockmodeling, introduced here, makes it possible to analyze networks without any prior knowledge about them. For a formalization, let us repeat some definitions and notations:

Given a clustering $C = \{c_1, c_2, \ldots, c_n\}$, let $\beta(c_i, c_j)$ denote the set of all ideal blocks corresponding to block $R(c_i, c_j)$. $\beta(c_i, c_j)$ defines the desired relationship between c_i and c_j. We note $\mathcal{F}_{ij}^B$ the error of clustering expressed as measures of difference from the ideal block $B \in \beta(c_i, c_j)$. $\mathcal{F}_{ij}^B$ is a distance between B and $R(c_i, c_j)$. Let e_{ij} be the minimal error for the block $R(c_i, c_j)$:

$$e_{ij} = \min_{B \in \beta(c_i, c_j)} \mathcal{F}_{ij}^B.$$

The global error is therefore:

$$P(C) = \sum_{c_i, c_j \in C} e_{ij}$$

Definition 2.7.1. *Let $N = (V, R)$ be the network and K be a integer, the generalized blockmodeling is defined as problem of finding a clustering C made of K clusters such that minimizes the sum of discrepancy measures between the investigated empirical block model and the set of ideal models.*

Definition 2.7.2. *The extended generalized blockmodeling is defined as problem of finding the partition size and the set of ideal blocks that reduces the global error. The extended generalized blockmodeling is the following combinatorial optimization problem :*

$$\min_{C \in \mathbb{C}} P(C)$$

That is the extended problem relaxes the cardinality constraint of the clustering and we search among all possible clusterings.

To get a concrete idea of the number of possible partitions, let's take as an example a set with five units $V = \{a, b, c, d, e\}$. Considering that the number of partitions is fixed to two, and the position of ideal blocks are pre-defined, we are going to have a set of 15 possible combinations.

In general, however, if there are n units, there are:

$$2^{n-1} - 1$$

different partitions with two clusters. The number of partitions exponentially increases with the number of units.

	c_1	c_2	*partition*
1	a	bcde	{a, bcde}
2	b	acde	{b, acde}
3	c	abde	{c, acde}
4	d	abce	{d, abce}
5	e	abcd	{e, abcd}
6	ab	cde	{ab, cde}
7	ac	bde	{ac, bde}
8	ad	bce	{ad, bce}
9	ae	bcd	{ae, bcd}
10	bc	ade	{bc, ade}
11	bd	ace	{bd, ace}
12	be	acd	{be, acd}
13	cd	abe	{cd, abe}
14	ce	abd	{ce, abd}
15	de	abc	{de, abc}

Table 2.16: *All partitions with* $k = 2$

For the extended generalized blockmodeling, where we search among all possible clusterings, meaning n units into k clusters, the number of all possible partitions is equal to the second-order Stirling number sum-up to the all possible combinations of the number of ideal blocks z in k positions :

$$\mathcal{S}(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^{k-i}\binom{k}{i}i^n + \binom{z}{k}$$

If we wanted to cluster the preceding five units into three clusters, we could search for the best clustering over the set of 25 partitions. In contrast, the number of all possible partitions of 50 units into 10 clusters is:

$$\mathcal{S}(50,10) = 562949953421311$$

Clearly, searching across all partitions to locate those partitions with the smallest value of a criterion function is a tough computational problem.

For general graphs, most clustering problems are NP-hard. On the other hand, 1-dimensional clustering problem can be solved efficiently. The complexity of most 2-dimenslonal clustering problems is not known [8].

In the next chapter we present the integer linear programing model proposed to tackle the extended generalized problem defined in this section.

2.7.1 Concluding Remarks

Chapter 3

ILP for Extended Generalized Blockmodeling

Blockmodeling is a technique which has been developed for analysing networks, and especially social networks (which are, most of the time, given through a binary relation between peoples or units). It consists in clustering (i.e. partitioning) the nodes of the network in such a way that the relationships between two clusters as well as the relationships inside a cluster match as well as possible some predefined models (the so-called ideal blocks). In this chapter, we will provide the first Combinatorial Optimization model to Blockmodeling. We will provide numerical experiments and general comments on the approach.

3.1 Introduction

The aim of this chapter is to summarize the model that has been implemented and numerically tested. It is assumed that the set of ideal blocks (the targets of the clusterisation) are the following ones :

1. Null block: this block is made of "zeros". Actually, according to the kind of block to which it is assigned, there exists 3 types of null blocks. Whenever, the null block is assigned to a diagonal block, 3 variations have to be considered: Null block with a null diagonal, Null block with a diagonal of 1 and Null block without diagonal (without any request for the diagonal). Whenever the Null block is assigned to a non-diagonal block,its own diagonal does not make sense.
2. Complete block: This block is made of one with the same 3 variations as for Null block whenever the Complete block is assigned to a diagonal block.
3. Row Regular block: At least one 1 in each row.
4. Column Regular block: symmetric case of the Row Regular, each column has at least one 1.

5. Regular block: an ideal block which is both row and column regular.
6. Row Dominant block: this block must have at least one line of 1, except on the diagonal whenever the Row Dominant block is assigned to a diagonal block.
7. Column Dominant block. Symmetric Ideal block of the Row Dominant block (that is a Column Dominant block has a column of one - may be except on the diagonal).
8. Row Functional: same as the Row Regular block except that each line must have exactly one 1.
9. Column Functional: symmetric of the previous one, same as Column Regular, except that each column must contain exactly one 1.

The remainder of this chapter is organized as follows. In the next section, we give the data of our problem while at section 3.4 we present the decision variables, which basically consist of the set of quantities that need to be determined in order to solve our problem. The entire model is presented in section 3.5, where we start by presenting the objective function used to evaluate the quantitative criterions, imposed by the shared constraints 3.5.2 and 3.5.3 which are the set of inequality and equality that express the limitations on decisions. Note that modelling some of the above ideal blocks leads to introducing additional variables, that we will introduce whenever it will be needed. In section **??** we provide the numerical experiments and in the last section we present the conclusions and some remarks.

3.2 The proposed ILP approach

3.3 Data

The data are the following ones:

- A relation matrix s, or a binary relation R, or a graph $G = (V, E)$ since the 3 entities are equivalent. In the remaining of the paper, we will work with the relation matrix s, that is: $s_{ij} = 1$ if and only if iRj, which, in turn, is equivalent to $(i, j) \in E$. Note that s is symmetric $\Leftrightarrow$ R is symmetric $\Leftrightarrow$ G is undirected.
- N the number of nodes in G.
- K the number of clusters that is requested by the user. In our generic approach, K is either the exact number of clusters that the user wishes or a maximum number of clusters (that is, the user can let the program choose the right number of clusters between 1 and K). Obviously, these two options lead to different models. In this paper, we will present the model whenever the program chooses the number of clusters.
- $\beta_{k,l}$ the set of ideal blocks that have to be considered for the block (k, l). In practice, it seems very difficult to establish such a set a priori. We therefore decided to

consider only two configurations (also to be defined by the user): β_d and β_o. β_d is the set of ideal blocks to be considered for the diagonal blocks, while β_o is the set of ideal blocks for off-diagonal blocks, i.e. $\forall k \in \{1, ..., K\}$, $\beta_{k,k} = \beta_d$ and $\forall k \in \{1, ..., K\}, \forall l \in \{1, ..., K\}, l \neq k$, $\beta_{k,l} = \beta_o$

- $P^b_{k,l}, \forall k \in \{1, ..., K\}, \forall l \in \{1, ..., K\}$ and $\forall b \in \beta_{k,l}$, a penalty coefficient for assigning b to (k, l).

3.4 Decision Variables

In this section, for the sake of clarity, we only introduce the "shared" decision variables, that are the decision variables which are indispensable for modelling the problem, whatever the sets $\beta_{k,l}$ are. Nevertheless, modelling some particular ideal blocks leads to introducing new variables. They will be listed when necessary.

- $F^b_{k,l}$ the error (or deviation) of block (k, l) if the ideal block b is chosen for (k, l).

$$F^b_{kl} = \begin{cases} \text{deviation of } (k,l) \text{ with respect to } b & \text{if } b \text{ is assigned to } k,l \\ 0 & \text{otherwise} \end{cases}$$

- $w^b_{kl} = \begin{cases} 1 & \text{if } b \text{ is assigned to } (k,l) \\ 0 & \text{otherwise} \end{cases}$
- $x_{ik} = \begin{cases} 1 & \text{if entity } i \text{ is assigned to cluster } k \\ 0 & \text{otherwise} \end{cases}$
- $y_{ijkl} = x_{ik}x_{jl} = \begin{cases} 1 & \text{if } i \text{ is assigned to cluster } k \text{ and } j \text{ to cluster } l \\ 0 & \text{otherwise} \end{cases}$
- $e_k = \begin{cases} 1 & \text{if cluster } k \text{ is empty} \\ 0 & \text{otherwise} \end{cases}$

  Note that this variable is forced to zero whenever the user wishes to fix the number of clusters.

We can now introduce the model.

3.5 Model

3.5.1 Objective Function

We want to minimize the total (and penalized) deviation:

$$\min \sum_{k=1}^{K} \sum_{l=1}^{K} \sum_{b \in \beta_{k,l}} P_{k,l}^{b} F_{k,l}^{b}$$

3.5.2 Shared Constraints

Shared Constraints are constraints which are independent from the ideal blocks.

- Each node i must belong to one cluster:

$$\sum_{k=1}^{K} x_{ik} = 1 \quad \forall i \in \{1...N\} \tag{3.1}$$

- A non-empty cluster muts have at least one node

$$\sum_{i=1}^{N} x_{ik} \geq 1 - e_k \quad \forall k \in \{1...K\} \tag{3.2}$$

- Each block must have an ideal block:

$$\sum_{b \in \beta_{k,l}} w_{ik}^{b} = 1 \quad \forall k \in \{1...K\} \quad \forall l \in \{1...K\} \tag{3.3}$$

- Linearisation constraints:

$$y_{ijkl} \leq x_{ik} \quad \forall k \in \{1...K\} \quad \forall l \in \{1...K\} \quad \forall i \in \{1...N\} \quad \forall j \in \{1...N\} \tag{3.4}$$

$$y_{ijkl} \leq x_{jl} \quad \forall k \in \{1...K\} \quad \forall l \in \{1...K\} \quad \forall i \in \{1...N\} \quad \forall j \in \{1...N\} \tag{3.5}$$

$$y_{ijkl} \geq x_{ik} + x_{jl} - 1 \quad \forall k \in \{1...K\}, \forall l \in \{1...K\}, \forall i \in \{1...N\}, \forall j \in \{1...N\}, i \neq j \tag{3.6}$$

3.5.3 Ideal Block Constraints

In this subsection, the model for computing the error for each block and each available ideal block are given.

Null Block Constraints

For a Null Block, the error is the number of 1 that the block (k, l) contains. As indicated previously, there exist 3 variations for the null block, when it is assigned to a diagonal block: with a zero diagonal, a one diagonal or without taking into account the diagonal. Hence, we will consider 3 constraints according to the kind of null block and to their position.

- If $k \neq l$, that is for a non-diagonal block or for $k = l$ without taking into account the diagonal (NBWD in the remaining):

$$F_{kl}^0 \geq \sum_{i=1}^{N} \sum_{j \neq i} s_{ij} y_{ijkl} + N^2 (w_{kl}^0 - 1) \quad \forall (k,l) \text{such that NBWD } \in \beta_{kl} \tag{3.7}$$

  Note that the maximum deviation for the Null Block is N^2 and, thus, if the Null block is not selected for (k, l), the right hand of the constraints becomes non positive, so that the constraint is no more active.

- if $k = l$ and the Null Block is actually a Null Block with diagonal of 1(NBDO in the remaining), then the error must also include the number of 0 on the diagonal. For this purpose, it is needed to introduce a new variable δ_k^0 to summarize the number of 0 on the diagonal. We then have to introduce a new constraint:

$$\delta_k^0 = \sum_{i=1}^{N} (1 - s_{ii}) x_{ik} \quad \forall k \in \{1, ..., K\} \tag{3.8}$$

  The constraint for computing the error is thus:

$$F_{kk}^1 \geq \delta_k^0 + \sum_{i=1}^{N} \sum_{j \neq i} s_{ij} y_{ijkk} + N^2 (w_{kk}^1 - 1) \quad \forall (k,l) \text{such that NBDO } \in \beta_{kl} \tag{3.9}$$

- if $k = l$ and the Null Block is actually a Null Block with diagonal of 0 (NBDZ), the deviation must now include the number of 1 on the diagonal. Hence, we have to introduce a new variable δ_k^1 and a new constraint:

$$\delta_k^1 = \sum_{i=1}^{N} s_{ii} x_{ik} \quad \forall k \in \{1, ..., K\} \tag{3.10}$$

  and then:

$$F_{kk}^2 \geq \delta_k^1 + \sum_{i=1}^{N} \sum_{j \neq i} s_{ij} y_{ijkk} + N^2 (w_{kk}^2 - 1) \quad \forall (k,l) \text{such that NBDZ } \in \beta_{kl} \tag{3.11}$$

Complete Block Constraints

The Complete block is quite similar to the Null Block, except, obviously, that it is full of 1. We thus have to consider the same three cases:

- If $k \neq l$, that is for a non-diagonal block or for $k = l$ without taking into account the diagonal (CBWD):

$$F_{kl}^3 \geq \sum_{i=1}^{N} \sum_{j \neq i} (1 - s_{ij}) y_{ijkl} + N^2 (w_{kl}^3 - 1) \quad \forall (k,l) \text{such that CBWD } \in \beta_{kl} \quad (3.12)$$

- if $k = l$ and the Complete Block is actually a Complete Block with diagonal of (CBDZ):

$$F_{kk}^4 \geq \delta_k^1 + \sum_{i=1}^{N} \sum_{j \neq i} (1 - s_{ij}) y_{ijkk} + N^2 (w_{kk}^4 - 1) \quad \forall (k,l) \text{such that CBDZ } \in \beta_{kl} \quad (3.13)$$

- if $k = l$ and the Complete Block is actually a Complete Block with diagonal of 1 (CBDO):

$$F_{kk}^5 \geq \delta_k^0 + \sum_{i=1}^{N} \sum_{j \neq i} (1 - s_{ij}) y_{ijkk} + N^2 (w_{kk}^5 - 1) \quad \forall (k,l) \text{such that CBDO } \in \beta_{kl} \quad (3.14)$$

Row Regular Block Constraints

For this block (RRB in the remaining), the deviation is actually the number of lines which are not 1-covered. In other words, we need to count the number of null lines. That can be done by introducing new variables:

$$\alpha_{ik} = \begin{cases} 1 & \text{if } i \text{ belongs to cluster } k \text{ and if the corresponding line is not 1-covered} \\ 0 & \text{otherwise} \end{cases}$$

We thus have:

$$\sum_{j \neq i} s_{ij} y_{ijkl} + \alpha_{ik} \geq x_{ik} \quad \forall i \in \{1, ..., N\}, \forall k \in \{1, ..., K\} \quad (3.15)$$

and:

$$F_{kl}^6 \geq \sum_{i=1}^{N} \alpha_{ik} + N(w_{kl}^6 - 1) \quad \forall (k,l) \text{such that RRB } \in \beta_{kl} \quad (3.16)$$

Column Regular Block Constraints

This is the symmetric case of the Row Regular Block (CRB in the remaining). We thus need to count the number of null lines in the block (k, l):

$$\gamma_{jl} = \begin{cases} 1 & \text{if } j \text{ belongs to cluster } l \text{ and if the corresponding column is not 1-covered} \\ 0 & \text{otherwise} \end{cases}$$

with the following constraint:

$$\sum_{i \neq j} s_{ij} y_{ijkl} + \gamma_{jl} \geq x_{jl} \quad \forall j \in \{1, ..., N\}, \forall l \in \{1, ..., K\} \tag{3.17}$$

and:

$$F^7_{kl} \geq \sum_{j=1}^{N} \gamma_{jl} + N(w^7_{kl} - 1) \quad \forall (k, l) \text{such that CRB } \in \beta_{kl} \tag{3.18}$$

Regular Block Constraints

This block (RB) is both Row and Column Regular, so that:

$$F^8_{kl} \geq \sum_{i=1}^{N} \gamma_{il} + \sum_{i=1}^{N} \alpha_{ik} + 2N(w^8_{kl} - 1) \quad \forall (k, l) \text{such that RB } \in \beta_{kl} \tag{3.19}$$

Row Dominant Block Constraints

This ideal block (RDB) must contain at least one row of 1, may be except on the diagonal (whenever it is assigned to a diagonal block). Once more, we actually have to consider several cases.

- If $k \neq l$, that is for a non-diagonal block, the deviation is thus the minimum number of 1 that we have to add to reach a complete line of 1. Hence, the "error" related to node i, if it belongs to cluster k, is exactly the difference between the number of columns in block (k, l) and the number of 1 on line i, that is:

$$\sum_{j=1}^{N} x_{jl} - \sum_{j=1, j \neq i}^{N} s_{ij} y_{ijkl}$$

We therefore need to introduce a new variable:

$$\vartheta_{ikl} = \begin{cases} \text{missing number of 1 in block } l \text{ on line } i & \text{if } i \text{ belongs to } k \\ 0 & \text{otherwise} \end{cases}$$

and a new constraint:

$$\vartheta_{ikl} \geq \sum_{j=1}^{N} x_{jl} - \sum_{j=1, j\neq i}^{N} s_{ij} y_{ijkl} + N(x_{ik} - 1)$$

Furthermore, since we are looking for the smallest number of 1, we must also have:

$$F_{kl}^{9} \leq \vartheta_{ikl}$$

and

$$F_{kl}^{9} \geq \sum_{i=1}^{N} z_{ikl}\vartheta_{ikl} + N(1 - w_{kl}^{9})$$

where z_{ikl} is a binary variable which indicates if i is achieving the smallest error for block (k, l). Thus, an additional constraint must be added:

$$\sum_{i=1}^{N} z_{ikl} = 1 - e_k$$

and, since we can't select i for achieving the minimum if it does not belong to block k:

$$z_{ikl} \leq x_{ik}$$

Nevertheless, we have to linearise the products $z_{ikl}\vartheta_{ikl}$ which implies to introduce a new and non-negative variable o_{ikl}. Finally, modelling the Row Dominant block for a non-diagonal block induces the following constraints:

$$\vartheta_{ikl} \geq \sum_{j=1}^{N} x_{jl} - \sum_{j=1, j\neq i}^{N} s_{ij} y_{ijkl} + N(x_{ik} - 1) \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.20}$$

$$F_{kl}^{9} \leq \vartheta_{ikl} \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.21}$$

$$F_{kl}^{9} \geq \sum_{i=1}^{N} o_{ikl} + N(1 - w_{kl}^{9}) \quad \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.22}$$

$$\sum_{i=1}^{N} z_{ikl} = 1 - e_k \quad \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.23}$$

$$z_{ikl} \leq x_{ik} \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.24}$$

$$o_{ikl} \leq N z_{ikl} \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.25}$$

$$o_{ikl} \leq \vartheta_{ikl} \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.26}$$

$$o_{ikl} \geq \vartheta_{ikl} + N(z_{ikl} - 1) \quad \forall i \in \{1...N\}, \forall (k,l) \text{s.t. RDWD } \in \beta_{kl} \tag{3.27}$$

- For a diagonal block ($k = l$), the Row Dominant Block without taking into account the diagonal, following the same steps as above, gives:

$$\vartheta_{ikk} \geq \sum_{j=1,j\neq i}^{N} x_{jl} - \sum_{j=1,j\neq i}^{N} s_{ij} y_{ijkk} + N(x_{ik} - 1) \tag{3.28}$$

$$F_{kk}^{10} \leq \vartheta_{ikk} \tag{3.29}$$

$$F_{kk}^{10} \geq \sum_{i=1}^{N} o_{ikk} + N(1 - w_{kk}^{10}) \tag{3.30}$$

$$\sum_{i=1}^{N} z_{ikk} = 1 - e_k \tag{3.31}$$

$$z_{ikk} \leq x_{ik} \tag{3.32}$$

$$o_{ikk} \leq N z_{ikk} \tag{3.33}$$

$$o_{ikk} \leq \vartheta_{ikk} \tag{3.34}$$

$$o_{ikk} \geq \vartheta_{ik} + N(z_{ikk} - 1) \tag{3.35}$$

- For a diagonal block ($k = l$), the Row Dominant Block with diagonal of 1, following the same steps as above, gives:

$$\vartheta_{ikk} \geq \sum_{j=1,j\neq i}^{N} x_{jl} - \sum_{j=1,j\neq i}^{N} s_{ij} y_{ijkk} + (1 - s_{ii}) x_{ik} + N(x_{ik} - 1) \tag{3.36}$$

$$F_{kk}^{11} \leq \vartheta_{ikk} \tag{3.37}$$

$$F_{kk}^{11} \geq \sum_{i=1}^{N} o_{ikk} + N(1 - w_{kk}^{11}) \tag{3.38}$$

$$\sum_{i=1}^{N} z_{ikk} = 1 - e_k \tag{3.39}$$

$$z_{ikk} \leq x_{ik} \tag{3.40}$$

$$o_{ikk} \leq N z_{ikk} \tag{3.41}$$

$$o_{ikk} \leq \vartheta_{ikk} \tag{3.42}$$

$$o_{ikk} \geq \vartheta_{ik} + N(z_{ikk} - 1) \tag{3.43}$$

- And for a diagonal block ($k = l$), the Row Dominant Block with diagonal of 0 gives:

$$\vartheta_{ikk} \geq \sum_{j=1,j\neq i}^{N} x_{jl} - \sum_{j=1,j\neq i}^{N} s_{ij} y_{ijkk} + s_{ii} x_{ik} + N(x_{ik} - 1) \tag{3.44}$$

$$F_{kk}^{12} \leq \vartheta_{ikk} \tag{3.45}$$

$$F_{kk}^{12} \geq \sum_{i=1}^{N} o_{ikk} + N(1 - w_{kk}^{12}) \tag{3.46}$$

$$\sum_{i=1}^{N} z_{ikk} = 1 - e_k \tag{3.47}$$

$$z_{ikk} \leq x_{ik} \tag{3.48}$$

$$o_{ikk} \leq N z_{ikk} \tag{3.49}$$

$$o_{ikk} \leq \vartheta_{ikk} \tag{3.50}$$

$$o_{ikk} \geq \vartheta_{ik} + N(z_{ikk} - 1) \tag{3.51}$$

Column Dominant Block Constraints

This is obviously the symmetric case of the Row Dominant Block. Hence, we still have 4 cases to consider:

- For a non-diagonal block (k, l):

$$\vartheta'_{jkl} \geq \sum_{i=1}^{N} x_{ik} - \sum_{i=1,i\neq j}^{N} s_{ij} y_{ijkl} + N(x_{jl} - 1) \tag{3.52}$$

$$F_{kl}^{13} \leq \vartheta_{jkl} \tag{3.53}$$

$$F_{kl}^{13} \geq \sum_{j=1}^{N} o'_{jkl} + N(1 - w_{kl}^{13}) \tag{3.54}$$

$$\sum_{j=1}^{N} z'_{jkl} = 1 - e_l \tag{3.55}$$

$$z'_{jkl} \leq x_{jl} \tag{3.56}$$

$$o'_{jkl} \leq N z'_{jkl} \tag{3.57}$$

$$o'_{jkl} \leq \vartheta'_{jkl} \tag{3.58}$$

$$o'_{jkl} \geq \vartheta'_{jkl} + N(z'_{jkl} - 1) \tag{3.59}$$

where ϑ', o' and z' are variables similar to the previous section.

- For a diagonal block ($k = l$), the Column Dominant Block without taking into account the diagonal gives:

$$\vartheta'_{jkk} \geq \sum_{i=1, j \neq i}^{N} x_{ik} - \sum_{i=1, i \neq j}^{N} s_{ij} y_{ijkk} + N(x_{jk} - 1) \tag{3.60}$$

$$F^{14}_{kk} \leq \vartheta'_{jkk} \tag{3.61}$$

$$F^{14}_{kk} \geq \sum_{j=1}^{N} o'_{jkk} + N(1 - w^{14}_{kk}) \tag{3.62}$$

$$\sum_{j=1}^{N} z'_{jkk} = 1 - e_k \tag{3.63}$$

$$z'_{jkk} \leq x_{jk} \tag{3.64}$$

$$o'_{jkk} \leq N z'_{jkk} \tag{3.65}$$

$$o'_{jkk} \leq \vartheta'_{jkk} \tag{3.66}$$

$$o'_{jkk} \geq \vartheta_{jk} + N(z_{jkk} - 1) \tag{3.67}$$

- For a diagonal block ($k = l$), the Column Dominant Block with diagonal of 1 gives:

$$\vartheta'_{jkk} \geq \sum_{i=1, i \neq j}^{N} x_{ik} - \sum_{i=1, i \neq j}^{N} s_{ij} y_{ijkk} + (1 - s_{jj}) x_{jk} + N(x_{jk} - 1) \tag{3.68}$$

$$F_{kk}^{15} \leq \vartheta'_{jkk} \tag{3.69}$$

$$F_{kk}^{15} \geq \sum_{j=1}^{N} o'_{jkk} + N(1 - w_{kk}^{15}) \tag{3.70}$$

$$\sum_{j=1}^{N} z'_{ikk} = 1 - e_k \tag{3.71}$$

$$z'_{jkk} \leq x_{jk} \tag{3.72}$$

$$o'_{jkk} \leq N z'_{jkk} \tag{3.73}$$

$$o'_{jkk} \leq \vartheta'_{jkk} \tag{3.74}$$

$$o'_{jkk} \geq \vartheta'_{jk} + N(z'_{jkk} - 1) \tag{3.75}$$

- And for a diagonal block ($k = l$), the Column Dominant Block with diagonal of 0 gives:

$$\vartheta'_{jkk} \geq \sum_{i=1, i\neq j}^{N} x_{ik} - \sum_{j=1, j\neq i}^{N} s_{ij} y_{ijkk} + s_{jj} x_{jk} + N(x_{jk} - 1) \tag{3.76}$$

$$F_{kk}^{16} \leq \vartheta'_{jkk} \tag{3.77}$$

$$F_{kk}^{16} \geq \sum_{j=1}^{N} o'_{jkk} + N(1 - w_{kk}^{16}) \tag{3.78}$$

$$\sum_{j=1}^{N} z'_{jkk} = 1 - e_k \tag{3.79}$$

$$z'_{jkk} \leq x_{jk} \tag{3.80}$$

$$o'_{jkk} \leq N z'_{jkk} \tag{3.81}$$

$$o_{jkk} \leq \vartheta'_{jkk} \tag{3.82}$$

$$o'_{jkk} \geq \vartheta'_{jk} + N(z'_{jkk} - 1) \tag{3.83}$$

Row Functional Block Constraints

The Row Functional Ideal Block (RFB in the remaining) is defined as having exactly one 1 per line. As a consequence, the deviation of block (k,l) with respect to the Row Functional Block is the sum over the lines of the error on each line which, in turn, is equal to the absolute value of the difference between the number of 1 on the line and 1. We thus need to introduce a new variable which will represent this absolute value (taking into account that we are minimizing the total deviation):

$$\theta_{ikl} = \begin{cases} |\sum_{j=1}^{N} s_{ij} y_{ijkl} - 1| & \text{if } i \text{ belongs to } k \\ 0 & \text{otherwise} \end{cases}$$

and the corresponding constraints:

$$\theta_{ikl} \geq -1 + \sum_{j=1}^{N} s_{ij} y_{ijkl} + N(x_{ik} - 1) \quad \forall i \in \{1, ..., N\}, \forall (k,l) \text{ s.t. RFB is in} \beta_{kl} \tag{3.84}$$

$$\theta_{ikl} \geq 1 - \sum_{j=1}^{N} s_{ij} y_{ijkl} + N(x_{ik} - 1) \quad \forall i \in \{1, ..., N\}, \forall (k,l) \text{ s.t. RFB is in} \beta_{kl} \tag{3.85}$$

which ensures that θ_{ikl} is actually equal to the absolute value whenever $i \in k$. The global deviation is then:

$$F_{kl}^{17} \geq \sum_{i=1}^{N} \theta_{ikl} + N^2 (w_{kl}^{17} - 1) \quad \forall (k,l) \text{ s.t. RFB is in} \beta_{kl} \tag{3.86}$$

Column Functional Block Constraints

The Column Functional Ideal Block (CFB in the remaining) is the symmetric of the Row Functional Block, so that we need to define:

$$\theta'_{jkl} = \begin{cases} |\sum_{i=1}^{N} s_{ij} y_{ijkl} - 1| & \text{if } j \text{ belongs to } l \\ 0 & \text{otherwise} \end{cases}$$

and to add the following constraints:

$$\theta'_{jkl} \geq -1 + \sum_{i=1}^{N} s_{ij} y_{ijkl} + N(x_{jl} - 1) \quad \forall j \in \{1, ..., N\}, \forall (k,l) \text{ s.t. CFB is in} \beta_{kl} \tag{3.87}$$

$$\theta'_{jkl} \geq 1 - \sum_{i=1}^{N} s_{ij} y_{ijkl} + N(x_{jl} - 1) \quad \forall j \in \{1, ..., N\}, \forall (k,l) \text{ s.t. CFB is in} \beta_{kl} \tag{3.88}$$

$$F_{kl}^{18} \geq \sum_{j=1}^{N} \theta'_{jkl} + N^2(w_{kl}^{18} - 1) \quad \forall (k,l) \text{ s.t. CFB is in} \beta_{kl} \tag{3.89}$$

3.6 Computational Experiment

To evaluate the scalability and behavior of ILP, we generated synthetic datasets using the LFR benchmark provided by Lancichinetti et al. (2008) [35]. The advantage of using a synthetic data is that it allows control of the data distributions used for testing so that it helps to make a fair comparison of algorithm performance. LFR is a benchmark solution that generates artificial networks which resemble real-world networks and also provide the built-in community structure.

Knowing that it is impossible to define a single metric that will provide a fair comparison in all possible situations, the benefits of LFR over other methods is that it represents the heterogeneity in the distributions of node degrees and community sizes. The heterogeneity distributions of node degree, whose tails often decay as power laws, is an important feature of real network responsible for a number of remarkable features, such as resilience to random failures/attacks [3], and the absence of a threshold for percolation [16] and epidemic spreading [37].

LFR requires as input parameters: the exponent values of the power law of the degree distribution γ and of the community-size distribution β, amount of inter-community connections defined by the mixing parameter μ, an average node degree k_{min}, and the maximum degree k_{max}.

The algorithm generates an unspecified number of communities with a defined number of vertices, with an average degree that can vary and are distributed according to a power-law. In addition, the community sizes also follow a power-law distribution, taking in account the amount of inter-community connections.

In our experiments we set the exponent values of γ and β to 2 and 1 respectively, following other networks examples (Clauset et al. 2009). The average degree k_{min} is adjust regarding the size of the networks, as it follow $k_{min} = \{4, 4N/9, 2N/3\}$ and $k_{max} = (N-1)$. Four different values of the mixing parameter $\mu = \{0.1, 0.3, 0.2, 0.6\}$ are used. The first one corresponds to well-separated communities, while the second one is not far from the limit where the networks have no community structure. For each configuration of the synthetic datasets, we generate 10 different community-model for simmetric and directed networks. In the presented set of experiments, the network size used are $N = 15$ and $N = 20$, once the proposed ILP model is limited to instances sizes where $N < 25$.

The results suggest that ILP Model is sensitive to changes in the mixing parameter μ, and in most examples, performances get better when the average degree increases and mixing parameter decreases. This may be due to an increase of the density inside clusters that makes the community more visible. On the other hand, if the links between clusters increase in the same proportion, the effect is distributed among many communities.

The ILP model presented in this chapter explores the symmetry of the problem by generating half the variable space for a non oriented graph. Such a strategy can be used since variables can be exchanged without changing the structure of the problem.

All reported results show on average symmetric networks perform better, even if time can vary significantly for instances with bigger values for k_{min}.

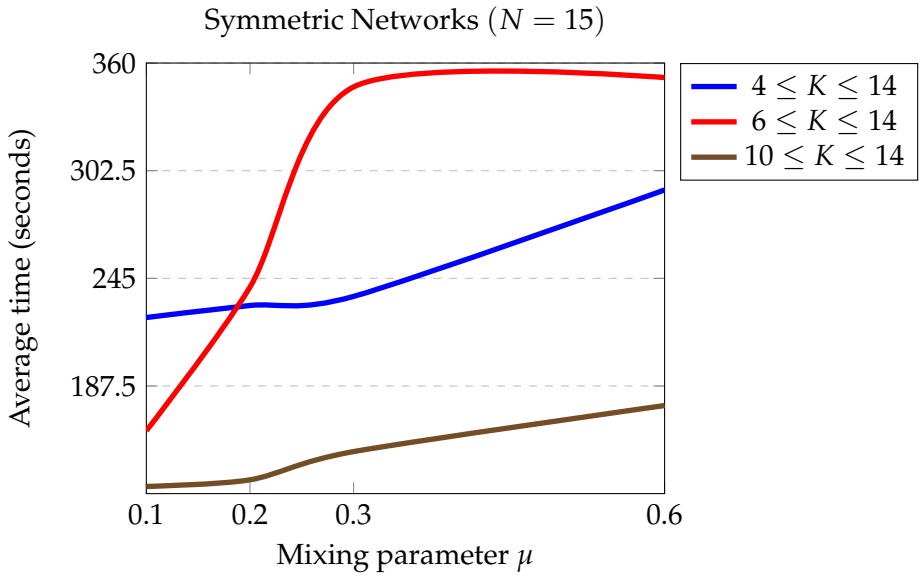
The experiments were conducted on anllllllll/ AMD 2.6 GHz server with 500 GB of memory and running Debian GNU/Linux 7 and developed using C++.

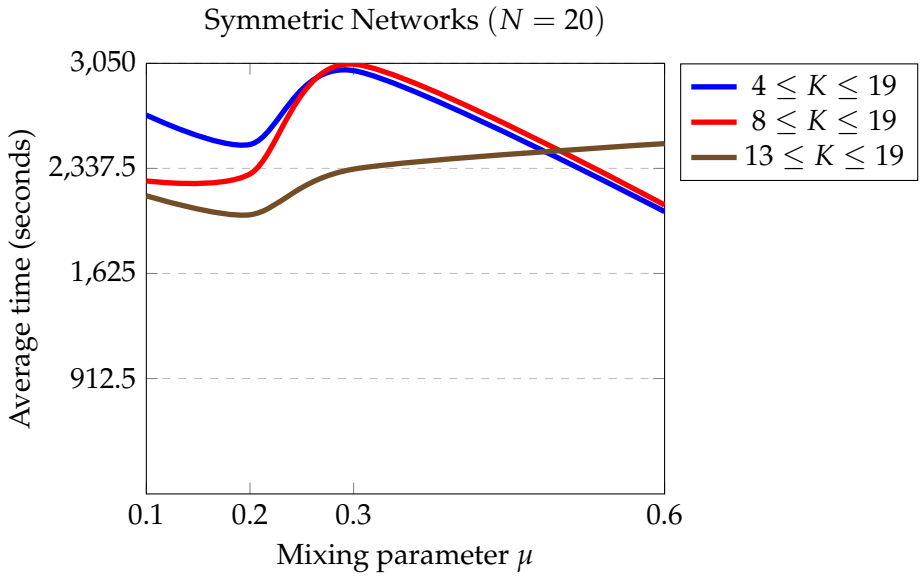
N = 15 $(4 \geq k \geq 14)$				N = 15 $(6 \geq k \geq 14)$			N = 15 $(10 \geq k \geq 14)$
μ	Average time		μ	Average time		μ	Average time
0.1	224,065		0.1	163,597		0.1	133,843
0.2	230,532		0.2	240,340		0.2	137,36
0.3	235,35		0.3	347,256		0.3	152,462
0.6	291,206		0.6	352,239		0.6	177,044
N = 20 $(4 \geq k \geq 19)$				N = 20 $(8 \geq k \geq 19)$			N = 20 $(13 \geq k \geq 19)$
μ	Average time		μ	Average time		μ	Average time
0.1	2699,465		0.1	2252,961		0.1	2152,047
0.2	2500,000		0.2	2300,025		0.2	2023,062
0.3	3000,583		0.3	3044,687		0.3	2332,837
0.6	2044,532		0.6	2089,571		0.6	2505,503

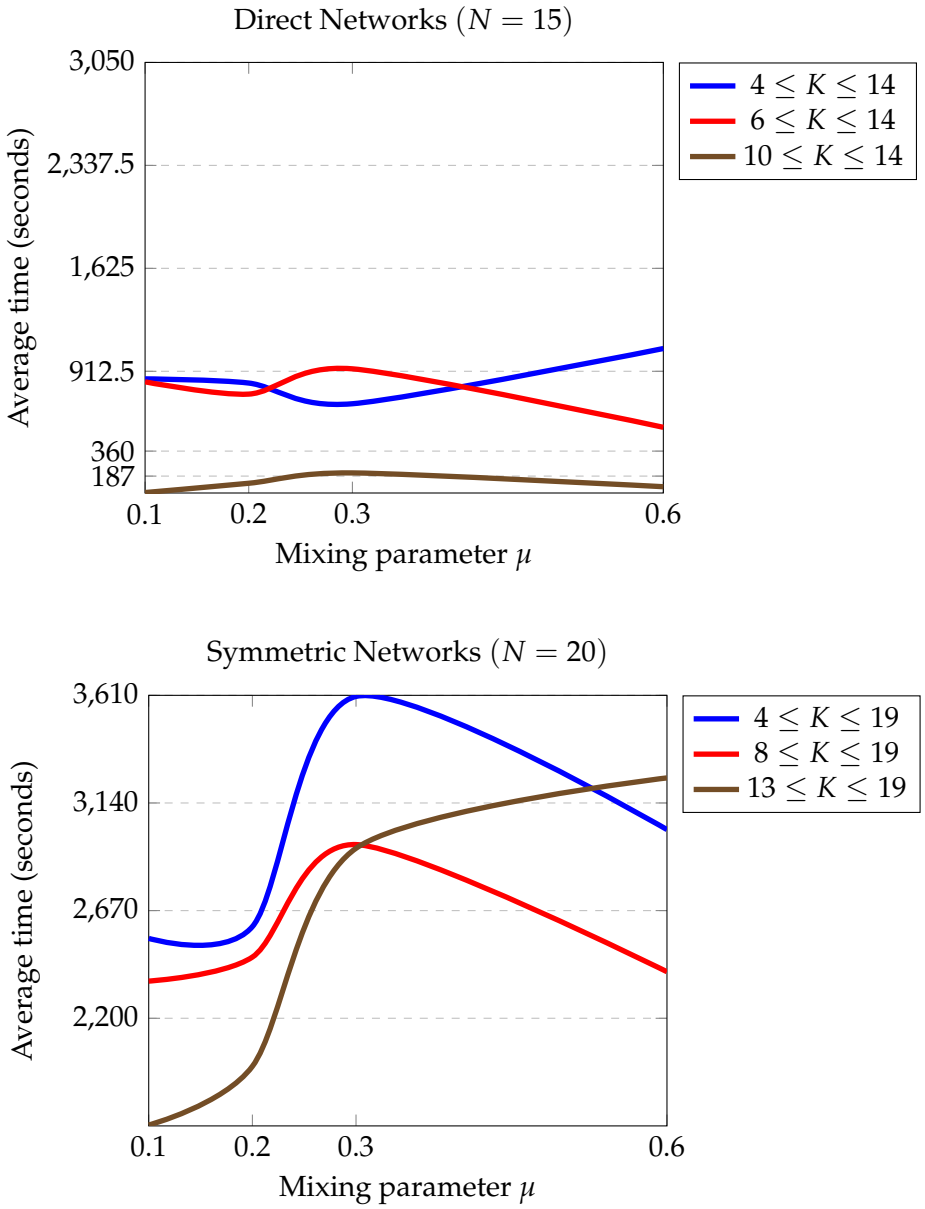
Table 3.1: *Symmetric Networks*

N = 15 $(4 \geq k \geq 14)$				N = 15 $(6 \geq k \geq 14)$			N = 15 $(10 \geq k \geq 14)$
μ	Average Time		μ	Average Time		μ	Average Time
0.1	859,904		0.1	839,556		0.1	73,128
0.2	830,98		0.2	754,43		0.2	137,36
0.3	687,713		0.3	929,399		0.3	208,984
0.6	1069,941		0.6	524,647		0.6	113,48
N = 20 $(4 \geq k \geq 19)$				N = 20 $(8 \geq k \geq 19)$			N = 20 $(13 \geq k \geq 19)$
μ	Average Time		μ	Average Time		μ	Average Time
0.1	2548,841		0.1	2362,241		0.1	1733,811
0.2	2600,750		0.2	2467,056		0.2	1990,017
0.3	3605,17		0.3	2958,363		0.3	2942,173
0.6	3023,843		0.6	2403,17		0.6	3249,491

Table 3.2: *Directed Networks*


Symmetric Networks ($N = 15$)
Average time (seconds)
360
302.5
245
187.5
0.1
0.2
0.3
0.6
Mixing parameter μ
$4 \leq K \leq 14$
$6 \leq K \leq 14$
$10 \leq K \leq 14$


Symmetric Networks ($N = 20$)
Average time (seconds)
3,050
2,337.5
1,625
912.5
0.1
0.2
0.3
0.6
Mixing parameter μ
$4 \leq K \leq 19$
$8 \leq K \leq 19$
$13 \leq K \leq 19$



3.7 Concluding Remarks

The model presented in this chapter includes 89 families of constraints and a huge number of variables, even for small graphs. Nevertheless, we have implemented a lighter (and better but more complicated to expose) version of this model: it turns out that the "basic" variables x_{ik} artificially induce symmetries. In order to eliminate these artificial symmetries and to reduce the number of x and y variables, we just defined x_{ik} for $k = 0, ..., \min(i, K)$. We also eliminated the variables which are necessary equals ($y_{ijkl} = y_{jilk}$). As a consequence, the expressions of the above constraints are slightly

modified.

Furthermore, we (still partially) developed a preprocessing step to:

- take advantage of the symmetry of the problem when it has to. If the graph is undirected, then, $F^b_{kl} = F^{b'}_{lk}$ if b and b' are symmetric ideal blocks.
- avoid generating "dominated" blocks: for instance, if the graph has no loop (i.e. if $s_{ii} = 0 \forall i$) then the Complete Block with diagonal of 0 can only give the same deviation than the Complete Block without diagonal, and both of them, will be better than the Complete block with diagonal of 1. It is then sufficient to consider only one of these 3 ideal blocks.

Chapter 4

VNS for Extended Generalized Blockmodeling

In this chapter, we will present a non-exact approach to generalized extended block modeling, followed by six sections. Section 1 presents a brief description of the limitations of the generalized block modeling problem and ILP formulation. Section 2, is devoted to the overview of relevant literature in the field. Then, in section 3, we reviewed the generalized extended formulation problem in order to facilitate the description of section 4, where a new approach is presented to the extended generalized blockmodeling problem based on the VNS algorithm. The chapter finishes with a computational evaluation of the proposed algorithm by comparing it to other relevant works, followed by concluding remarks.

4.1 Introduction

Intuitive clustering consists of discovering natural groups (clusters) of similar elements in data set. The importance of uncovering network sub-structures within a network is historically well grounded. Such efforts are justified by a relevant number of scientific fields with pratical applications, such as medicine (vanMechelen et al., 2004), biological science (Madeiraa nd Oliveira, 2004; Preli c et al.,2006; van Uitert et al., 2008), psychology (Schepers and van Mechelen, 2011), political science (Doreian et al., 2013; Mische and Pattison, 2000), computer science (Protti et al.,2009), industrial engineering (Selim et al., 1998), and organizational science (Davist al., 1941; Galaskiewicz, 1985), as well as the general literature on classification (Mirkin et al., 1995; van Rosmalen et al., 2009; Wilderjans et al., 2013).

The extended generalized blockmodeling formulation belongs to a class of discrete optimization problems with a finite but often also large solution space. Moreover, the generalization embraces many variants of the problem, such as graph editing, clique partitioning problem and so on which is known to be NP-hard (Protti et al., 2009).

Naturally, ILP methods can provide optimal solutions, but, on the other hand, it requires more computing time. Some exact formulations has been designed for blockmodeling problems: Brusco et al. (2013); Brusco and Steinley, 2009), these are restricted to problems of size 25×25 and smaller. In contrast, an approximate procedure designed by Doreian et al. (2004, 2005) that often find a globally optimal blockmodels for modest sizes of networks indicating to be scalable for larger networks.

Regarding non-exact methods, the VNS algorithm has been indicated in many different studies as a good alternative to tackle blockmodeling clustering problems, usually with efficient performance.

According Chan et al. (2014), the current state-of-the-art algorithm (Doreian et al. 2005), does not scale beyond networks of 100 vertices, generalised blockmodelling could only be applied to a limited number of small networks. In addition, even though there are density-based blockmodelling approaches that scale beyond 100 vertices (Xing et al.2010), the blockmodelling problems which they solve do not have nor consider block types. Therefore, these existing approaches can not be used to solve the extended generalized block modeling approach, since it requires prior knowledge of block types in order to optimize block types.

In this chapter we present a VNS heuristic approach designed to extended generalized blockmodeling class of problems, capable to deal with instances with 300 vertices.

4.2 Literature overview

The increasing interest in social network analyses, in the last few years, combined with the limitations imposed by direct methods, has led to a variety of research efforts devoted to the development of new approaches to tackle block-modeling problems through non-exact methods.

Most blockmodeling applications focus on problems where the purpose is to group a set of objects. Among the first methods used to accomplish such task are Ward's hierarchical grouping method (1963) and p-median clustering (Mulvey and Crowder, 1979). Another common alternative is to apply some variant of the K-means partitioning algorithm (Forgy, 1965; Hartigan and Wong, 1979; MacQueen, 1967), that focuses on the minimization of the sum of the squared Euclidean distance of each object from its cluster centroid.

According to Dorean et al. (2005) the main limitations with those first studies, is that they use the data indirectly to cluster the objects based on some distance measures, as discussed in chapter 2, section indirect methods.

Alternatively Doreian et al. (2004, 2005) propose a relocation heuristic for blockmodeling based on structural equivalence, that deals directly with the data. The algorithm starts with a random partition and improves it using basically two operations: by transferring objects from one cluster to another and exchanging the cluster memberships of objects in different clusters. The relocation algorithm manages to converge to a

blockmodel that is locally optimal in the sense that there is no transfer or exchange that will reduce the number of inconsistencies with ideal block structure. The algorithm is restarted multiple times, to generate a large number of random initial partitions, increasing then the chances of finding the global optimum (or, at least, avoiding a poor local optimum).

A specific study dedicated to generalized blockmodeling is presented by Brusco and Steinley (2007) [11]. In this paper they focus on generalized block modeling using a VNS method to cluster a two-mode binary matrix. The two-mode clustering problems arise when it is necessary to cluster two disjointed sets of entities of a matrix. As examples that illustrate such problems, we can mention: clustering a group of women and social events attended (Davis, Gardner, and Gardner, 1941), clustering Supreme Court justices and a collection of judicial cases (Doreian et al., 2004), and clustering government organizations and political projects relevant to the organizations (Mische and Pattison, 2000).

Brusco and Steinley (2007) [11] ended the study concluding their algorithm performs extremely well, but the value of the method is diminished by the assumption that block placements are known. In addition, according to the authors, their approach has two main limitations: the first one is associated with the method and analyses reported in the paper. All simulation and analyses are restricted to structural equivalence. Second, is the assumption that the number of clusters for both row and column objects was known. As mention by Brusco and Steinley (2007) [11], the determination of the appropriate number of clusters remains a challenging issue in one-mode cluster analysis, and is even more complex in two-mode clustering.

Later on, Brusco et al. (2013) publish a VNS method for a two-mode block-modeling problem in social network analysis, where they do a extensive comparative study using artificial and real networks, the VNS heuristic proposed outperformed relocation heuristic (RH) and tabu search (TS) method for all problems. Again, the technique is based in the assumption that block placements are fixed, as the size of partition as well. Overall, their results suggest that the VNS heuristic is a promising approach.

Another interesting study is by Chan et al. (2014) [32] using evolutionary computing. They present two new algorithms: first, a genetic algorithm based approach and the second, a simulated annealing based one. Their approach permits partial or complete pre-specification of block models, allowing confirmatory and exploratory types of analysis to be performed. Although for all analyzed experiments the image matrices are pre-specified. Genetic algorithms are known to be able to tackle hard combinatorial optimisation problems. However the success of any (GA) approach really depends on the design of its search operators and appropriate integration. Chan et al.(2014) had developed a efficient approach, faster than existing ones and capable to deal with medium-sized real world networks.

In this chapter we described our VNS approach to tackle the extend generalized blockmodeling designed for medium-sized real-world networks that do not require pre-specified information.

4.3 Formulation

To facilitate the description of the VNS approach proposed in this chapter, let us recall some basic notations and definition of extended generalized blockmodeling problem.

1. A network is a set of objects NxN which can be defined as a graph $G = (V, R)$, where V is a set of nodes and R a set of edges.
2. The set of edges are represented by the binary matrix $S : (v_i, v_j) \in R \Leftrightarrow S_{ij} = 1$, if vertex v_i is connected to vertex v_j, and 0 otherwise.
3. A cluster c is a set of nodes of $V = \{v_1, \ldots, v_n\}$ that share structural characteristics in terms of relationship B. Let's consider C as the clustering which divides the relation R into blocks: $R(c_k, c_l) = R \cap c_k \times c_l$.
4. If S is the adjacency matrix of G. The block (k, l) associated to the partitions C_k and C_l is a sub-matrix of S, where the line indexes are in C_k, and the column indexes in C_l.
5. An ideal block B is type of graph and can be see as a matrix, used to define the desired relationship between the clusters c_i and c_j.
6. Let $\mathcal{F}_{ij}^{B}$ be clustering deviation measure, which expresses error between the block (k, l) and an ideal block $B \in \beta(c_i, c_j)$, and e_{ij} be the minimal error for the block $R(c_i, c_j)$:
$$e_{ij} = \min_{B \in \beta(c_i, c_j)} \mathcal{F}_{ij}^{B}.$$
7. The global error is therefore:
$$P(C) = \sum_{c_i, c_j \in C} e_{ij}$$

Definition 4.3.1. *The extended generalized blockmodeling is defined as problem of finding the partition size and the set of ideal blocks that reduces the global error. The extended generalized blockmodeling is the following combinatinal optimization problem:*

$$\min_{C \in \mathbf{C}} P(C)$$

4.4 The variable neighborhood search algorithm

Metaheuristics are a general framework to build heuristics. The Variable Neighborhood Research (VNS) is a metaheuristic framework to solve combinatorial optimization problems, which explores a solution space by performing iteratively neighborhood changes.

The VNS was first introduced by Mladenović and Hansen in 1997, it has been suggested as an effective methodology to deal with a variety class of blockmodeling problems by different authors.

The VNS consists basicaly of a stochastic component, shaking phase, and a deterministic component, the local search. The shaking randomized selection of a neighbor, the local search compare the solution obtained with the incumbent one and will be accepted as a new starting point, if an improvement was made, otherwise it will be rejected. The iterative process is repeated until the stopping criteria are met.

Let $N = \{N^{(1)}, N^{(2)}, \ldots, N^{(k)}\}$ denote a finite set of neighborhoods, where $N^k(s)$ is the set of solutions in the k^{th} neighborhood of solution s, the basic steps of the algorithm are announced as follows:

1. Initialization:
    - Set the neighborhood structure N^k
    - Determine an initial solutions s
    - Set $k = 1$
2. Repeat:
    - Shaking: Generate a random solutions s' in $N^k(s)$
    - Local search: Apply some local search method with s' as initial solution, let s'' be the new local optimum obtained.
    - Move or not: If s' is better than s the incumbents, then sets $s = s'$ and continue the search with $k = 1$ else $k = k + 1$

4.4.1 Initialization

The initial solution can be created in many ways. Hansen and Mladenović (2002) in [25], present a scheme for a large class of problems, in order to obtain an efficient VNS implementation. They say, based in their experiments, that VNS results depend very little on the chosen initialization rule. Their suggestion is: the simplest rule is thus best one. They also states that might be such problems which this approach is not extended.

Happens that many VNS researches conducted on the blockmodeling problem shows that the selection of a good initial solution is crucial for avoiding convergence to poor local maxima.

Some implementations of the blockmodeling, such as Brusco and Steinley (2007)[11], use purely random assignment of objects to clusters, to establish initial partition. This approach can guarantees a good coverage of the search space, but for large networks it require many iterations until a local optimum is reached.

Another common approach used on the blockmodeling problem, consists of constructing the initial partition using some degree of randomness, by fixing few elements into some clusters and let the rest of the elements be assigned randomly. The draw back of this approach is that, for each iteration, the partition may not change that much, increasing the probability of getting stuck in a local minima.

Knowing that the initial solution plays an important role, by helping to decrease the time an heuristic need to achieve an acceptable outcome, we design a initialization process which measures distance between candidate solutions.

The initialize procedure randomly generates all possible partition sizes, then calculates the deviation value $P(C)$ for each partition, choosing as first incumbent the candidate solution which holds the smallest value of $P(C)$. P holds the size of the candidate solutions sorted in descending order with respect to deviation measure, P will be used in the shaking phase.

Algorithm 1

```
1:  function INITIALISE( P)  ▷ σ(n) is a permutation randomly generated and n = |V|
2:      i ← n
3:      t ← σ(i)
4:      C its a random partition with t clusters
5:      while i ≥ 1 do
6:          if (P(C) = 0) then
7:              return C
8:          end if
9:          t ← σ(i − 1)
10:         C* its a random partition with t clusters
11:         if P(C) > P(C*) then
12:             C ← C*
13:             j ← |P|
14:             P_{j+1} ← t
15:         end if
16:         i ← i − 1
17:     end while
18:     return C
19: end function
```

In table 4.1, we enumerate execution steps of initialize procedure for the graph G wich has as set of nodes $V = \{a, b, c, d, e\}$.

i	$\sigma(i)$	random partition $C_{\sigma(i)}$	$P(C)$	$P(C^*)$	$P = \{\emptyset\}$
1	$\sigma(1) = 3$	$C_{\sigma(1)} = \{a, b\}\{c, d\}\{e\}$	20	-	$P = \{3\}$
2	$\sigma(2) = 2$	$C_{\sigma(2)} = \{a, d\}\{b, c, e\}$	18	20	$P = \{2, 3\}$
3	$\sigma(3) = 5$	$C_{\sigma(3)} = \{a\}\{b\}\{c\}\{d\}\{e\}$	19	18	$P = \{2, 3\}$
4	$\sigma(4) = 1$	$C_{\sigma(4)} = \{a, b, c, d, e\}$	22	18	$P = \{2, 3\}$
5	$\sigma(5) = 4$	$C_{\sigma(5)} = \{a\}\{b, c\}\{d\}\{e\}$	16	22	$P = \{4, 2, 3\}$

Table 4.1: *initialize execution*

4.4.2 Shaking

Shaking is the process of skipping the current solution to a random neighborhood element, using or not some sort of strategy to generate random perturbations. It allows the algorithm to randomly explore the solution space, preventing the process from getting stuck in minimal locations. A shake generally does not improve overall quality of the process, but it provides a different point of view in the search space. Our proposed VNS algorithm uses two shaking neighborhood strategies:

- **Shaking cluster elements:** This strategy generates minor perturbations in the current solution. Algorithm 2 presents the function SHAKECELEMENTS, the idea here is to add a new empty cluster, and randomly choose a node and move to the empty cluster.
- **Shaking size of partition:** This approach significantly moves away from the current solution, it's also the component which adds more diversification in the search space; such statregy is also known in the literature as intensified shaking.

  The procedure chooses a partition size in P to generate a completely new random solution, jumping to a complete new neighborhood.

  Due to this diversification, the procedure can expand the search to unexplored regions in the solution space, this expansion consists of visiting solutions that have not been examined previously, which does not guarantee any improvement in the value of the objective function, only generates another candidate entry.

  Algorithm 3 presents the function SHAKEPSIZE which is done in very simple manner as most VNS-based heuristics.

Algorithm 2

1: **function** SHAKECELEMENTS(C) ▷ $\lambda(C')$Returns the number of clusters in a partition C'
2: $k \leftarrow \lambda(C)$
3: $C' \leftarrow C$
4: add the empty cluster c_{k+1} in C'
5: for the partition C'move a node from c_k to c_{k+1}
6: **return** C'
7: **end function**

Algorithm 3

1: **function** SHAKEPSIZE(P)
2: $k \leftarrow \sigma(P)$
3: generate randomly the partition C' with k clusters
4: **return** C'
5: **end function**

Figure 4.1 shows the comparison of shake strategies. The shaking cluster element

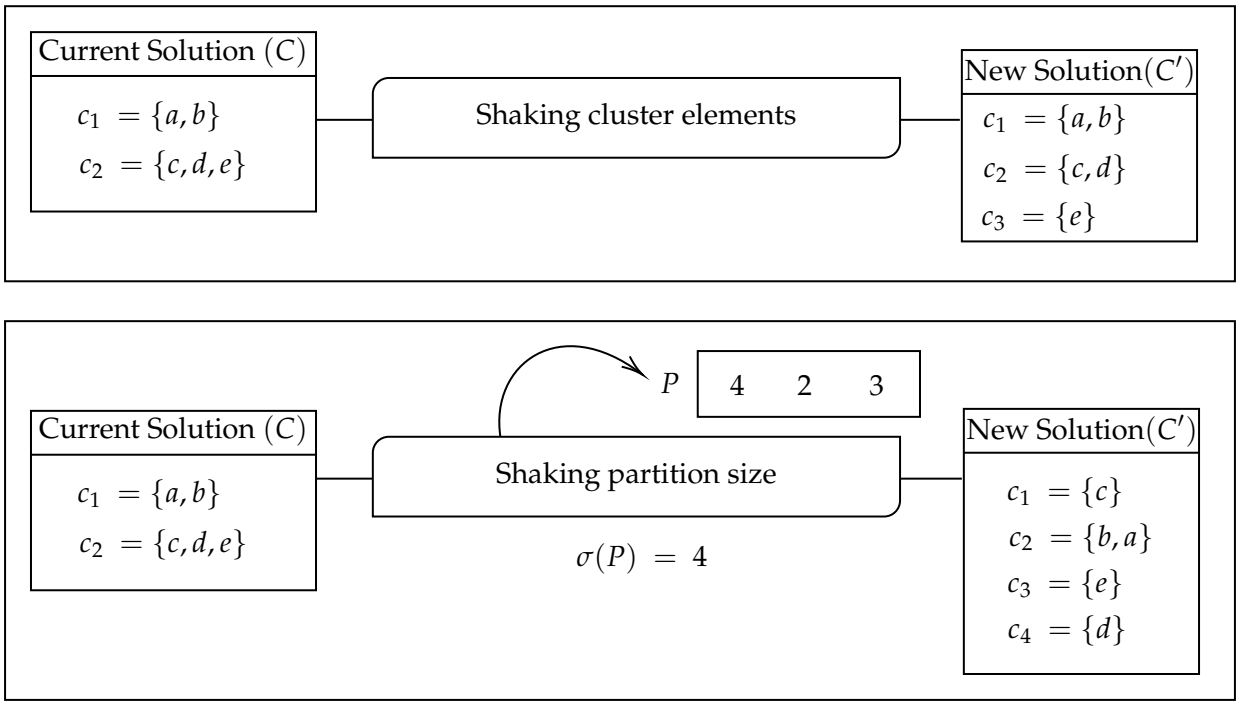


Figure 4.1: *Shaking mechanism schema*

strategy generates a new C' solution based on C, so there will always be fixed factor. We can also notice in the figure 4.1, that in the second approach, when compared to the first one, has as only predefined component, the partition size, and it generates a completely new random solution C'.

4.4.3 Local Search

Local search is based on the concept of a neighborhood. Identifying an appropriate neighborhood improvement methodology depends on the type of problem and overall strategy. This is an important step in designing a VNS heuristic, as it defines local search performance and the quality of solution.

The most popular neighborhood improvement methods are: first improvement and best improvement. Hansen and Mladenović suggest, if the initial solution is chosen randomly, the first improvement may be most appropriate, but when using some constructive heuristic, best improvement rule might be the best.

Algorithm 4 presents the function LOCALSEARCH. Once having defined the initial solution, the local search perform systematically MOVES and SWAPS on the current solutions, until no further moves and swaps are allowed.

Algorithm 4

```
1: function LocalSearch(C)
2:     C ← Move(C)
3:     C ← Swap(C)
4:     return C
5: end function
```

The most traditional approach for MOVE operations, is to take an element u from a cluster c_i and places it into the cluster c_j. In our implementation, while moving elements, the function can add and remove clusters from the partition.

Algorithm 5

```
1: function Move(C)                          ▷ Let C be an initial partition
2:     Let C' be the new clustering
3:     Let v be a vector with n random elements, where n = |V|
4:     for all u ∈ V do
5:         for all (c_j ∈ C) and (u ∉ c_j) do
6:             move u in c_j
7:             Let C* be the new clustering
8:             if P(C*) < P(C) then
9:                 C' = C*
10:                remove all empty cluster in C
11:            end if
12:        end for
13:        if λ(C) < N then        ▷ λ(C)Returns the number of clusters in a partition C
14:            Add empty cluster c_j to C
15:            move u in c_j
16:            Let C* be the new clustering
17:            if P(C*) < P(C) then
18:                C' = C*
19:            end if
20:        end if
21:    end for
22:    return C'
23: end function
```

The second one is swap, it interchanges elements between clusters. Let $v \in c_i$ and $w \in c_j$, the swap procedure takes the element v and place it in c_j and element w and places into the cluster c_i, if the operation improves the value of $P(C)$ then the new incumbent will be accepted.

Algorithm 6

```
1: function SWAP(C)                          ▷ Let C be an initial partition with k clusters
2:     Choose randomly a node v ∈ V
3:     Let i be the cluster index of v in C
4:     for all (c_j, c_l) ∈ C do
5:         for all pairs of nodes (v, w) ∈ c_j c_l do
6:             Swap v and w
7:             Let C^ij be the new clustering
8:             Let C* = min_{B∈β(c_i,c_j)} F_ij^B
9:             if P(C) > P(C*) then
10:                C = C*
11:            end if
12:        end for
13:    end for
14:    return C
15: end function
```

The idea behind the proposed approach, relies on the fact that the local minima, with respect to one or several neighborhoods, are relatively close to each other. We use the best improvement strategy while moving elements between clusters and first improvement when performing swaps.

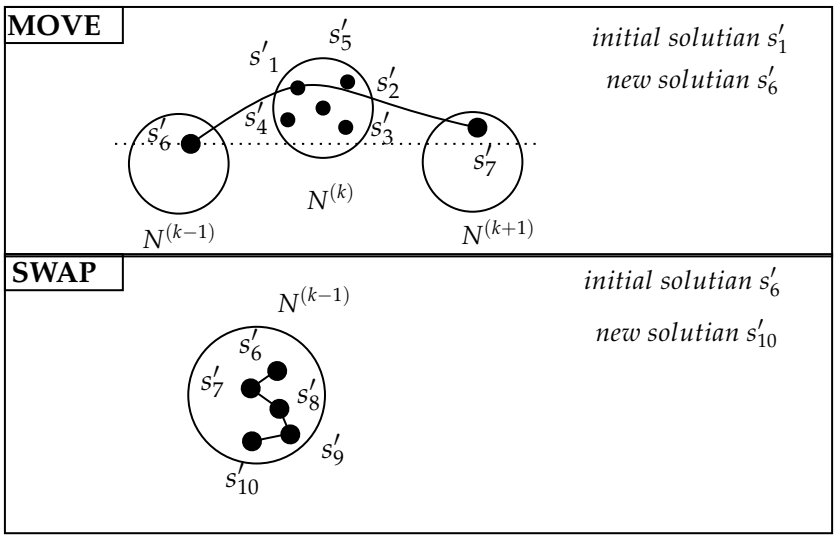


Figure 4.2: *Local seach strategy*

Figure 4.2 displays the local search approach, using moves and swaps, as neighborhood operation strategy. The move operation, starts with a pre-defined neighbour s'_1, performing a descent in the neighborhood $N^{(k)}$, exploring all existent neighbours, and also close neighbour in the neighborhood $N^{(k+1)}$ and $N^{(k-1)}$. The new incumbent is going to be set as the one which has the smallest value among all visited solutions, s'_6.

The swap operation receives as input the the best solution found while perfoming moves, in the figure s'_6, and it will start perfoming exchanges between clusters, changing the current solution every time a better solution is found.

Figure 4.3 illustrate the basic idea behind move and swap operations, while move operates on structurally different neighborhoods $N^{(1)}, \ldots, N^{(k)}$, swap typically operates at one type of neighborhood with variable depth, for a neighborhood $N^{(k)}$ it changes k variables.

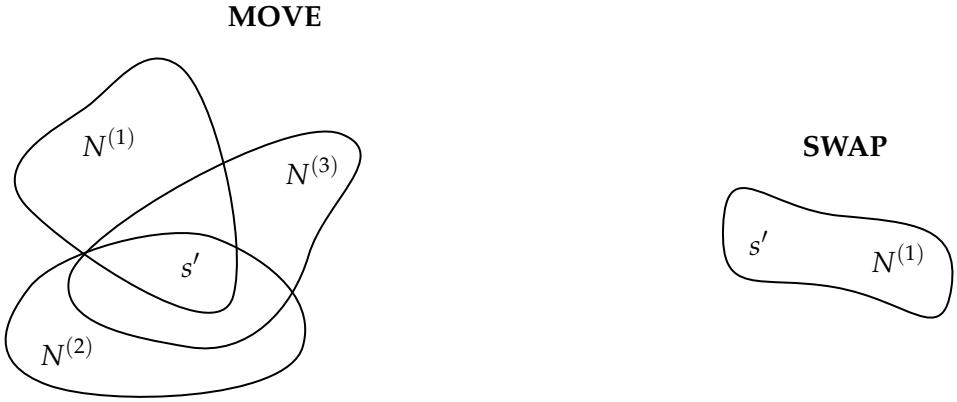


Figure 4.3: *Illustration of the neighborhoods used by move and swap*

4.4.4 VNS for the extended generalized blockmodeling

Algorithm 7 presents the VNS function and in the previous sections we have explained in details the the functions embedded in the general procedure.

Let a set N^k denotes the set of neighborhood structures $(k = 1, \ldots, k_{max})$, the method begins by calling INITIALISE function, which generates k_{max} random solutions, where the best is selected to be $C^{current}$, as given in algorithm 1. If $P(C^{current}) = 0$, the algorithm stops. Otherwise, local search is performed.

The ILS, begins with move operations, making a descent in the neighborhood N^k, then it explores a close solution in neighborhoods N^{k+1} andN^{k-1}, by adding and removing a cluster in a move operation.

The local search finish by exchange nodes between cluster in a given neighborhood, the swap phase as display in algorithm 6. The best solution will be replaced by the new one if the solution is improved.

After, depending on the size of the current solution, shake elements or shakePsize will be executed. The first one introduces a new neighborhood structure. The second jumps to another neighborhood among the best ones kept in P.

The VNS loop is performed until the running time $t_{elapsed}$ reaches t_max or the number of visited neighborhood k_{cur} reaches k_{max}.

Algorithm 7

function VNS-EGBM(t_{max}, k_{max})
  $C^{current} \leftarrow$ INITIALISE(P)
  $C^{best} \leftarrow C^{current}$
  $t_{elapsed} = 0$
  while $(P(C^{current}) > 0$ and $t_{elapsed} \leq t_{max}$ and $k_{cur} \leq k_{max})$ **do**
    $C^{current} \leftarrow$ LOCALSEARCH($C^{current}$)
    if $P(C^{best}) > P(C^{current})$ **then**
      $C^{best} \leftarrow C^{current}$
    end if
    if $(\lambda(C^{current}) \neq n)$ **then** ▷ λReturns the number of clusters of partition
      $C^{current} \leftarrow$ SHAKECELEMENTS($C^{current}$)
    else
      $C^{current} \leftarrow$ SHAKEPSIZE(P)
    end if
    $k_{current} \leftarrow k_{current} + 1$
    $t_{elapsed} \leftarrow t_{elapsed} + 1$
  end while
  return C^*
end function

4.5 Computational Experiment

In this section, we first analyze the efficiency of the proposed method and then the quality of the solutions found. To this end, we use artificial datasets, generated with the LFR benchmark, to evaluate the scalability and optimisation. Then real datasets are used to evaluate the speed and optimization of our VNS approach, regarding to the evolutionary computation method proposed by Chan et al. (2014), since their numerical results outperforms other studies.

4.5.1 Artificial datasets and experimental settings

To test scalability, we generated a family of networks with increasing sizes. We evaluated our VNS approach in a population of synthetic network with sizes of 50, 100, 200, 300, 400, and 500. For each size, we generated ten different instances with the LFR algorithm, in order to test efficiency, once LFR generates unspecified communities based

on the following settings: μ mix parameter, which indicates the amount of connections between communities, the average node degree k_{min}, and the maximum node degree set to k_{max}.

To generate the six synthetic networks, the following parameter settings were used: $\mu = 0.3$, $k_{min} = 6\%$ of network size, and $k_{max} = 10\%$ of network size. Based on the previous configuration, the LFR program produces three files:

1. The network file contains the list of edges;
2. A community file contains a list of the nodes and their membership;
3. The last one, is named statistics, is the file that contains the degree distribution (in logarithmic bins), the community size distribution, and the distribution of the mixing parameter;

The network file is the input to the heuristic, the second file: community along with statistics, are used to evaluate and analyze the quality of the heuristic solution. It turns out that this configuration set produces charts with nontrivial topological features; This type of topology often occurs in graphs that model real systems. These networks are also known as complex networks, and finding topological resources that can differentiate a community and an intercommunity relationship is a very challenging task [14], [44]

Figure 4.4 display the correspondence between nodes and communities, and also community statitics generated by LFR algorithm for a network with 50 nodes, named instance 5. Figure 4.9, shows the network communities distribution generated by LFR highlighted in different colors.

Since LFR algorithm produces random topology features for the built communities, in the first conducted experiment, one hundred percent of the results, VNS found a partition that differs from the partition generated by the community builder.

To illustrate this scenario, let's analyse the results of instance 5, after running VNS heuristic in extended generalized mode, that is, without imposing restrictions on position, block type, or size, it finds $P(C) = 0$.

Based only on the pattern of links, the network is divided into two commnunities: where the ties inside $C1$ has a row-regular block pattern, and pattern of links in $C2$ are said column-regular, as display in figure 4.7. The relantionship inter-community are also following the same pattern, and depending on reading orientation, it can be classified as row-regular or column-regular.

This topological features shows a orientation in the relationship intra and inter community, such information can be useful, knowing that every type of block has a meaning, this specific one shows that's possible to split the graph following the relationship orientation. Such topological configuration may be hand when studing the flow of spreading information. Table 4.2 shows that running the extended generalized mode in population of networks with increasing sizes, it finds $P(C) = 0$ in a short time.

This happens because the extended generalized mode is very flexible, allowing the

	50	100	200	300	400	500
deviation measure	0	0	0	0	0	0
time (seconds)	0	6	92	229	58	968

Table 4.2: *VNS heuristic in extended generalized mode*

heuristic to quickly converge to a local minimum. Although the extended generalized blockmodeling has more block types than the conventional one, it will also have a larger set of multiple solutions, helping to converge $P(C) = 0$.

Now let's look at the built-in community and topology features of instance 5. If we calculate the block modeling fit for the LFR partition, we can see that for this partition there is no ideal block set, at least defined in the literature, which can perfectly fit the generated community. Figure 4.5 shows the closest ideal blockmodel pattern, the deviation measure and the communities. In table 4.6, display the permuted and blocked relational matrix.

Going even further in the analysis of network instance 5, we run the VNS, this time fixing the number of communities to twelve, same generated by LFR, the partition found has another community distribution that reduces even more $P(C)$. Figures 4.11 and 4.12 display the new community arrangement.

To better assess scalability, using the population mentioned before, as second experiment, we reduce the set of block types β to complete blocks placed in the diagonal and null blocks off-diagonal. Knowing that among the population used in the experiments, there is no network that can meet the topological constraints and generate partition with $P(C) = 0$.

Even decreasing the number of ideal block to two, such change increased substantially the running time and objective cost. This behaviour is complete understandable and also expected because now we solving the graph partion problem, which falls under the category of NP-hard problem, using a graph which does not have such connections pattern.

	50	100	200	300	400	500
deviation measure	16	38	108	187	248	-
time (seconds)	55	1136	25598	148856	504515	-

Table 4.3: *Synthetic Communities*

```
Node  Group         in-degree distribution
1      8            1   0.22907
2      3            2   1.92531
3      4            3   0.305249
4      10           4   0.158027
5      2            5   0.0916281
6      2
7      4            in-degree distribution
8      8            1   0.04
9      5            2   0.64
10     11           3   0.14
11     5            4   0.1
12     9            5   0.08
13     6
14     8
15     9            ---------------------------------------
16     1            out-degree distribution (probability density function of the degree)
17     2            1   0.269001
18     12           2   1.345
19     7            3   1.37609
20     1            4   0.0336251
21     4
22     11            out-degree distribution (degree-occurrences)
23     4            1   0.04
24     7            2   0.4
25     5            3   0.54
26     6            4   0.02
27     2
28     7            ---------------------------------------
29     3            community distribution (size-occurrences)
30     3            3   0.333333
31     3            4   0.166667
32     11           5   0.5
33     12
34     5
35     3            ---------------------------------------
36     10           mixing parameter (in-links)
37     8            0           0.32
38     10           0.2         0.04
39     10           0.25        0.1
40     9            0.333333    0.12
41     12           0.4         0.04
42     1            0.5         0.36
43     6            1           0.02
44     1
45     6            ---------------------------------------
46     8            mixing parameter (out-links)
47     1            0           0.48
48     6            0.333333    0.22
49     2            0.5         0.12
50     5            0.666667    0.12
                    1           0.06
```

Figure 4.4: *Community and statistics file example* $N = 50$

```
DestiatonMesuare = 36
ElapsedTime      = 0 (in seconds)
-----------------------------------------------------
PARTITION
-----------------------------------------------------
C1 = { 16 , 20 , 42 , 44 , 47 }
C2 = { 5 , 6 , 17 , 27 , 49 }
C3 = { 2 , 29 , 30 , 31 , 35 }
C4 = { 3 , 7 , 21 , 23 }
C5 = { 9 , 11 , 25 , 34 , 50 }
C6 = { 13 , 26 , 43 , 45 , 48 }
C7 = { 19 , 24 , 28 }
C8 = { 1 , 8 , 14 , 37 , 46 }
C9 = { 12 , 15 , 40 }
C10 = { 4 , 36 , 38 , 39 }
C11 = { 10 , 22 , 32 }
C12 = { 18 , 33 , 41 }
-----------------------------------------------------
IDEAL BLOCK DEVIATION MEASURE
-----------------------------------------------------
  0   0   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   1   0   0   0   1   1   0
  1   0   0   0   0   0   0   2   0   0   0   1
  1   0   0   0   0   1   1   0   1   0   0   1
  1   0   0   1   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   1   0   1   1   0   1   1   1
  1   1   0   0   0   0   1   0   0   1   0   0
  0   1   0   0   0   0   0   1   0   1   1   0
  0   0   1   0   0   0   0   0   0   0   0   0
  0   0   1   0   1   1   0   1   1   0   0   0
  0   0   0   1   1   0   0   0   0   0   0   0
-----------------------------------------------------
CLOSEST IDEAL BLOCK PATTERN
-----------------------------------------------------
 REG. NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL
 NULL REG. NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL
 NULL NULL REG. NULL NULL NULL NULL NULL NULL NULL NULL NULL
 NULL NULL NULL REG. NULL NULL NULL NULL ROWD NULL NULL NULL
 NULL NULL NULL NULL REG. NULL NULL NULL NULL NULL NULL NULL    REG. -> regular block
 NULL NULL NULL NULL NULL REG. NULL NULL NULL NULL NULL NULL    NULL -> null block
 NULL NULL NULL NULL NULL NULL COLF NULL NULL NULL NULL NULL    ROWD -> row dominant
 NULL NULL NULL NULL NULL NULL NULL REG. NULL NULL NULL NULL    COLF -> column functional
 NULL NULL NULL NULL NULL NULL NULL NULL REG. NULL NULL NULL
 NULL NULL NULL NULL NULL NULL NULL NULL NULL REG. NULL NULL
 NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL COLF NULL
 NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL REG.
-----------------------------------------------------
```

Figure 4.5: *Blockmodel pattern, deviation measure and community distribution*

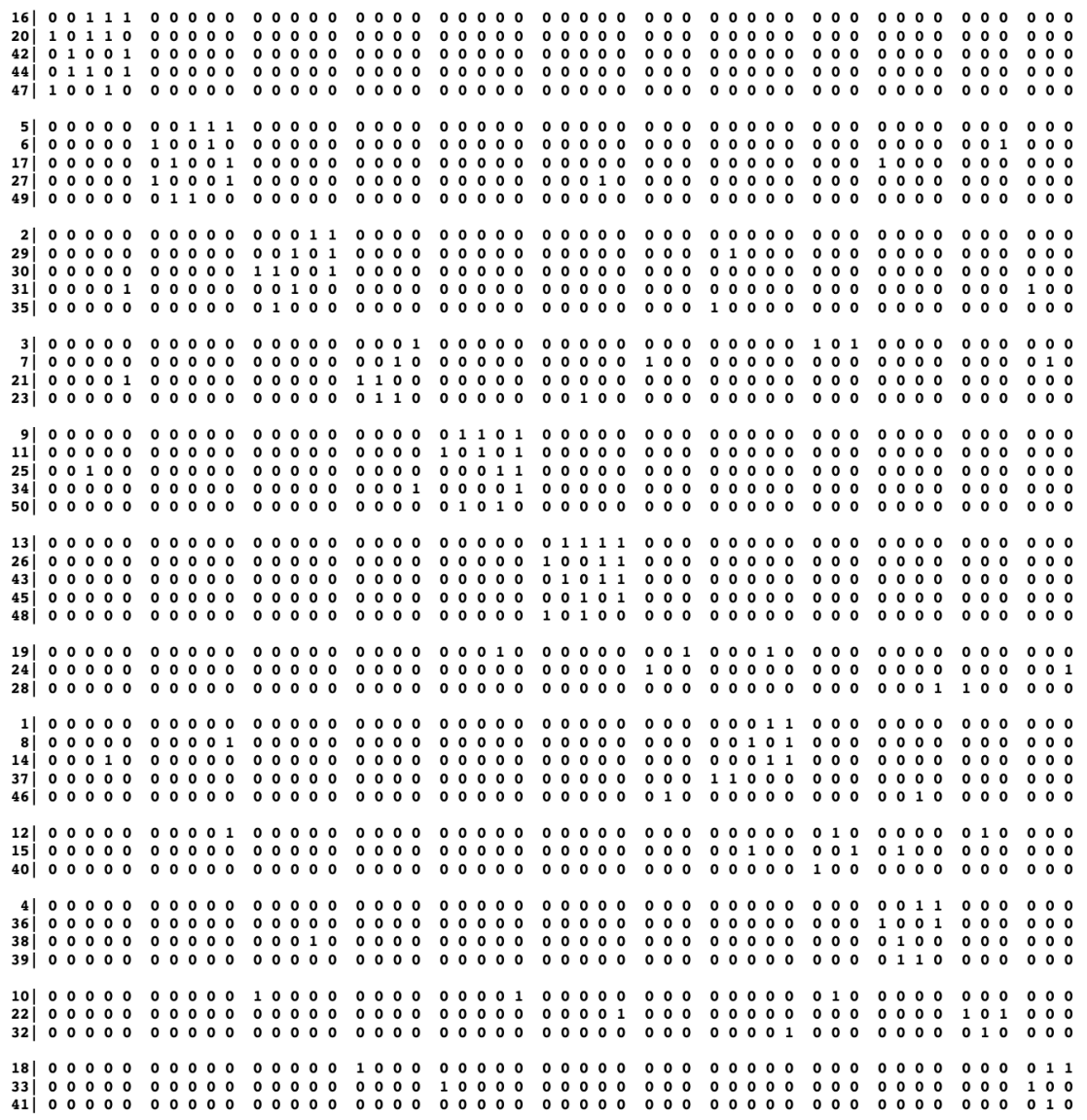


Figure 4.6: *Relational matrix*

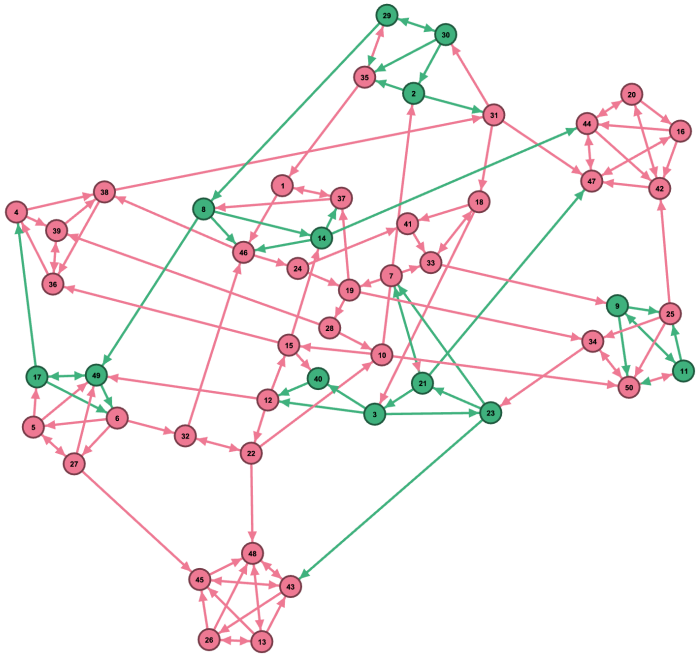


Figure 4.7: *Partition with the smallest value of* $P(C)$

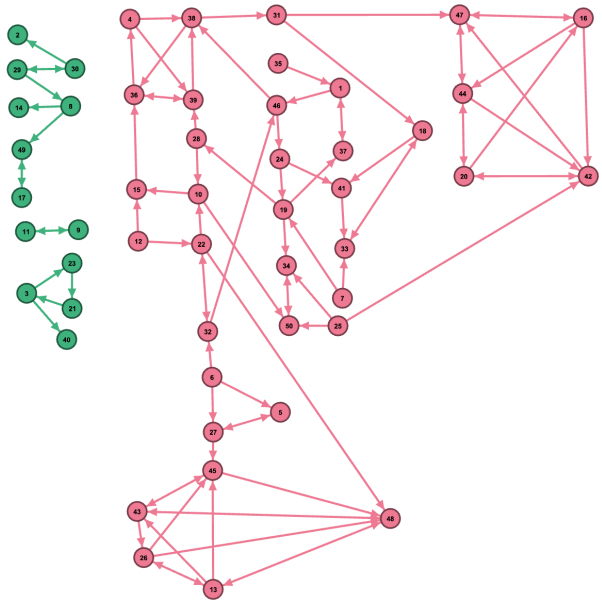


Figure 4.8: *Partition with the smallest value of* $P(C)$

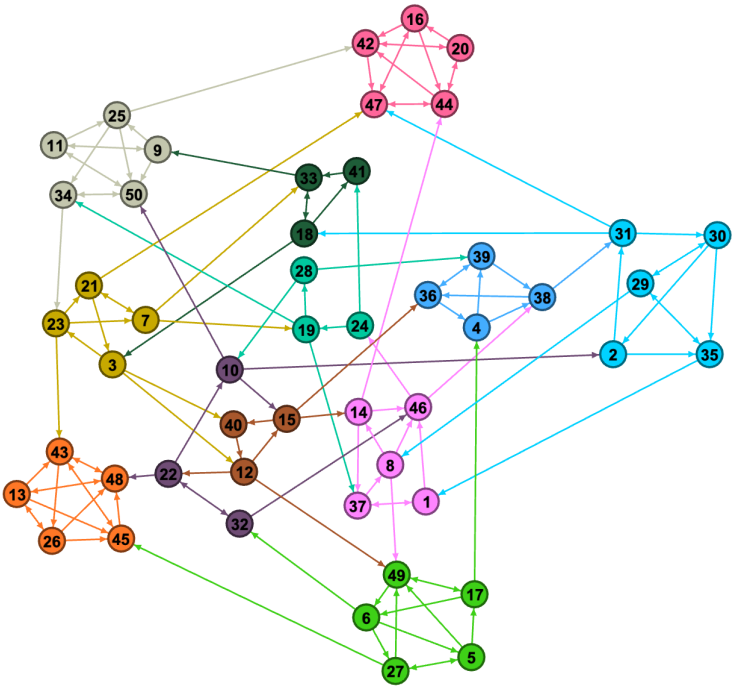


Figure 4.9: *LFR community.*

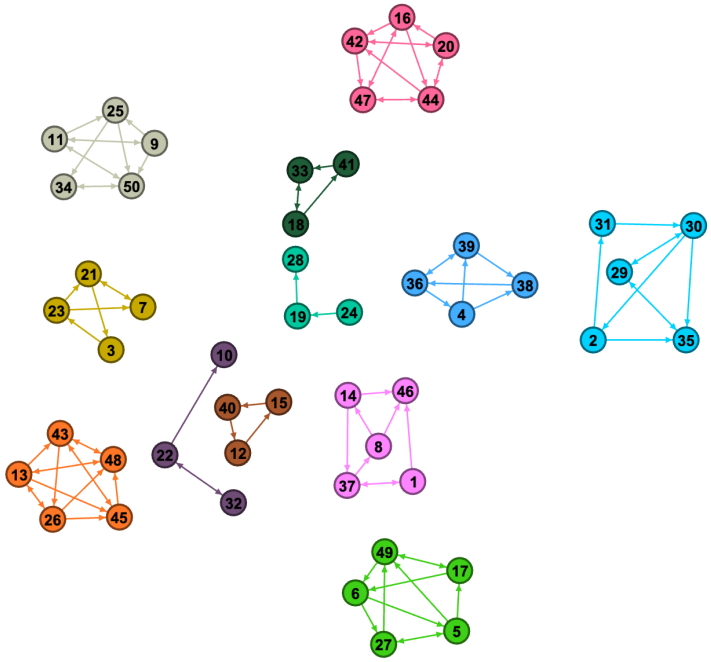


Figure 4.10: *LFR connections within communities*

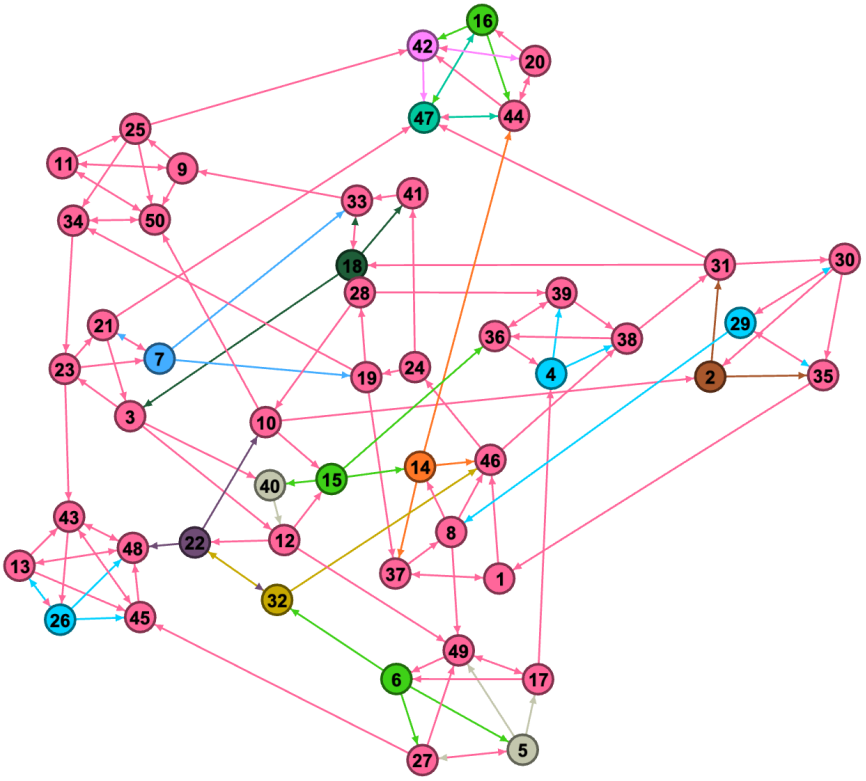


Figure 4.11: *New community distribution*

```
-------------------------------------------------------
DestiatonMesuare = 0
ElapsedTime      = 32 (in seconds)
-------------------------------------------------------
PARTITION
-------------------------------------------------------
C1 = { 14 }
C2 = { 42 }
C3 = { 32 }
C4 = { 7 }
C5 = { 47 }
C6 = { 40 , 5 }
C7 = { 2 }
C8 = { 22 }
C9 = { 18 }
C10 = { 6 , 15 , 16 }
C11 = { 26 , 29 , 4 }
C12 = { 20 , 38 , 28 , 37 , 25 , 43 , 23 , 27 , 13 , 30 , 39 , 33 , 49 ,
        50 , 35 , 31 , 19 , 34 , 24 , 45 , 3  , 44 , 17 , 48 , 1  , 10 ,
        46 , 8 , 36 , 11 , 21 , 9 , 41 , 12 }
```

Figure 4.12: *Network representation*

4.5.2 Benchmark algorithms and real datasets

To evaluate the performance of the proposed VNS heuristic, we use a comparative experiment scenario, where performance reflects a measure of speed and optimization. We compare the results find by our proposed VNS approach with the results in [32], since they tackle the generalized blockmodeling problem for real medium size networks.

The three datasets used in our experiments are Enron's email dataset, a communication network for summer doctoral schools, and an airport route network. The network resources have been kindly provided by Chan Jeffrey.

The first one, Enron email dataset, is the email communication network among Enron employees at three different time periods, and we using only the period t_1, where the number of vertices and edges are 270 and 1202, respectively. The second is doctoral summer school, this network represents the interaction between participants before and during a summer school. The graph consists of 73 vertices and 1,139 edges. The last one is the airport routing network, which represents the flight routes between airports, where each vertex represents an airport, with directed edges representing a flight route. The network consisted of 561 and nd 10,956 edges.

To make a fair performance comparison, we use the same set of parameters as used in [32], where the partition size and set of ideal block are fixed as display in table 4.4. For the airport routing and Enron email dataset the KL-based algorithm takes to long to finish, so results are not reported in table 4.4.

	Doctoral summer school					
	Obj. Cost		Time (in seconds)		Partition size	Block pattern
Algorithm	Mean	SD	Mean	SD	4 x 4	complete regular row-regular column-regular
VNS	102	6	90	2		
GA	859.3	114.1	24.2	4.2		
KL-based	701	0	27,056.3	845.5		
	Enron email dataset					
	Obj. Cost		Time (in seconds)		Partition size	Block pattern
Algorithm	-		Mean	SD	3 x 3	complete regular row-regular column-regular
VNS	789		200	50		
GA	2358		516.14	158.27		
KL-based	N-A		N-A	N-A		
	Airport routing					
	Obj. Cost		Time (in seconds)		Partition size	Block pattern
Algorithm	-		Mean	SD	3 x 3	regular
VNS	0		2500	273.5		
GA	0		1543	334.33		
KL-based	N-A		N-A	N-A		

Table 4.4: *VNS, GA and KL-based, comparison*

4.6 Concluding Remarks

We have evaluated our method on both synthetic and real-world networks. The artificial networks are used to evaluate scalability and quality of solution once the community distribution is known. We also assess the performance of our VNS approach by comparing it to benchmark algorithms such as GA and KL, designed to solve the generalized blockmodeling for mediam sizes real networks with fixed parameters.

We are not able to compare the speed and optimization results of the extended generalized blockmodeling, since the other approaches require knowing the size of the partition and a pre-definition of the ideal models.

Besides that, the experiments conducted in this chapter show that the results found by the heuristic converge to a satisfactory solution, may not be the best of all the solutions to the problem, but it is still valuable because finding it does not require a prohibitively long time and allow the treatement of medium size networks.

As for all networks used in this chapter, there is no ground truth for validation, and the analysis is constraint to numerical results. The validation of blockmodeling fitness, community distribution is conducted in chapter 5, where real case scenario is presented.

All the experiments in this chapter were conducted on AMD 2.6 GHz server with 500 GB of memory and running Debian GNU/Linux 7 and developed using C++.

Chapter 5

Extended generalized blockmodeling and bibliometric analysis

To be able to identify emerging trends from research domains and map scientific knowledge, provides an appropriate understanding of how to progress in a new research direction. Such studies are usually carried out through the analysis of electronic literature available online.

Bibliometrics is the field that studies publication patterns and develop indicators for the evolution of scientific activities using statistical and mathematical models. It's focused on discovering and understanding regularities that exist in the way information is produced and used. Units of analysis can be words, metadata fields, publications, authors, journals, research groups, institutions, sub-fields, disciplines or geographic regions [47].

Natural language processing (NLP) is a range of computational techniques for the automatic analysis and representation of human language [12]. Its goal is to aid computers to understand human language.

In this context, Digital Libraries (DL) are definitely a place to study the combination of bibliometric techniques and Natural Language Processing (NLP), in order to improve the ability to explore relationships between entities of interest in metrics research.

There are some methods in the literature that work directly at the text level in bibliometrics and use NLP techniques. The most common are: Porter (2006); Zitt and Bassecoulard (1994); Glenisson et al. (2005). However, only few take real advantage of techniques (NLP).

In this thesis, we are interested in the approach adopted in TermWatch proposed by Sanjuan et. Al [41], [29], [30] which consists of a combination of techniques from three disciplines: Natural Language Processing (NLP), Data Mining and Graph The-

ory, mainly because its approach does not require great databases or long learning processes.

Our goal here is to show that the extended generalized blockmodeling is a good alternative for the graph decomposition process in the TermWatch pipeline. In this chapter, we evaluate the use of the extended generalized blockmodeling approach in the field of Natural Language Processing (NLP) and bibliometrics through the analysis of terrorism research datasets between 1990-2006. These datasets are publication records of peer-reviewed journal articles downloaded from the Web of Science (WoS).

The chapter is organized as it follows: Section 5.1 gives a general description of TermWatch System. In section 5.2 we present the terminological graph extraction process in details. Then in section 5.3 we show how an association graph can highlight a family of formal concepts and their relations based on the unique atom decomposition. At section 5.4 we show the analyzes and results obtained using the extended generalized blockmodeling as alternative to graph decomposition and some conclusions from this experiment.

5.1 Overview of TermWatch

As mentioned earlier, TermWatch is the result of a combination of techniques from Natural Language Processing (NLP), Data Mining and Graph Theory. It's designed to map research topics from unstructured texts and track their evolution in time [41].

In TermWatch, (NLP) techniques are used to extract significant units of text and to identify relevant information between them, called multi-word terminology units. These multi-word terminological units are nothing more than text chunks. Such text chunks correspond to domain concepts where the linguistic relations are lexical, syntactic and semantic variations.This first step of grouping of term variants ensures that semantically closed terms that reflect different aspects of the same topic end up in the same cluster at the end of the process.

Based on data mining (DM) techniques, TermWatch implements a hierarchical clustering algorithm specially designed for the characteristics of multi-word terms. This algorithm groups the terms of several words into close semantic classes called components; co-occurrence information is optional. Clusters are represented as an undirected graph.

Then the system decompose the undirected graph which are complex graph into more legible subgraphs that representes a coherent networks of research topics. This allows to split complex terminological networks of topics extracted by TermWatch based on their graph theoretic properties in order to identify sub-structures that represent highly connected sets of topics called central atom and distinct sets of topics called peripheral atoms.

Even though the system consists of a combination of techniques from three different disciplines, the entire process, which starts with the extraction of raw texts provenient

from a set of documents and reaches the mapping of the domain's topics, can be divided into five main stages: multiword term extraction, term variants identification, term clustering, graph decomposition and visualization.

Roughly, a step-by-step strategy, moving from input texts to mapping topics, consists of: first, from an input corpus composed of raw texts, building a scientific corpus that represents a research question. Second, terminological noun phrases (NPs) of maximal length are extracted from the scientific corpus. It can be done throught TreeTagger Schmid (1994) or any POS tagger algorithm.

In corpus linguistics, part-of-speech tagging (POS tagging or PoS tagging or POST), also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context — i.e., its relationship with adjacent and related words in a phrase, sentence, or paragraph [1].

After the POS tagging, the pipeline select NPs based on their syntactic structure and also using an enhanced term weighting function in order to retain only domain terms.

In the third step, terms with semantic variants of each others are detected and clustered using a hierarchical algorithm, the result is a three-level structure of domain terms. The first level are the terms. The second level are components that group together terms semantically closed or terms synonyms. Then the clustering algorithm proposed by Ibekwe-SanJuan (1998a) is applied to this second level, to group terms based in a weighted graph of term variants. Components and clusters are labeled according to their most active term and can be used as document features.

The final step consists of decomposing association graphs into atoms. In this stage a atom is a subgraph without clique separators, and each clique corresponds to a formal concept.

By definition [7], [43], a clique separator of G, is a set of pairwise adjacent vertices (that means a complete subgraph), whose removal disconnects the graph.

The major atoms are detected and visualised using force-directed positioning algorithms. The periphery of large atoms is highlighted, to reveal new concepts that arise in a domain which are normally represented by a larger central atom.

5.2 Terminological graph extraction

5.2.1 Corpus Tagging

Word forms are often ambiguous in their part-of-speech (POS)[42]. For example, the English word "bear" can be used as a verb "Your efforts will bear fruit" or as a noun "She saw a bear". As shown in the example, this ambiguity is normally resolved by the context of a word.

Automatic part-of-speech taggers use contextual predictability to mark words. A POS tag is a label assigned to each word in a text corpus to indicate the part-of-speech and also other grammatical categories such as tense, number (plural/singular), case etc.

In some cases, the POS tag is useful to distinguish the word sense (the meaning of the word), as shown in the above sentences, the word bear has completely different senses. In other cases, it is used to explain the syntactic role of a word and we can often infer semantic information from this due to our knowledge of how this syntactic role is commonly used semantically.

POS tag is used into a variety of NLP tasks and are very useful, since it provide linguistic signal on how a word is being used within the scope of a phrase, sentence, or document.

To tag the corpus termWatch uses TreeTagger[42]. The TreeTagger method uses a decision tree and probabilities for part-of-speech tagging in texts and it has been applied successfully to different languages like German, English, French, Italian, Dutch or Spanish. Gohring A. (2009) reported a macro-average of 85.44% precision and 80.77% recall, and a micro-average of 93.53% precision 93.53% recall for Spanish [40]. Table 5.1 display treeTagger sample output for the sentence *The TreeTagger is easy to use*.The first column display the word, the second column the tag label, the third column is the tag description.The last column is the lemma, the lemma is a word that stands at the head of a definition in a dictionary.

word	**pos**	**description**	**lemma**
The	DT	singular determiner/quantifier (this, that)	the
TreeTagger	NP	proper noun or part of name phrase	TreeTagger
is	VBZ	verb, 3rd. singular present	be
easy	JJ	adjective	easy
to	TO	infinitive marker to	to
use	VB	verb, base form	use

Table 5.1: *Example of tags from TreeTagger.*

5.2.2 Term Extraction

Once the corpus is tagged, multi-word terms are extracted through contextual rules. As a rule example, let's consider the one shown in figure 5.2, used to extract substantive terminological sentences using preposition *of*. As result example can be a simple noun phrases (NPs) like *stress disorder* or a complex one like *post-traumatic stress disorder*.

Besides rules, transformation operations, like permutation are used for grouping together syntactic variants of the same concept, since terms are extracted in two possible syntactic structures: NPs with prepositional attachment (execution of innocent victims) and compounds (innocent victims' execution). In addition, no limit is imposed on the

length of the terms extracted. More details about rules and operation can be found in [28]. Unlike other common approaches, such as *bag-of-word*, the extraction phase uses rules and transformation operators that respect the structure of domain terminology, allowing to detect emerging domain terms.

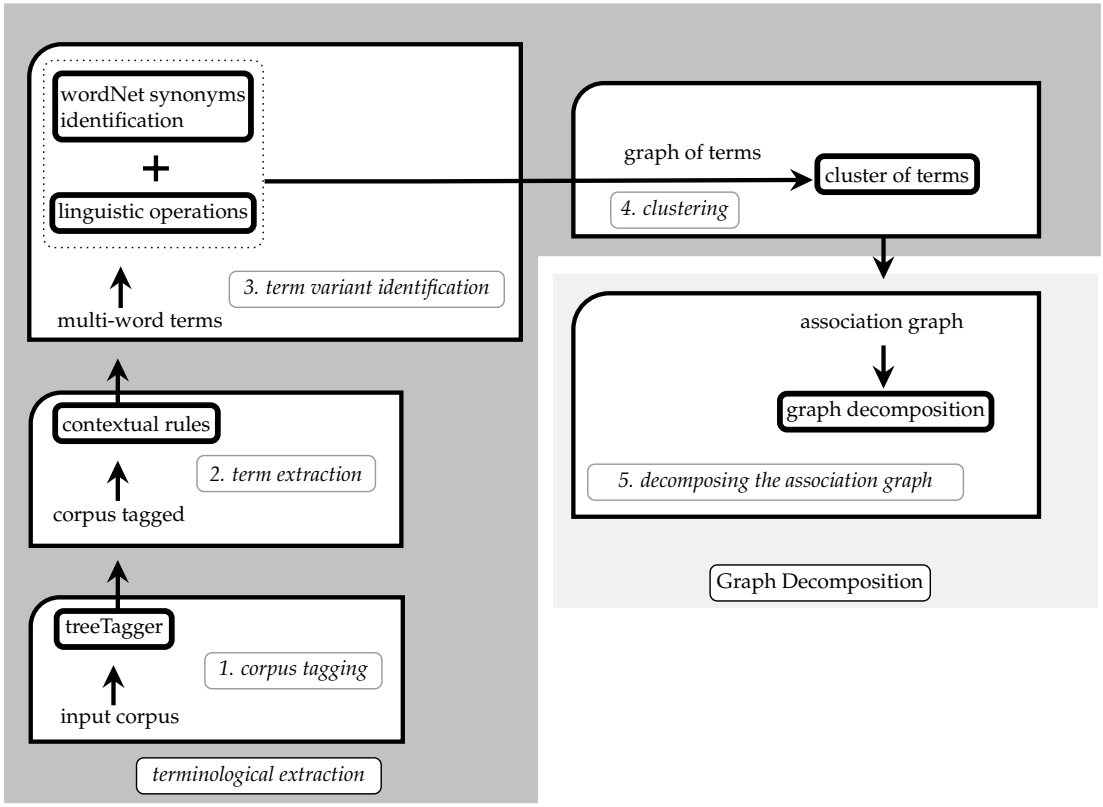


Figure 5.1: *Termwatch pipeline*

if <mod>⋆<N >+of<mod>⋆<N >+<prep1><verb><mod>⋆<N >+
then return:<mod>⋆<N>+ of<mod>⋆<N>+ **and**<mod>∗<N>+
where: < mod > is a determiner or an adjective
<N> is any of the noun tags
<prep1> is all the prepositions excluding *"of"*
⋆is the Kleene's operator (zero or n occurrences of an item)
+ is at least one occurrence

Figure 5.2: *Example of contextual rules used to extract multi-word terms*

5.2.3 Term variants identification

In TermWatch, the identification of term variants, also known as terminological variations, is made through linguistic operations, such as lexical *inclusion* and *substitution*, and also through rules to identify synonym links when searching for multi-word terms in a WordNet synset.[1]

Lexical inclusion refers to operations such as insertion (*severe poisoning* $\rightarrow$ *severe food poisoning)*, modifier or head word expansion *(disaster intervention* $\rightarrow$ *disaster intervention call)*, where a shorter term is incorporated into a longer term.

The lexical substitution, is used in the case terms of identical length, sharing a subset of lexical items and has one divergent item as *(political violence threat*$\rightarrow$ *political violence campaign)*.

Lexical substitutions generates highly connected graphs of term variants (cliques) that can include a certain amount of noise (false relationships), as substitutions operations tend to indicate a loose type of semantic association between terms.

To avoid such noises the graph is filtered using two criteria: keeping only substitutions that involve terms of length ≤ 2 if the words in the same grammatical position are found in the same WordNet Synset.

Termwatch also uses WordNet synset to extract synonym links between multiple words; therefore, multi-word terms (MWT's) are considered to be in a synonymy relation if two of their words are in the same WordNet synset, occupy the same grammatical role in the terms and are found in the same position.

Table 5.2 displays synonyms found by using WordNet synset extraction approach, italic words belongs to the same synset. Thanks to those synonymy links, concepts that

Term	**Synonym identified using WordNet synsets**
september 11 *wake*	september 11 *aftermath*
united states federal *agency*	united states federal *bureau*
risk society *conception*	risk society *concept*
Trauma type	*injury* type
Life-threatening problem	serious problem
Cyber-terrorist attack	hacker attack

Table 5.2: *Synonyms acquired from the terrorism corpus using WordNet synsets*

appear under different names are not dispersed across different clusters at the end of the process.

Table 5.3 gives examples of the different relations identified and the number of terms involved for the terrorism corpus.

[1] WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synset are the groupings of synonymous words that express the same concept.

variation type	example of term	example of variant	number of terms	number of links
Spelling	trauma center	trauma centre	93	138
Left exp.	food contamination	pet food contamination	1,799	2,709
Insertion	poisoning case	poisoning medical intervention case	41	60
Right exp.	disaster intervention	disaster intervention call	2,884	4,326
Modifier sub	acute stress disorder	posttraumatic stress disorder	14,062	95,651
Head sub.	political violence threat	political violence campaign	13,810	125,385
Wordnet Mod. sub	trauma severity	injury severity	185	99
Wordnet Head sub.	terrorist financing	terrorist funding	396	217

Table 5.3: *Terminological variations identified between terms in the terrorism corpus*

5.2.4 Term clustering

This step consists of clustering the graph of semantic term variants; to do so, TermWatch implements CPCL (Clustering by Preferential Clustered Link) [27] approach.

Terms (nodes) are grouped according to the type of relationship (edges), that depending on the linguistic meaning, the relationship xRy can be classified into one of two possible types of roles: COMP or CLAS.

CPCL has two clustering stages: first terms are clustered into connected components, throught COMP relations, secondly connected components are clusterd into classes, this time throught CLAS relations.

COMP relations, are those that induce near-semantic equivalence or links of synonyms, such as spelling variants. This type of connection is generated from operations such as: permutations, WordNet synonyms, expansions and insertions of modifiers. These types of relations form components representing the paradigms in the corpus but don't say anything about the association between these topics. So they are used as a prior clustering stage since COMP relations affect the topic change to a smaller degree. The transitive closure COMP* partition the whole set of terms into components.

As stated in [41] these components are not isolated and they are linked by transversal CLAS relations implying a change of headword, thus bringing to light the associations between research topics in the corpus.

While COMP relationships are used to represent links between synonyms, CLAS relationships involve a topical change between two terms, that is, where the main word is different. For instance, the shift of focus from "criminal assault" to the victim in "criminal assault victim". This type of relationship results from operations such as head expansion and replacement.

CLAS relations cluster components based on the following principle: two components c_i and c_J are clustered if the link between them is stronger than the link between either of them and any other component c_k. which has not been clustered neither with c_i nor with c_J. A link (edge) is draw between two terms (nodes) if one is a COMP relation of the other, then the first step is to clustering terms for which there is a sequence of variations in COMP.

First TermWatch partition the graph of term variations in two classes COMP and

CLAS. Then, based in the COMP subgraph, clusters (components) are labeled by its most active term. This step ensures that semantically close terms (nodes) which reflect different aspects of the same topic end up in the same cluster at the end of the process. Lastly, components formed by COMP relations are clustered using the agglomerative hierarchical process using the weight of CLAS relationships between each component.

Terms in component "terrorist attack"
terrorist attack, presumed terrorist attack, limited terrorist attack, national terrorist attack, international terrorist attack, explosive terrorist attack, deliberate terrorist attack, deliberate smallpox terrorist attack, smallpox attack, covert smallpox attack, chemical terrorist attack, th terrorist attack, year terrorist attack
Some components in the clique around "terrorist attack"
anthrax infection, toxic chemical, medium representation, 9/11 event, september 11 at- tack, current PTSD, new york time, pharmaceutical industry, american history, united kingdom, potential terrorist, militant islam, safety sense, national terrorist attack im- pact, distress symptom, decontamination area, immigration policy

Table 5.4: *Main component of the cluster "terrorist attack" and related cluster*

Table 5.4 shows as example, the content of the biggest cluster automatically labeled as "terrorist attack" (label of its main component). The other terms in the cluster result from co-occurrence[2] links. The lower part of this table, displays the nodes that surrounding around the cluster "terrorist attack" which forms a clique.

5.3 Association Graph analysis

Analyzing the relationship between clusters and documents is also necessary, as it allows identification of the association rules and, in doing so, reveals dependencies between formal concepts into the corpus.

As defined by Agrawal et al, a cluster is said to be related to a document if the cluster contains at least one term in the document. Clusters are then considered as items and each document defines an itemset.

More formally, let us define $\mathcal{D}$ as a collection of documents $d_1, d_2, \ldots, d_n$ and each document d_x as an itemset. We will reference them as *document itemsets*, and *clusters* are then considered as items.

Let $\mathcal{S}$ be an integer threshold, so a *frequent itemset* is a set of items that are included in at least $\mathcal{S}$ *document itemsets*. Frequent occura ence of itemset allows the revelation of hidden dependences in general.

Frequent itemsets of size 2 induce an association graph where nodes are items and there is a link between two nodes i and j if the pair $\{i, j\}$ is a frequent itemset.

[2] co-occurrence is the counting of paired data within a collection unit

This association graph is usually too dense to be visualized; it's common to use techniques to select resources based on some measures, such as mutual information or log probability.

According to Sanjuan [41] this approach has two drawbacks: first, the resulting graph structure depends on the selected measure. Second, it is not adapted to highlight larger itemsets. Sanjuan also states that to visualize large frequent itemsets on the association graph, a decomposition approach is needed that preserves cliques induced by frequent itemsets.

Termwatch uses Formal Concept Analysis (FCA) as formal framework [46] for association rules discovery. FCA is a mathematical theory about concepts and concept hierarchies that allows the derivation of concept hierarchies from datasets. It models concepts as units of thought, consisting of two parts: extensions and intensions. The extension consists of all objects belonging to the concept. The intension consists of all attributes common to all those objects.

FCA can be understood as conceptual clustering method, which clusters simultanously objects and their descriptions.

Using the FCA theory an itemset can be assimilated as a concept used in some documents. That is, assuming that formal concept is a extension of $\mathcal{D}$, and let an intension be a set of items I, such that a document d_x is related to all items in I if and only if d_x is in $\mathcal{D}$. In this way, a formal concept determines a correspondence between a set of documents and a set of items.

A frequent itemsets are *intensions* of some *formal concept*. Identify in the association graph the closed frequent itemsets is necessary to better visualize *intensions*, and the graph decomposition method seems to be an interesting approach once such techniques have a tendance to preserve the cliques induced by closed frequent itemsets.

Figure 5.3 displays TermWatch summarized scheme with focus in the association graph generation. The top left side of the image illustrate sa corpus of documents constituted by raw text.

The upper right corner shows the steps of terminological extraction phases. In this stage, after the corpus tagging, noun phrases (NPs) are extracted from ISI abstracts, then multi-word terms are extracted throught contextual rules, as describe in section 5.2.2.

Then the identification of term variants is done through linguistic operations, such as left expansions, insertions and so on, as describe in n section 5.2.3. The resulting is a graph of terms where nodes are terms and there is one edge between each term and its variants.

The graph of terms is clustered first based on COMP relations. These clusters are called then components, and only those with at least two vertices are considered. Then, clustered again using the agglomerative hierarchical process based on the weight of CLAS relations, and each of these components are labelled by the NP having the most number of variants, as explained in section 5.2.4

The bottom of figure 5.3, is the composition of three illustrations: a) shows the clusters resulting from terminological extraction phase; In illustration b) components (clusters) are now considered as items, and the correspondence between documents and itemsets is made; c) displays the association graph generated, where components (nodes) are items and the edge is the results from frequent itemset with size 2.

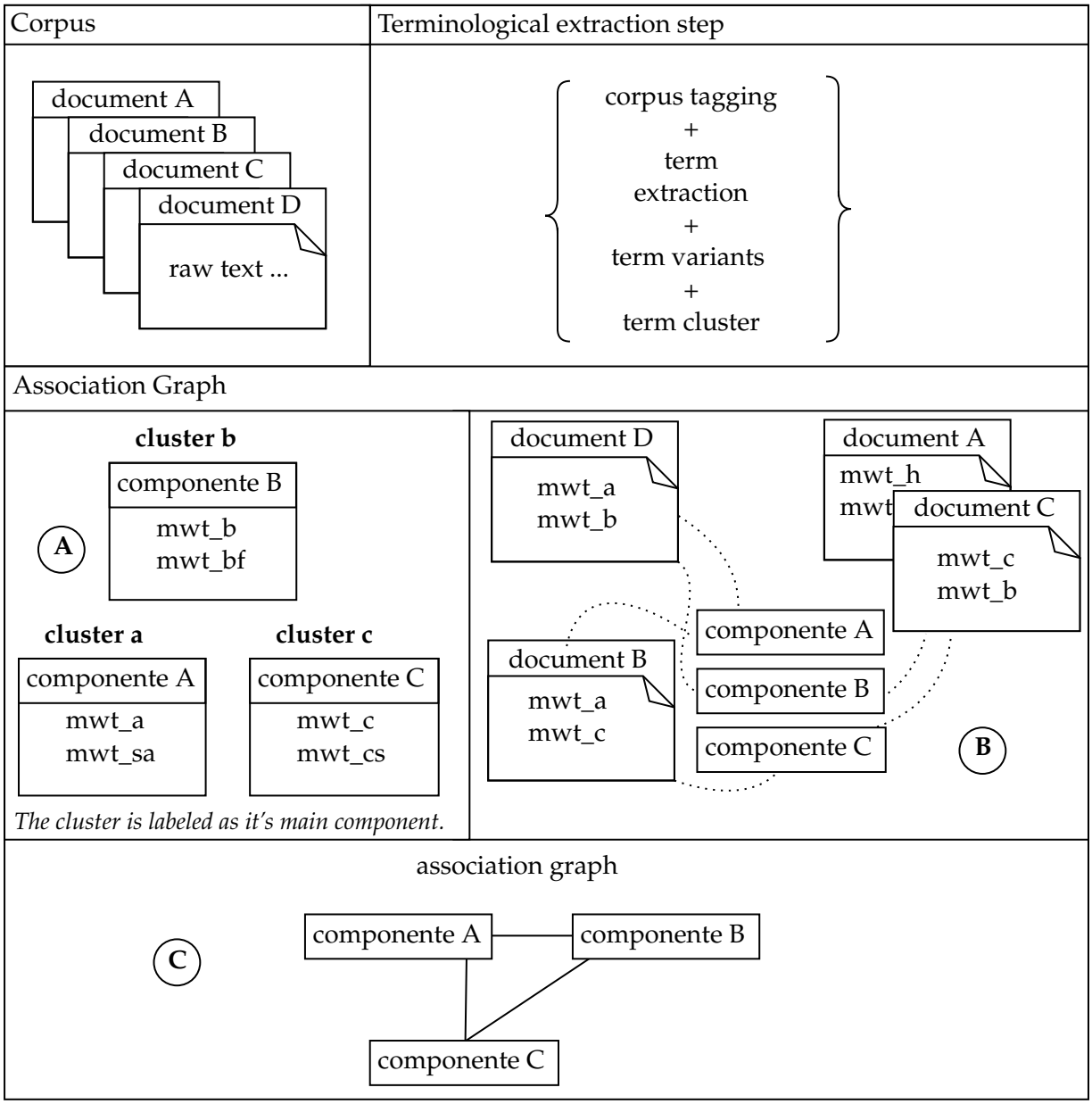


Figure 5.3: *TermWatch summarized scheme with focus in the association graph.*

5.3.1 Graph decomposition

To study the way clusters are associated with documents, termWatch uses a graph decomposition approach. Let us start by recalling some notations, $G = (V, R)$ is a graph with the node set V and the edge set as R. We denote an edge as pair (v_1, v_2), of nodes $v_1, v_2 \in V$.

A decomposition of a graph G is a set of edge-disjoint subgraphs $\{H_1, H_2, \ldots, H_n\}$ that partition the nodes of G, such that every edge of G belongs to exactly one H_x. If H_x is a subset of the nodes, $G(H_x)$ is the subgraph of G induced by H_x; if $G(V - H_x)$ is disconnected, H_x is a separator.

Various types of separators decompositions have been studied by several authors by imposing conditions on the paths in the decomposition [5]. We are interested in the minimum clique separator, so let us provide here few other definitions; A *clique* is a set of pairwise adjacent vertices; A subset of vertices H_x is a minimal separator of G if there exist v_i, $v_j \in G$ such that v_i and v_j are not connected in $G(V - H_x)$, and H_x is minimal for inclusion with this property. Then H_x is called minimal clique separator if $G[H_x]$ is a clique.

To decompose graphs subjacent to corpus, TermWatch uses a graph decomposition technique based on minimum clique separators that, once removed from the original graph, will result in several separate subgraphs, called atoms. By definition, an atom is a sub-graph where there is no clique separator.

M.D. Biha et al. (2007) states that in this kind of graph, it often occurs that there is a central atom and several small ones. We refer to the biggest atom as the central atom.

This technique reveals the atomic structure of a graph, allowing the uncover ing of special concepts that work as interfaces between sub-domains or between domain kernels and external related objects, as well as intrinsically related concepts to the heart of the domain.

The advantage of atom decomposition is that it's unique. In addition, the number of atoms and their distribution size are good indicators of their structural complexity, besides it can be computed in quadratic time.

The atom decomposition algorithm implemented in termwatch [17], computes the atomic graph structure and generates subgraphs:

1. the sub-graph that constitutes the central atom if it exists.

2. the network of atoms to visualize those at the periphery and the way they are connected to the central atom.

5.4 Case study

To conduct our experiments, we used TermWatch to map the dynamics of research in terrorism research between 1990-2006.

In our studies we concentrate efforts into the central atom, labelled biological terrorism, the goal is to unfold his internal structure.

The central atom has temporal information, which characterises four periods with thematic variations.

As discussed in the session before, Termwatch graph decomposition approach is focus into reveal unique way to structure graph into coherent sub-networks allowing a close view of a real concept network. However, this graph remains huge and because it's large size is quite difficult to visualize relevant informations.

We show here that blockmodeling can help to make apparent information in the graph generated in the previous decomposition; By testing different types of ideal blocks, properties such as: temporality, topic homogeneity/heterogeneity may emerge from the clustering.

The experiment with blockmodeling aims precisely to reveal what kind of property it's possible to "isolate", since a cluster, by definition, shares a property. We will see later on, in the numerical results, that this is the case, since blockmodeling may help:

1. to identify clusters of terms used in the same period (temporal homegeneity). Thus to automatically guess the number of periods (temporal homogeneity in the clusters);

2. to identify clusters of terms talking about the same topics. Hence to automatically detect topics (topic homogeneity in the cluster)

3. to construct clusters of terms talking about different topics. (topics heterogeneity in the clusters)

The generalized extended block modeling heuristic takes as input data the central atom, composed of 876 nodes and 2157 edges. We use three different types of blockmodels and, for all of them, N as the maximum number of partitions.

The difficulty in finding clusters said temporal homogeneity, is that the association graph edge represent co-occurence (co-apparition) in documents. Edge may exist between two term found in documents of two different periods. So temporal information is not directly included in the graph. It follows that identifying the periods corresponds to the documents where the nodes appears is not trivial.

Such decomposition difficulty allows us to show, through the probabilistic study, below, that the emerged properties cannot be found randomly.

5.4.1 Probability studies on terms graph

Let us consider a graph of terms with n nodes (terms) covering n periods or thematics. In a mathematical point of view a clustering of G nodes can be viewed as a partition of its nodes.

Definition 5.4.1. *A clustering of G will be said to be « temporal homogeneous » (resp.« thematic homogeneous ») iff each cluster contains nodes of the same period (resp. thematic).*

Temporal (or thematic heterogeneity will refer, at the opposition to at least two different periods (or thematics) in each cluster. We could distinguish two types of heterogeneity. A weak one corresponding to the explanations above. And a strong one where each cluster contains not a only, at least two periods (thematics), but all periods (thematics).

We will give some probabilities and asymptotic results for homogeneity and heterogeneity concepts. Although the definitions given for the concepts deal with period and thematic, they can be generalized for any type of properties.

It suffices to replace « periods » or « thematics » by « types » (or « classes », or « groups »). In this case, the terms are of r types (or classes) and the definitions of homogeneity and heterogeneity remain unchanged using « types » instead of period (and thematic). So from now, we consider that r in the number of type and will talk about homogeneous or heteregeneous types. The number of ways to partitions n elements is known as the Bell Number B_n that corresponds to the number of clusterings and is given by the recurrence relation:

$$B_n = \sum_{k=0}^{n-1} B_k C_k^n$$

where $B_0 = B_1 = 1$, and C_k^n the number of subsets of k elements in a set of n elements, for example:

$$C_k^n = \frac{n!}{k!(n-k)!}$$

There is no known « simple » non-recurrent expression of B_n depending directly on n. However, some asymptotic approximation exist. One of them, given below, will be practical usefulness for our study.

Theorem 5.4.2 (De Brujn, 1981)**.** *For any integer $n \geq 2$, we have*

$$\frac{\log B_n}{n} = \log(n) - \log(\log(n)) - 1 + \frac{\log(\log(n))}{\log(n)} + \frac{1}{\log(n)} + \frac{1}{2}\left[\frac{\log(\log(n))}{\log(n)}\right]^2 + O\left[\frac{\log(\log(n))}{(\log(n))^2}\right].$$

From this approximation, we will derive a simpler expression where "dominant" parts are emphasized.

Corollary 5.4.2.1. *For any integer $n \geq 2$, we have*

$$\frac{\log B_n}{n} = n\log(n) - n - n\log(n)\varepsilon(n)$$

where $\lim_{n\to+\infty} \varepsilon(n) = 0$.

Proof. It is directly obtained from Theorem 5.4.2 by taking:

$$\varepsilon(n) = \frac{\log(\log(n))}{\log(n)} - \frac{\log(\log(n))}{\log(n)^2} - \frac{1}{\log(n)^2} - \frac{1}{2\log(n)}\left[\frac{\log(\log(n))}{\log(n)}\right]^2 - \frac{1}{\log(n)}O\left[\frac{\log(\log(n))}{\log(n)^2}\right].$$

All the expression in $\varepsilon(n)$ convenges to O when n tends to $+\infty$. □

From the corollary, it can be also seen that:

$$\lim_{n\to+\infty} \log(B_n) = \lim_{n\to+\infty} n\log(n) - n.$$

This equality of the limits will play an important role on the asymptotic results on probabilities. The following lemma will be also regularly used in our proofs.

Lemma 5.4.3. *Let $\{\mu_n\}_{n\geq 0}$ be a real valued series. Then, $\lim_{n\to+\infty} \log(\mu_n) = -\infty$ if and only if $\lim_{n\to+\infty} \mu_n = 0$.*

Proof. The function « log » being a continuous real valued function, we clearly have

$$\lim_{n\to+\infty} \log(\mu_n) = 0 \Rightarrow \lim_{n\to+\infty} \log(\mu_n) = \log(\lim_{n\to+\infty} \mu_n) = -\infty.$$

Inversely, if $\lim_{n\to+\infty} \log(\mu_n) = -\infty$ then

$$\forall M \in \mathbb{R}^+,\ \exists N \in \mathbb{N} |\ \forall n \geq N,\ \log(\mu_n) \leq -M.$$

It follows that $\mu_n = e^{\log(\mu_n)} \leq e^{-M}$. But $\lim_{n\to+\infty} e^{-M} = 0$ and since $\mu_n \geq 0$ we obtain $\lim_{n\to+\infty} \mu_n = 0$. □

Now, let us notice $m_i \geq 1, i = 1, 2, \ldots, r$ the number of terms (nodes) in the graph of the same type i. Using Bell numbers the following result can be proved.

Theorem 5.4.4. *The number of homogeneous clusterings (or partition) is:*

$$\prod_{i=1}^{r} B_{m_i} = B_{m_1} \times B_{m_2} \times \ldots \times B_{m_r}.$$

Proof. To have an homogeneous partition, it suffices to partition separately each group of terms of the same type. An homogeneous partition will be then the union of the group partition. There are B_{m_i} ways to partition the nodes of type i. The total number of homogeneous partition of the n nodes is thus the B_{m_i} product. □

Knowing the total number of homogeneous clusterings, we can now write an expression giving the probability to have an homogeneous clustering if the clusters are built randomly.

Theorem 5.4.5. *Let us build a clustering with a process for which we know that all clusterings will have equal chance to be chosen (uniform distribution). In this case the probability to obtain homogeneous clustering is:*

$$\mathbb{P}_m = \frac{\prod i = 1^r B_{m_i}}{B_n}$$

where m is the vector of components $m_1, m_2, \ldots, m_r$; for instance $m = (m_1, m_2, \ldots, m_r)$.

Proof. Probability definition: number of homogeneous clusterings divided by the total number of clusterings. □

In the rest of the study, some asymptotic results on the limit of $\mathbb{P}_m$, when $\| m \| \to +\infty$ will be given. Saying that $\| m \| \to +\infty$ means that some components m_i tends to $+\infty$, not necessarily all and uniformly. The notation $\| . \|$ represents any vector norm (for instance the euclidean morm). Notice that, as a consequence, when $\| m \| \to +\infty$, we also have $n \to +\infty$, since $n = m_1 + m_2 + \ldots + m_r$.

Theorem 5.4.6. *If $r \geq 2$ then*

$$\lim_{\|m\| \to +\infty} \mathbb{P}_m = 0.$$

Proof. This result shows that bigger is n and closer to 0 is the probability to find randomly homogeneous clusters, if we have at least 2 differents term (node) types. Notice that when $r = 1$ (or period) $\mathbb{P}_m = 1$ since any terms refer to the same period. This explains why we put $r \geq 2$. To prove the theorem we will make use of Corollary 5.4.2.1 and Lemma 5.4.3.

$$\log(\mathbb{P}_m) = \log\left(\prod_{i=1}^{r} B_{m_i}\right) - \log(B_n)$$
$$= \left(\sum_{i=1}^{r} B_{m_i}\right) - \log(B_n).$$

By corollary 5.4.2.1,

$$\lim_{\|m\|\to+\infty} \log(\mathbb{P}_m) = \lim_{\|m\|\to+\infty} \left(\sum_{i=1}^{r} \log(B_{m_i})\right) - n\log(n) + n$$

But as $m_1 + m_2 + \ldots + m_r = n$ we have

$$\lim_{\|m\|\to+\infty} \log(\mathbb{P}_m) = \lim_{\|m\|\to+\infty} \sum_{i=1}^{r} \left[\log(B_{m_i}) - m_i\log(n) + m_i\right].$$

Now, let us analyze the limit of each term in brackets:

$$\left[\log(B_{m_i} - m_i\log(n) + m_i\right].$$

For each i there is two cases.

Case 1: m_i is bounded i.e there exists a real constant M such that $m_i \leq M$ when $\| m \| \to +\infty$. In this case, since:

$$\log(B_{m_i}) - m_i\log(n) + m_i = \log(n)\left(\frac{B_{m_i}}{n^{m_i}}\right)$$

then

$$\lim_{\|m\|\to+\infty} \log(B_{m_i}) - m_i\log(n) - m_i \geq \left(\lim_{\|m\|\to+\infty} \frac{B_{m_i}}{n^{m_i}}\right) = -\infty.$$

Because B_{m_i} is also bounded and

$$\lim_{n\to+\infty} \frac{B_{m_i}}{n^{m_i}} = 0$$

thus

$$\lim_{\|m\|\to+\infty} \log(B_{m_i}) - m_i\log(n) - m_i = -\infty.$$

Case 2: m_i is not bounded. Thus it tends to $+\infty$. Corollary 5.4.2.1 can be re-used for $\log(B_{m_i})$. We have :

$$\lim_{\|m\|\to+\infty} \log(B_{m_i}) - m_i\log(n) - m_i = \lim_{\|m\|\to+\infty} m_i\log(m_i) - m_1\log(n) + m_i$$
$$= \lim_{\|m\|\to+\infty} \log\left(\frac{m_i}{n}\right)^{m_i}.$$

Notice that for any i, $\frac{m_i}{n} < 1$. Now two sub-cases.

Sub-case 1: if $\lim_{\|m\|\to+\infty} \log \frac{m_i}{n} = 1$ then $\lim_{\|m\|\to+\infty} \log \left(\frac{m_i}{n}\right)_i^m = 0$. But also, as

$$\frac{m_i}{n} + \frac{n - m_i}{n} = 1$$

we have

$$\lim_{\|m\|\to+\infty} \frac{n - m_i}{n} = 0.$$

As consequence, for any $j \neq i$, $\lim_{\|m\|\to+\infty} \log \left(\frac{m_j}{n}\right)^{m_j}$. It follows that for this index j, we will necessary have:

$$\lim_{\|m\|\to+\infty} \log(B_{m_j}) - m_j \log(n) - m_j = -\infty.$$

Sub-case 2: $\frac{m_i}{n}$ has no limit or $\lim_{\|m\|\to+\infty} \frac{m_i}{n} \neq 1$.
In this case, it will exist $\varepsilon < 1$ such that $\frac{m_i}{n} \leq \varepsilon < 1$ for any m. Thus $\left(\frac{m_i}{n}\right)^{m_i} \leq \varepsilon^{m_i}$. It follows that

$$\lim_{\|m\|\to+\infty} ln \left(\frac{m_i}{n}\right)^{m_i} = -\infty.$$

To resume, we have proved that in any case, it always exists an index i for which $\lim_{\|m\|\to+\infty} \log(B_{m_i}) - m_i \log(n) + m_i = -\infty$. And for any i,

$$\lim_{\|m\|\to+\infty} \log(B_{m_i}) - m_i \log(n) + m_i \leq 0.$$

Thus $\lim_{\|m\|\to+\infty} \mathbb{P}_m = -\infty$. Implying by Lemma 5.4.3 that $\lim_{\|m\|\to+\infty} \mathbb{P}_m = 0$ which conclude the proof.

□

Based on this theorem, we can say that finding randomly homogeneous partition is hard as n increase. The added-value of the blockmodeling approach can be validated with this theorem. If homogeneous clusterings are frequently found on large size instances, this means that it is not the result of randomness but of the fact that the approach catchrs some properties allowing homogeneity in the clusters. Let's now try to have some insights on heterogeneity.

Theorem 5.4.7. *Assuming again that the clusterings are uniformly distributed, the probability to obtain a strong heterogeneous clustering is :*

$$\mathbb{H}_m = \frac{\sum_{k=1}^{m_1} (k!)^{r-1} \prod_{i=1}^{r} S(m_i, k)}{B_n}.$$

where

- $S(m_i, k)$ *is the number of partitions of the* m_i *terms in k subsets.*
- *we also assume that :* $m_1 \leq m_2 \leq \ldots \leq m_r$.

Proof. To be sure that each cluster in a clustering contains all types, let us proceed as follows. Suppose that the partition has k subsets, we can partitioned in k subsets each group of terms, of the same type, and then combine the subsets to obtain k strong heterogeneous clusters. Let us illustrate this process by the following picture:

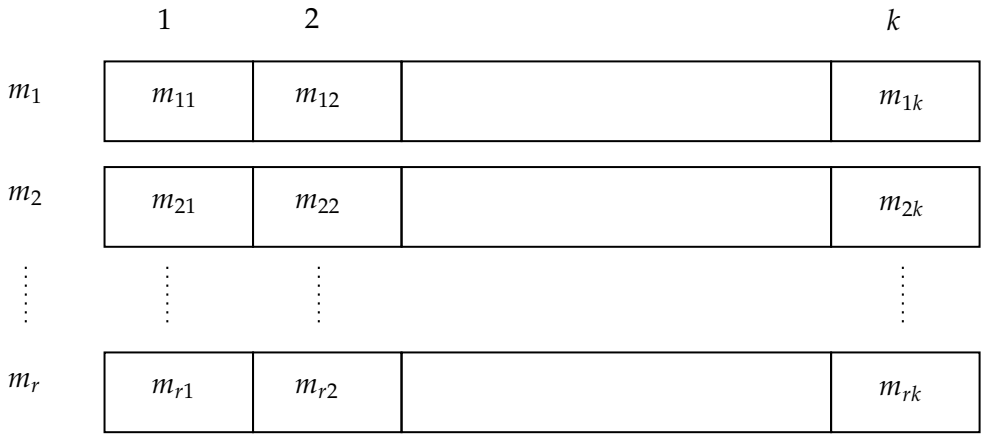


Figure 5.4: *Illustration process*

Each table represents the m_i terms of type i. For each i, $\{m_{i1}, m_{i2}, \ldots, m_{ik}\}$ is a partition of the m_i terms in k subsets $m_{i1}, m_{i2}, \ldots, m_{ik}$. A strong heterogeneous cluster, can be obtained by picking one subset in each table. For instance,

$$m_{11} \cup m_{21} \ldots \cup m_{r1}$$

Notice that many $m_{i1}, m_{i2}, \ldots, m_{ik}$ are subsets. Their union is thus a subset of the n terms containing all term types (by construction) if we repeat this picking process k times we will generate k subsets of the terms, each one being strongly heterogeneous. For instance:

$$\begin{array}{c} m_{11} \cup m_{21} \ldots \cup m_{r1} \\ m_{11} \cup m_{21} \ldots \cup m_{r1} \\ \vdots \quad \vdots \quad \vdots \quad \vdots \\ m_{1k} \cup m_{2k} \ldots \cup m_{rk} \end{array}$$

is a custering of the n terms in k subsets, each one having all term types. If the partition of each group of terms with same i is giving , there are $(k!)^{r-1}$ ways to generate a strongly heterogeneous clustering by the picking process. And for each group of terms of similar types, there are $S(m_i, k)$ possible partitions in k subsets. The total number of strong heterogeneous clusterings of the n terms, in a subsets, is thus:

$$(k!)^{r-1} \times \prod_{i=1}^{r} S(m_i, k)$$

By summing on k, we derive the total number for any number of subsets k. But as $m_1 \leq m_2 \leq \ldots \leq m_n$, the sum on k must stop at $k = m_1$. Because it is not possible to partition the m_1 terms of type i, in more than m_1 subsets. The probability comes directly by dividing with B_n. □

Giving some asymptotic results for strong heterogeneity is more difficult than for homogeneity. The theorem presents a result valid when $\| m \| \to +\infty$ but for a bounded value of m_1.

Theorem 5.4.8. *Let us suppose that $m_1 \leq m_2 \ldots m_r$, and m_1 never exceed a constant M when m varies then*

$$\lim_{\|m\| \to +\infty} \mathbb{H}_m = 0.$$

Proof. Its is known that

$$S(m_i, k) = \frac{1}{k!} \sum_{j=0}^{k} (-1)^{k-j} C_j^k j^{m_i}$$

$$\prod_{i=1}^{r} S(m_i, k) = \frac{1}{(k!)^r} \prod_{i=1}^{r} \left[\sum_{j=0}^{k} (-1)^{k-j} C_j^k j^{m_i} \right].$$

As m_1 is bounded, when m varies, so as for k. As a consequence

$$\begin{aligned} \lim_{\|m\| \to +\infty} \prod_{i=1}^{r} S(m_i, k) &= \lim_{\|m\| \to +\infty} \frac{1}{(k!)^r} \times k^{m_1 + m_2 + \ldots + m_r} \\ &= \lim_{\|m\| \to +\infty} \frac{1}{(k!)^r} \times k^n \end{aligned}$$

Thus

$$\lim_{\|m\|\to+\infty} \mathbb{H}_m = \lim_{\|m\|\to+\infty} \frac{\sum_{k=1}^{m_1} \frac{k^n}{k!}}{B_n}$$

As B_n tends to infinity

$$\lim_{\|m\|\to+\infty} \mathbb{H}_m = 0.$$

□

Under the assumptions made all over the theorems finding randomly homogeneous of strongly heterogeneous clusterings are asymptotically hard. But it is important to highlight the fact that theorem 4 cand 6 occurs for infinite values of n. Observing « visually » is not possible in practice because of the very large instance sizes, nor tractable in practice. For tractable instances, theorem 3 and 5 give exact probabilities.

5.5 Experiment analysis

As described in section 5.4, the network used as case of study, has temporal information and thematic variation, carried in the way nodes are connected with each other. In this section we discuss how temporal information and thematic variation is used to measure if the partition generated by our approach is a relevant representation of the network structure.

To analyze the quality of the clusters generated by our method, we first resampled the partition generated by the heuristic, reducing it to 150 nodes, but as we respect the cluster size proportions the total number of the nodes in a partition can vary. Secondly, information about time interval and thematic classification was added to all nodes.

To quantify the relevance of the partitions generated by our proposed method, we first calculate the coefficient of variance to determine how homogeneous (heterogeneous) the clusters of a given partition are, with respect to the period and thematic variation.

The coefficient of variation (CV) is defined as the ratio between the standard deviation σ and the average μ, $c_v = \frac{\sigma}{\mu}$, is a standardized measure of dispersion, wich allow compare variables that have different averages.

As the standard deviation represents how dispersed the data is in relation to an average, when comparing samples with different averages, it's use can generate interpretation errors.

Thus, when comparing two sets of data, the most homogeneous will be the one with the lowest coefficient of variation. As the partitions we compare do not have the same

number of clusters, and we are interested in the global indicator, we use the average of (CV) to define which partition is more homogeneous than another.

Using the terrorism network graph, two type of experiments were performed, referenced here as experiment A and experiment B. In experiment A, the set of ideal blocks are: null blocks off diagonal and complete blocks in the diagonal. Experiment B, seeks for null blocks off diagonal and row (column) dominant blocks for the diagonal. The main reason we are using, in both experiments, A and B, non-diagonal blocks, is that we are interested in analyze connections inside of cluster and not in between clusters.

By comparing the numerical results of experiment A and B, we concluded that experiment A has the most homogeneous partitions, and the reason for that is, the set of ideal block used are more stricted than B. In addition, this set of ideal blocks tends to build partitions with more clusters than experiment B.

This is also due to the fact that the set of ideal blocks used in experiment A forces the heuristic to concentrate its efforts in finding partitons such that, inside of cluster connections should be perfect (complete block in the diagonal), reducing the number of links between clusters (null block of diagonal). So reducing the size of clusters, also reduces objective value of the block modeling problem.

The second aspect that can be seen in the numerical results, when analyzing cluster variance in the partitions (please check table: 5.7, 5.8, 5.9, 5.10, 5.11), is that for the group of instances in experiment A, the partitions generated, are composed by cluster of nodes (terms) with low rate of variation of thematic and period. In addition, the total column shows that, for all partitions in experiment A, the size of the clusters that compose the partition tends to be the same, with small differences, even for the partition that has a bigger desviation error. Table 5.6 displays the deviation error for the partitions in experiment A and B.

The opposite of experiment A, is experiment B, in terms of numerical results, tables: 5.12, 5.13, 5.14, 5.15, 5.16. The tables show a heterogenous distribution of nodes within between clusters (column labeled total), and high degree of dispersion between clusters (column labeled variance), concerning to both, period and thematic classification.

This is due to the fact that set of ideal block in experiment B, be less stricted, and have a better match in concerning to the characteristics of the terrorism research network. We can also see, by comparing the column named cluster across all tables, that all partitions of experiment B have a partition size two times smaller than experiment A.

Another important fact to be highlighted here concerning to the set of ideal blocks used experiment B is: as it seeks to maximise the number of bridge nodes inside of cluster reducing the links between cluster, as result we have partitions with the highest degree of heterogeneity, with respect to period and thematic classification. This set of ideal block is suitable to the analysis of the evolutions of intensions over time across different thematics.

time interval	period	thematic	thematic description
1992-1995	1	1	body injuries in terrorist bombing
	1	2	is health care in response to the threat of biological and chemical
	1	3	post traumatic stress disorders
1996-1999	2	1	body injuries in terrorist bombing
	2	2	is health care in response to the threat of biological and chemical
	2	3	post traumatic stress disorders
2000-2003	3	1	body injuries in terrorist bombing
	3	2	is health care in response to the threat of biological and chemical
	3	3	post traumatic stress disorders
2004-2006	0	0	bioterrorism has emerged and become prominent

Table 5.5: *Classification of time interval, period, thematic and thematic description.*

partition	experiment A	experiment B
1	47692	928
2	47794	957
3	47852	1067
4	47896	1108
5	48038	1161

Table 5.6: *Deviation Error*

5.5.1 Concluding Remarks

In this chapter, we introduce the TermWatch approach to map knowledge, automatically extracting terminology graphs. We used TermWatch to map the dynamics of terrorism research between 1990 and 2006 and, as a result, we have several disjointed subgraphs called atoms.

We apply extended generalized block modeling to unfold the internal structure of the central atom, for this we use two different sets of ideal blocks: complete block and null block, called experiment A and second dominant and null column, experiment B.

Experimentation shows that set A builds clusters with nodes that emerged in a maximum of 2 different periods with little thematic variability. While experiment B generates partitions with better variability in terms of period and theme, each one seems more suitable to study the evolution of terms over time.

However, as the central atom remains large for the ILP model, we use the heuristic approach to study this problem, even if it is proven to converge to a satisfactory result, we cannot guarantee the optimal result. In addition, the blockmodeling framework can find multiple equally optimal solutions. But it is valuable because it provides an insight of how the connections between the nodes are created and evolve over time.

	percentage by of nodes period					number of nodes by period						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	33,3	66,7	0,0	0,0	0,0	3	6	0	0	0	9	2
1	20,0	0,0	60,0	20,0	0,0	2	0	6	2	0	10	2
2	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
3	55,6	33,3	11,1	0,0	0,0	5	3	1	0	0	9	2
4	0,0	0,0	40,0	60,0	0,0	0	0	4	6	0	10	3
5	10,0	0,0	30,0	40,0	20,0	1	0	3	4	2	10	2
6	40,0	30,0	30,0	0,0	0,0	4	3	3	0	0	10	2
7	11,1	33,3	22,2	33,3	0,0	1	3	2	3	0	9	1
8	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
9	0,0	22,2	33,3	0,0	44,4	0	2	3	0	4	9	1
10	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
11	20,0	40,0	0,0	20,0	20,0	2	4	0	2	2	10	1
12	33,3	66,7	0,0	0,0	0,0	3	6	0	0	0	9	2
13	50,0	50,0	0,0	0,0	0,0	5	5	0	0	0	10	3
14	33,3	22,2	11,1	33,3	0,0	3	2	1	3	0	9	1
partition	24,8	29,1	26,2	14,2	5,7	35	41	37	20	8	141	-

	percentage of nodes by thema					number of nodes by thema						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	33,3	44,4	0,0	22,2	0,0	3	4	0	2	0	9	1
1	20,0	40,0	40,0	0,0	0,0	2	4	4	0	0	10	2
2	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
3	55,6	33,3	11,1	0,0	0,0	5	3	1	0	0	9	2
4	0,0	10,0	60,0	30,0	0,0	0	1	6	3	0	10	2
5	10,0	10,0	10,0	50,0	20,0	1	1	1	5	2	10	2
6	40,0	30,0	10,0	20,0	0,0	4	3	1	2	0	10	1
7	11,1	11,1	55,6	22,2	0,0	1	1	5	2	0	9	2
8	22,2	33,3	44,4	0,0	0,0	2	3	4	0	0	9	1
9	0,0	44,4	11,1	0,0	44,4	0	4	1	0	4	9	2
10	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
11	20,0	30,0	30,0	0,0	20,0	2	3	3	0	2	10	1
12	33,3	44,4	22,2	0,0	0,0	3	4	2	0	0	9	1
13	50,0	20,0	30,0	0,0	0,0	5	2	3	0	0	10	2
14	33,3	22,2	22,2	22,2	0,0	3	2	2	2	0	9	0
partition	24,8	27,0	31,2	11,3	5,7	35	38	44	16	8	141	-

Table 5.7: *Experiment A - Partition A1*

	percentage by of nodes by period					number of nodes by period						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
1	55,6	11,1	11,1	22,2	0,0	5	1	1	2	0	9	2
2	33,3	11,1	22,2	33,3	0,0	3	1	2	3	0	9	1
3	20,0	50,0	10,0	10,0	10,0	2	5	1	1	1	10	2
4	30,0	70,0	0,0	0,0	0,0	3	7	0	0	0	10	3
5	33,3	55,6	11,1	0,0	0,0	3	5	1	0	0	9	2
6	10,0	50,0	20,0	20,0	0,0	1	5	2	2	0	10	2
7	30,0	30,0	30,0	10,0	0,0	3	3	3	1	0	10	1
8	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
9	22,2	33,3	22,2	22,2	0,0	2	3	2	2	0	9	0
10	50,0	50,0	0,0	0,0	0,0	5	5	0	0	0	10	3
11	22,2	0,0	44,4	22,2	11,1	2	0	4	2	1	9	1
12	20,0	10,0	50,0	10,0	10,0	2	1	5	1	1	10	2
13	20,0	10,0	30,0	10,0	30,0	2	1	3	1	3	10	1
14	33,3	33,3	33,3	0,0	0,0	3	3	3	0	0	9	1
partition	28,2	32,4	24,6	10,6	4,2	40	46	35	15	6	142	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total of nodes	variation
0	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
1	55,6	22,2	0,0	22,2	0,0	5	2	0	2	0	9	2
2	33,3	33,3	22,2	11,1	0,0	3	3	2	1	0	9	1
3	20,0	30,0	40,0	0,0	10,0	2	3	4	0	1	10	1
4	30,0	60,0	10,0	0,0	0,0	3	6	1	0	0	10	2
5	33,3	33,3	0,0	33,3	0,0	3	3	0	3	0	9	1
6	10,0	50,0	30,0	10,0	0,0	1	5	3	1	0	10	2
7	30,0	40,0	30,0	0,0	0,0	3	4	3	0	0	10	2
8	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
9	22,2	0,0	55,6	22,2	0,0	2	0	5	2	0	9	2
10	50,0	20,0	30,0	0,0	0,0	5	2	3	0	0	10	2
11	22,2	33,3	33,3	0,0	11,1	2	3	3	0	1	9	1
12	20,0	20,0	20,0	30,0	10,0	2	2	2	3	1	10	0
13	20,0	20,0	10,0	20,0	30,0	2	2	1	2	3	10	0
14	33,3	44,4	0,0	22,2	0,0	3	4	0	2	0	9	1
partition	28,2	29,6	26,8	11,3	4,2	40	42	38	16	6	142	-

Table 5.8: *Experiment A - Partition A2*

	percentage by of nodes period					number of nodes by period						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
1	22,2	11,1	11,1	55,6	0,0	2	1	1	5	0	9	2
2	50,0	20,0	20,0	10,0	0,0	5	2	2	1	0	10	2
3	0,0	30,0	30,0	30,0	10,0	0	3	3	3	1	10	1
4	30,0	70,0	0,0	0,0	0,0	3	7	0	0	0	10	3
5	33,3	22,2	33,3	11,1	0,0	3	2	3	1	0	9	1
6	11,1	33,3	33,3	22,2	0,0	1	3	3	2	0	9	1
7	0,0	10,0	70,0	0,0	20,0	0	1	7	0	2	10	3
8	11,1	0,0	11,1	44,4	33,3	1	0	1	4	3	9	2
9	11,1	55,6	22,2	11,1	0,0	1	5	2	1	0	9	2
10	30,0	20,0	50,0	0,0	0,0	3	2	5	0	0	10	2
11	40,0	20,0	40,0	0,0	0,0	4	2	4	0	0	10	2
12	55,6	44,4	0,0	0,0	0,0	5	4	0	0	0	9	2
13	50,0	20,0	20,0	0,0	10,0	5	2	2	0	1	10	2
14	33,3	55,6	11,1	0,0	0,0	3	5	1	0	0	9	2
partition	26,8	30,3	26,1	12,0	4,9	38	43	37	17	7	142	-

	percentage by of nodes theme					number of nodes by theme						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
1	22,2	22,2	22,2	33,3	0,0	2	2	2	3	0	9	0
2	50,0	20,0	30,0	0,0	0,0	5	2	3	0	0	10	2
3	0,0	20,0	40,0	30,0	10,0	0	2	4	3	1	10	1
4	30,0	60,0	10,0	0,0	0,0	3	6	1	0	0	10	2
5	33,3	44,4	0,0	22,2	0,0	3	4	0	2	0	9	1
6	11,1	0,0	66,7	22,2	0,0	1	0	6	2	0	9	2
7	0,0	50,0	20,0	10,0	20,0	0	5	2	1	2	10	2
8	11,1	0,0	22,2	33,3	33,3	1	0	2	3	3	9	1
9	11,1	55,6	22,2	11,1	0,0	1	5	2	1	0	9	2
10	30,0	20,0	50,0	0,0	0,0	3	2	5	0	0	10	2
11	40,0	30,0	30,0	0,0	0,0	4	3	3	0	0	10	2
12	55,6	11,1	33,3	0,0	0,0	5	1	3	0	0	9	2
13	50,0	10,0	30,0	0,0	10,0	5	1	3	0	1	10	2
14	33,3	33,3	0,0	33,3	0,0	3	3	0	3	0	9	1
partition	26,8	26,1	29,6	12,7	4,9	38	37	42	18	7	142	-

Table 5.9: *Experiment A - Partition A3*

	percentage by of nodes period					number of nodes by period						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	22,2	33,3	22,2	22,2	0,0	2	3	2	2	0	9	0
1	33,3	55,6	11,1	0,0	0,0	3	5	1	0	0	9	2
2	60,0	20,0	10,0	10,0	0,0	6	2	1	1	0	10	2
3	22,2	0,0	22,2	55,6	0,0	2	0	2	5	0	9	2
4	22,2	77,8	0,0	0,0	0,0	2	7	0	0	0	9	3
5	10,0	20,0	50,0	10,0	10,0	1	2	5	1	1	10	2
6	30,0	10,0	30,0	0,0	30,0	3	1	3	0	3	10	1
7	40,0	20,0	40,0	0,0	0,0	4	2	4	0	0	10	2
8	66,7	33,3	0,0	0,0	0,0	6	3	0	0	0	9	2
9	0,0	33,3	33,3	33,3	0,0	0	3	3	3	0	9	1
10	0,0	66,7	22,2	11,1	0,0	0	6	2	1	0	9	2
11	0,0	10,0	60,0	20,0	10,0	0	1	6	2	1	10	2
12	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
13	40,0	10,0	0,0	50,0	0,0	4	1	0	5	0	10	2
14	70,0	0,0	20,0	10,0	0,0	7	0	2	1	0	10	3
partition	29,6	28,2	23,9	14,8	3,5	42	40	34	21	5	142	-

	percentage by of nodes theme					number of nodes by theme						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	22,2	0,0	55,6	22,2	0,0	2	0	5	2	0	9	2
1	33,3	33,3	0,0	33,3	0,0	3	3	0	3	0	9	1
10	0,0	66,7	22,2	11,1	0,0	0	6	2	1	0	9	2
11	0,0	20,0	50,0	20,0	10,0	0	2	5	2	1	10	2
12	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
13	40,0	20,0	10,0	30,0	0,0	4	2	1	3	0	10	1
14	70,0	20,0	0,0	10,0	0,0	7	2	0	1	0	10	3
2	60,0	20,0	20,0	0,0	0,0	6	2	2	0	0	10	2
3	22,2	0,0	11,1	66,7	0,0	2	0	1	6	0	9	2
4	22,2	66,7	11,1	0,0	0,0	2	6	1	0	0	9	2
5	10,0	50,0	10,0	20,0	10,0	1	5	1	2	1	10	2
6	30,0	30,0	10,0	0,0	30,0	3	3	1	0	3	10	1
7	40,0	20,0	30,0	10,0	0,0	4	2	3	1	0	10	1
8	66,7	0,0	33,3	0,0	0,0	6	0	3	0	0	9	2
9	0,0	22,2	66,7	11,1	0,0	0	2	6	1	0	9	2
partition	29,6	25,4	26,1	15,5	3,5	42	36	37	22	5	142	-

Table 5.10: *Experiment A - Partition A4*

	percentage by of nodes period					number of nodes by period						
Cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	27,3	72,7	0,0	0,0	0,0	3	8	0	0	0	11	3
1	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
2	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
3	22,2	22,2	44,4	0,0	11,1	2	2	4	0	1	9	1
4	30,0	20,0	10,0	20,0	20,0	3	2	1	2	2	10	1
5	20,0	30,0	20,0	30,0	0,0	2	3	2	3	0	10	1
6	0,0	11,1	33,3	11,1	44,4	0	1	3	1	4	9	1
7	11,1	0,0	33,3	55,6	0,0	1	0	3	5	0	9	2
8	55,6	44,4	0,0	0,0	0,0	5	4	0	0	0	9	2
9	22,2	22,2	33,3	22,2	0,0	2	2	3	2	0	9	0
10	33,3	33,3	33,3	0,0	0,0	3	3	3	0	0	9	1
11	0,0	66,7	22,2	11,1	0,0	0	6	2	1	0	9	2
12	33,3	22,2	11,1	33,3	0,0	3	2	1	3	0	9	1
13	33,3	11,1	33,3	11,1	11,1	3	1	3	1	1	9	1
14	22,2	22,2	44,4	11,1	0,0	2	2	4	1	0	9	1
partition	23,7	29,5	27,3	13,7	5,8	33	41	38	19	8	139	-

	percentage by of nodes theme					number of nodes by theme						
cluster	0	1	2	3	n/c	0	1	2	3	n/c	total	variation
0	27,3	63,6	9,1	0,0	0,0	3	7	1	0	0	11	3
1	22,2	11,1	66,7	0,0	0,0	2	1	6	0	0	9	2
2	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
3	22,2	44,4	11,1	11,1	11,1	2	4	1	1	1	9	1
4	30,0	10,0	20,0	20,0	20,0	3	1	2	2	2	10	1
5	20,0	10,0	50,0	20,0	0,0	2	1	5	2	0	10	2
6	0,0	33,3	22,2	0,0	44,4	0	3	2	0	4	9	1
7	11,1	11,1	33,3	44,4	0,0	1	1	3	4	0	9	1
8	55,6	22,2	22,2	0,0	0,0	5	2	2	0	0	9	2
9	22,2	22,2	33,3	22,2	0,0	2	2	3	2	0	9	0
10	33,3	33,3	0,0	33,3	0,0	3	3	0	3	0	9	1
11	0,0	66,7	22,2	11,1	0,0	0	6	2	1	0	9	2
12	33,3	33,3	11,1	22,2	0,0	3	3	1	2	0	9	1
13	33,3	33,3	22,2	0,0	11,1	3	3	2	0	1	9	1
14	22,2	22,2	44,4	11,1	0,0	2	2	4	1	0	9	1
partition	23,7	29,5	28,1	12,9	5,8	33	41	39	18	8	139	-

Table 5.11: *Experiment A - Partition A5*

	percentage of nodes by period					number of nodes by period						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	20,0	20,0	0,0	60,0	0	1	1	0	3	5	1
1	16,7	16,7	22,2	44,4	0,0	3	3	4	8	0	18	2
2	40,0	40,0	20,0	0,0	0,0	2	2	1	0	0	5	1
3	27,3	9,1	45,5	9,1	9,1	3	1	5	1	1	11	2
4	15,4	38,5	46,2	0,0	0,0	2	5	6	0	0	13	2
5	34,5	17,2	31,0	10,3	6,9	10	5	9	3	2	29	3
6	31,3	58,3	6,3	4,2	0,0	15	28	3	2	0	48	11
7	50,0	0,0	12,5	31,3	6,3	8	0	2	5	1	16	3
partition	29,7	31,0	21,4	13,1	4,8	43	45	31	19	7	145	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	40,0	0,0	0,0	60,0	0	2	0	0	3	5	1
1	16,7	11,1	44,4	27,8	0,0	3	2	8	5	0	18	2
2	40,0	20,0	40,0	0,0	0,0	2	1	2	0	0	5	1
3	27,3	18,2	18,2	27,3	9,1	3	2	2	3	1	11	1
4	15,4	15,4	61,5	7,7	0,0	2	2	8	1	0	13	3
5	34,5	24,1	10,3	24,1	6,9	10	7	3	7	2	29	2
6	31,3	35,4	22,9	10,4	0,0	15	17	11	5	0	48	5
7	50,0	6,3	18,8	18,8	6,3	8	1	3	3	1	16	3
partition	29,7	23,4	25,5	16,6	4,8	43	34	37	24	7	145	-

Table 5.12: *Experiment B - Partition B1*

	percentage of nodes by period					number of nodes by period						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	50,0	0,0	50,0	0,0	0	1	0	1	0	2	1
1	15,4	38,5	46,2	0,0	0,0	2	5	6	0	0	13	2
2	25,0	25,0	25,0	0,0	25,0	1	1	1	0	1	4	0
3	33,3	0,0	50,0	16,7	0,0	2	0	3	1	0	6	1
4	29,0	58,1	6,5	6,5	0,0	9	18	2	2	0	31	7
5	40,0	22,9	11,4	25,7	0,0	14	8	4	9	0	35	4
6	42,1	31,6	21,1	0,0	5,3	8	6	4	0	1	19	3
7	22,2	13,9	38,9	22,2	2,8	8	5	14	8	1	36	3
partition	30,1	30,1	23,3	14,4	2,1	44	44	34	21	3	146	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	50,0	50,0	0,0	0,0	0	1	1	0	0	2	1
1	15,4	15,4	61,5	7,7	0,0	2	2	8	1	0	13	3
2	25,0	0,0	50,0	0,0	25,0	1	0	2	0	1	4	1
3	33,3	66,7	0,0	0,0	0,0	2	4	0	0	0	6	2
4	29,0	41,9	16,1	12,9	0,0	9	13	5	4	0	31	4
5	40,0	8,6	37,1	14,3	0,0	14	3	13	5	0	35	5
6	42,1	26,3	10,5	15,8	5,3	8	5	2	3	1	19	2
7	22,2	30,6	25,0	19,4	2,8	8	11	9	7	1	36	1
partition	30,1	26,7	27,4	13,7	2,1	44	39	40	20	3	146	-

Table 5.13: *Experiment B - Partition B2*

	percentage of nodes by period					number of nodes by period						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	66,7	0,0	0,0	33,3	0,0	2	0	0	1	0	3	1
1	0,0	0,0	40,0	40,0	20,0	0	0	2	2	1	5	1
2	25,0	25,0	16,7	33,3	0,0	3	3	2	4	0	12	1
3	45,5	18,2	36,4	0,0	0,0	5	2	4	0	0	11	2
4	28,6	7,1	28,6	35,7	0,0	4	1	4	5	0	14	2
5	15,8	15,8	21,1	42,1	5,3	3	3	4	8	1	19	2
6	33,3	53,3	13,3	0,0	0,0	15	24	6	0	0	45	9
7	34,2	36,8	18,4	7,9	2,6	13	14	7	3	1	38	4
partition	30,6	32,0	19,7	15,6	2,0	45	47	29	23	3	147	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	66,7	0,0	33,3	0,0	0,0	2	0	1	0	0	3	1
1	0,0	0,0	40,0	40,0	20,0	0	0	2	2	1	5	1
2	25,0	0,0	50,0	25,0	0,0	3	0	6	3	0	12	2
3	45,5	27,3	9,1	18,2	0,0	5	3	1	2	0	11	1
4	28,6	28,6	21,4	21,4	0,0	4	4	3	3	0	14	1
5	15,8	10,5	31,6	36,8	5,3	3	2	6	7	1	19	2
6	33,3	28,9	35,6	2,2	0,0	15	13	16	1	0	45	6
7	34,2	31,6	15,8	15,8	2,6	13	12	6	6	1	38	3
partition	30,6	23,1	27,9	16,3	2,0	45	34	41	24	3	147	-

Table 5.14: *Experiment B - Partition B3*

	percentage of nodes by period					number of nodes by period						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	100,0	0,0	0,0	0,0	0,0	1	0	0	0	0	1	0
1	33,3	0,0	66,7	0,0	0,0	2	0	4	0	0	6	2
2	22,2	11,1	55,6	11,1	0,0	2	1	5	1	0	9	2
3	30,4	47,8	21,7	0,0	0,0	7	11	5	0	0	23	4
4	30,0	20,0	40,0	10,0	0,0	3	2	4	1	0	10	1
5	33,3	61,9	0,0	4,8	0,0	7	13	0	1	0	21	5
6	27,0	10,8	27,0	32,4	2,7	10	4	10	12	1	37	3
7	33,3	38,5	17,9	7,7	2,6	13	15	7	3	1	39	5
partition	30,8	31,5	24,0	12,3	1,4	45	46	35	18	2	146	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	100,0	0,0	0,0	0,0	0,0	1	0	0	0	0	1	0
1	33,3	33,3	33,3	0,0	0,0	2	2	2	0	0	6	1
2	22,2	11,1	55,6	11,1	0,0	2	1	5	1	0	9	2
3	30,4	26,1	43,5	0,0	0,0	7	6	10	0	0	23	4
4	30,0	20,0	50,0	0,0	0,0	3	2	5	0	0	10	2
5	33,3	38,1	28,6	0,0	0,0	7	8	6	0	0	21	3
6	27,0	16,2	24,3	29,7	2,7	10	6	9	11	1	37	2
7	33,3	28,2	17,9	17,9	2,6	13	11	7	7	1	39	3
partition	30,8	24,7	30,1	13,0	1,4	45	36	44	19	2	146	-

Table 5.15: *Experiment B - Partition B4*

	percentage of nodes by period					number of nodes by period						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	0,0	0,0	100,0	0,0	0	0	0	1	0	1	0
1	22,2	44,4	33,3	0,0	0,0	2	4	3	0	0	9	1
2	11,1	11,1	33,3	0,0	44,4	1	1	3	0	4	9	1
3	25,0	62,5	6,3	6,3	0,0	4	10	1	1	0	16	4
4	66,7	27,8	0,0	5,6	0,0	12	5	0	1	0	18	5
5	25,0	3,1	40,6	25,0	6,3	8	1	13	8	2	32	4
6	18,5	29,6	29,6	22,2	0,0	5	8	8	6	0	27	1
7	42,4	48,5	3,0	6,1	0,0	14	16	1	2	0	33	7
partition	31,7	31,0	20,0	13,1	4,1	46	45	29	19	6	145	-

	percentage of nodes by theme					number of nodes by theme						
cluster	0,0	1,0	2,0	3,0	n/c	0	1	2	3	n/c	total	variation
0	0,0	0,0	0,0	100,0	0,0	0	0	0	1	0	1	0
1	22,2	22,2	55,6	0,0	0,0	2	2	5	0	0	9	2
2	11,1	22,2	22,2	0,0	44,4	1	2	2	0	4	9	1
3	25,0	37,5	18,8	18,8	0,0	4	6	3	3	0	16	1
4	66,7	11,1	22,2	0,0	0,0	12	2	4	0	0	18	5
5	25,0	18,8	28,1	21,9	6,3	8	6	9	7	2	32	1
6	18,5	11,1	55,6	14,8	0,0	5	3	15	4	0	27	5
7	42,4	36,4	18,2	3,0	0,0	14	12	6	1	0	33	5
partition	31,7	22,8	30,3	11,0	4,1	46	33	44	16	6	145	-

Table 5.16: *Experiment B - Partition B5*

Chapter 6

Conclusions and Perspectives

In this chapter, we present a short overview of the dissertation, highlighting the contributions of every chapter to the work carried in this dissertation. Then, the chapter concludes with some ideas for future research on extended generalized blockmodeling.

The main goal of this dissertation is to present new approach named extended generalized blockmodeling for binary networks, and in order to establish a common language between the reader and this document, in chapter 1, we review basic notions such as network, cluster and clustering, block and ideal block, and the type of network measures. The most important contribution of this chapter is the discussion around the motivation and objectives and the definition basic notions

To put our approach into perspective, in chapter 2 we present the state-of-the-art of social network analysis and blockmodeling. In this chapter the fundamental ideas of blockmodeling framework for binary networks as structural equivalence and regular equivalence are presented, and also present a list of usual steps for solving a clustering problem.

Chapter 2, also presents an extended study accross the several blockmodeling procedures, and also propose to use Batagelj et al. (1992) classification, that divide blockmodeling procedures into two classes: direct and indirect approaches. The pros and cons of indirect and direct approaches are also reviewed. The most important contribution of this chapter, is the definition of generalized blockmodeling and introduction to the extended generalized blockmodeling, that is the most relevant representative of the direct approach for the dissertation.

Chapter 3 presents the first combinatorial optimization model to the extended generalized blockmodeling, and its main contribution to the chapter. We present the data of our problem, decision variables and objective function used to evaluate the quantitative criterions imposed by constraints, that express the limitations on decisions. We also provide numerical experiments that show the limit of the ILP model to networks with a maximum of 20 nodes.

In chapter 4, we propose the first non-exact approach to generalized extended block

modeling, based on the VNS algorithm as an alternative to the ILP model, main contribution of this thesis. In this chapter, the limitations of the ILP model is briefly described. The chapters also present some other heuristics and variants of the problem. The chapter ends with analysis of our proposed VNS heuristic and numerical results.

Finally in chapter 5, we evaluate the use of the extended generalized blockmodeling approach in the field of Natural Language Processing (NLP) and bibliometrics through the analysis of terrorism research datasets between 1990-2006.This chapter also provides a general description of the TermWatch system. The chapter ends with a numerical results which proofs the extended generalized blockmodeling is a useful tool to study the evolution of research topics in time. The analysis of the case of study conducted in this chapter is the third major contributuion of this thesis.

On carrying out the work on this thesis, several new questions arise and some open questions were left without being answered, in order to limit the scope of this thesis. All these questions are presented above in the following.

The extended generalized blockmodeling is currently and only developed for binary networks but the extension to work with valued networks, using network measures as new block types, would be a useful line of research.

The extended generalized blockmodeling model presented in this thesis is based on integer linear programing, and as this problem is a quadratic from nature the linearisation process add to the model a huge number of variables, even for small graphs. This is one of the reasons the model is limited to small instances. Studying different techniques to solve the problem using the exact model is definitely another useful line of research.

The extended generalized blockmodeling model presented in this thesis is designed to disjointed networks, however several existing network problems are represented by networks contained overlapping clusters, such adaptatation seems to be another interesting line of research.

List of Figures

List of Tables

Bibliography

[1] Part-of-speech tagging— Wikipedia, the free encyclopedia, 2020. [Online; accessed 22-April-2020].

[2] Alireza Abbasi, Liaquat Hossain, and Loet Leydesdorff. Betweenness centrality as a driver of preferential attachment in the evolution of research collaboration networks. *Journal of Informetrics*, 6(3):403 – 412, 2012.

[3] Reka Albert, Hawoong Jeong, and Albert-Laszo Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, 2000.

[4] Tasleem Arif. The mathematics of social network analysis: Metrics for academic social networks. *International Journal of Computer Applications Technology and Research*, 4:889–893, 11 2015.

[5] S. Arumugam, Ismail Sahul Hamid, and Abraham V.M. Decomposition of graphs into paths and cycles. *Journal of Discrete Mathematics*, 2013, 04 2013.

[6] Vladimir Batagelj. Some mathematics of network analysis. 02 1991.

[7] Anne Berry, Romain Pogorelcnik, and Genevieve Simonet. An introduction to clique minimal separator decomposition. *Algorithms*, 3, 06 2010.

[8] P Brucker. On the complexity of clustering problems. *Lecture Notes in Economics and Mathematical Systems*, 157, 01 1978.

[9] Michael Brusco, Patrick Doreian, Paulette LLoyd, and Douglas Steinley. A variable neighborhood search method for a two-mode blockmodeling problem in social network analysis. *Network Science*, 1(2):191–212, 2013.

[10] Michael Brusco, Patrick Doreian, Douglas Steinley, and Cinthia B. Satornino. Multiobjective blockmodeling for social network analysis. *Psychometrika*, 78(3):498–525, Jul 2013.

[11] Michael Brusco and Douglas Steinley. A variable neighborhood search method for generalized blockmodeling of two-mode binary matrices. *Journal of Mathematical Psychology*, 51(5):325 – 338, 2007.

[12] E. Cambria and B. White. Jumping nlp curves: A review of natural language processing research [review article]. *IEEE Computational Intelligence Magazine*, 9(2):48–57, May 2014.

[13] C. Chelmis and V. K. Prasanna. Social networking analysis: A state of the art and the effect of semantics. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 531–536, Oct 2011.

[14] Hocine Cherifi, Gergely Palla, Boleslaw K. Szymanski, and Xiaoyan Lu. On community structure in complex networks: challenges and opportunities. *Applied Network Science*, 4(1), Dec 2019.

[15] Allan Clifton and Gregory D. Webster. An introduction to social network analysis for personality and social psychologists. *Social Psychological and Personality Science*, 8(4):442–453, 2017.

[16] Reuven Cohen, Keren Erez, Daniel ben Avraham, and Shlomo Havlin. Breakdown of the internet under intentional attack. *Phys. Rev. Lett.*, 86:3682–3685, Apr 2001.

[17] Mohamed Didi Biha, Bangaly Kaba, Marie-Jean Meurs, and Eric SanJuan. Graph decomposition approaches for terminology graphs. In Alexander Gelbukh and Ángel Fernando Kuri Morales, editors, *MICAI 2007: Advances in Artificial Intelligence*, pages 883–893, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[18] Patrick Doreian, Vladimir Batagelj, and Anuka Ferligoj. Generalized blockmodeling of two-mode network data. *Social Networks*, 26(1):29 – 53, 2004.

[19] Patrick Doreian, Vladimir Batagelj, and Anuska Ferligoj. *Generalized Blockmodeling*. Structural Analysis in the Social Sciences. Cambridge University Press, 2004.

[20] Guillaume Ereteo, Michel Buffa, Fabien Gandon, Patrick Grohan, Mylsne Leitzelman, and Peter Sander. A state of the art on social network analysis and its applications on a semantic web. *SDOW2008*, 01 2008.

[21] Mohamed Farrag, Laila El Fangary, Mona Nasr, and Chaimaa Salama. A comparison study of different community detection approaches and its potential applications for online networks. *International Journal of Engineering Research and Application*, 7:2248–962236, 10 2017.

[22] Nigel Fielding, Raymond M. Lee, and Grant Blank. *The Sage Handbook of Online Research Methods*. JMLR.org, 01 2008.

[23] Santo Fortunato and Claudio Castellano. *Community Structure in Graphs*, pages 490–512. Springer New York, New York, NY, 2012.

[24] Robert A. Hanneman and Mark Riddle. *Introduction to social network methods*. University of California, Riverside, Riverside, CA, 2005.

[25] Pierre Hansen and Nenad Mladenović. *Developments of Variable Neighborhood Search*, pages 415–439. Springer US, Boston, MA, 2002.

[26] Silu Huang, James Cheng, and Huanhuan Wu. Temporal graph traversals: Definitions, algorithms, and applications. *CoRR*, abs/1401.1919, 2014.

[27] Fidelia Ibekwe-Sanjuan. A linguistic and mathematical method for mapping thematic trends from texts. pages 170–174, 08 1998.

[28] Fidelia Ibekwe-Sanjuan. Terminological variation, a means of identifying research topics from texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, USA, 1998. Association for Computational Linguistics.

[29] Fidelia Ibekwe-Sanjuan, Xavier Polanco, and Eric Sanjuan. Sdoc et termwatch : deux méthodes complémentaires de cartographie de thèmes. 01 2004.

[30] Fidelia Ibekwe-Sanjuan and Eric Sanjuan. Mapping the structure of research topics through term variant clustering: the termwatch system. *Proceedings RIAO 2004 "Coupling approaches, coupling media and coupling languages for information retrieval University of Avignon, France, April 26-28, 2004.*, pages 487–503, 04 2004.

[31] Skinner James, Edwards Allan, and Corbett Benjamin. *Research methods for sport management*. Routledge Abingdon, Oxon,New York, NY, 2015.

[32] Chan Jeffrey, Lam Samantha, and Hayes Conor. Generalised blockmodelling of social and relational networks using evolutionary computing. *Social Network Analysis and Mining*, 4(1), 2014.

[33] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

[34] Micheli Knechtel, Philippe Michelon, Serigne Gueye, and Luis Satoru Ochi. A neighborhood exploration approach with multi-start for extend generalized blockmodeling. *Electronic Notes in Discrete Mathematics*, 66:63 – 70, 2018. 5th International Conference on Variable Neighborhood Search.

[35] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 78 4 Pt 2:046110, 2008.

[36] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.

[37] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Phys. Rev. Lett.*, 86:3200–3203, Apr 2001.

[38] Silvia Portugal. What makes social networks move? an analysis of norms and ties*. *RCCS Annual Review*, 12 2009.

[39] Les Proll. Ilp approaches to the blockmodel problem. *European Journal of Operational Research*, 177(2):840 – 850, 2007.

[40] A Rios, A Göhring, and Martin Volk. A quechua-spanish parallel treebank. In *7th Conference on Treebanks and Linguistic Theories*, 2009.

[41] Eric SanJuan. Termwatch ii: Unsupervised terminology graph extraction and decomposition. In Ana Fred, Jan L. G. Dietz, Kecheng Liu, and Joaquim Filipe, editors, *Knowledge Discovery, Knowledge Engineering and Knowledge Management*, pages 185–199, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[42] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees, 1994.

[43] Robert E. Tarjan. Decomposition by clique separators. *Discrete Mathematics*, 55(2):221 – 232, 1985.

[44] Zoltan Toroczkai. Complex networks the challenge of interaction topology. *Los Alamos Sci.*, 29, 01 2005.

[45] Stanley Wasserman and Katherine Faust. *Social Network Analysis in the Social and Behavioral Sciences*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994.

[46] Rudolf Wille. *Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies*, pages 1–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[47] Dietmar Wolfram. Bibliometrics, information retrieval and natural language processing: Natural synergies to support digital library research. In *Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL)*, pages 6–13, June 2016.

Acronyms

VNS	Variable Neighborhood Search
ILP	Integer Liner Programming
GA	Genetic Algorithms
LFR	Lancichinetti–Fortunato–Radicchi
RH	Relocation Heuristic
TS	Tabu Search
NP	Nondeterministic Polynomial Time
MWT	Multi-Word Term
FCA	Formal Concept Analysis
NLP	Natural Language Processing
DM	Data Mining
POS	Part-of-speech
NPs	Noun Phrases
DL	Digital Libraries
WoS	Web of Science
CPCL	Clustering by Preferential Clustered Link