UNIVERSITY SORBONNE PARIS CITÉ

DOCTORAL THESIS

Attributed Network Clustering : Application to recommender systems

Author:
Issam FALIH

Reviewers:
Martin ATZMUELLER
Christine LARGERON
Seiichi OZAWA

Examiners:
Eric JANVIER
Djamel Abdelkader ZIGHED

Supervisors:
Younès BENNANI (Director)
Nistor GROZAVU (co-supervisor)
Rushed KANAWATI (co-supervisor)

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Computer Science

in the

Paris 13 University
Laboratoire Informatique de Paris Nord

8 March, 2018

" If we knew what it was we were doing, it would not be called research, would it?
– Albert Einstein."

University Sorbonne Paris Cité

Abstract

Faculty Name
Laboratoire Informatique de Paris Nord

Doctor of Computer Science

Attributed Network Clustering : Application to recommender systems

by Issam FALIH

In complex networks analysis field, much effort has been focused on identifying graphs communities of related nodes with dense internal connections and few external connections. In addition to node connectivity information that are mostly composed by different types of links, most real-world networks contains also node and/or edge associated attributes which can be very relevant during the learning process to find out the groups of nodes i.e. communities. In this case, two types of information are available: graph data to represent the relationship between objects and attributes information to characterize the objects i.e nodes.

Classic community detection and data clustering techniques handle either one of the two types but not both. Consequently, the resultant clustering may not only miss important information but also lead to inaccurate findings. Therefore, various methods have been developed to uncover communities in networks by combining structural and attribute information such that nodes in a community are not only densely connected, but also share similar attribute values. Such graph-shape data is often referred to as *attributed graph*.

This thesis focuses on developing algorithms and models for attributed graphs. Specifically, I focus in the first part on the different types of edges which represent different types of relations between vertices. I proposed a new clustering algorithms and I also present a redefinition of principal metrics that deals with this type of networks. Then, I tackle the problem of clustering using the node attribute information by describing a new original community detection algorithm that uncover communities in node attributed networks which use structural and attribute information simultaneously.

At last, I proposed a collaborative filtering model in which I applied the proposed clustering algorithms.

Keywords : Community Detection, Attributed Network, Multiplex, Clustering, Recommendation system, Collaborative filtering

University Sorbonne Paris Cité

Abstract

Faculty Name
Laboratoire Informatique de Paris Nord

Doctor of Computer Science

Attributed Network Clustering : Application to recommender systems

by Issam FALIH

Au cours de la dernière décennie, les réseaux (les graphes) se sont révélés être un outil efficace pour modéliser des systèmes complexes. La problématique de détection de communautés est une tâche centrale dans l'analyse des réseaux complexes. La majeur partie des travaux dans ce domaine s'intéresse à la structure topologique des réseaux. Cependant, dans plusieurs cas réels, les réseaux complexes ont un ensemble d'attributs associés aux nœuds et/ou aux liens. Ces réseaux sont dites : *réseaux attribués*. Mes activités de recherche sont basées principalement sur la détection des communautés dans les réseaux attribués.

Pour aborder ce problème, on s'est intéressé dans un premier temps aux attributs relatifs aux liens, qui sont un cas particulier des réseaux multiplexes. Un multiplex est un modèle de graphe multi-relationnel. Il est souvent représenté par un graphe multi-couches. Chaque couche contient le même ensemble de nœuds mais encode une relation différente.

Dans mes travaux de recherche, nous proposons une étude comparative des différentes approches de détection de communautés dans les réseaux multiplexes. Cette étude est faite sur des réseaux réels.Nous proposons une nouvelle approche centrée "graine" pour la détection de communautés dans les graphes multiplexes qui a nécessité la redéfinition des métriques de bases des réseaux complexes au cas multiplex.

Puis, nous proposons une approche de clustering dans les réseaux attribués qui prends en considération à la fois les attributs sur les nœuds et sur les liens.

La validation de mes approches a été faite avec des indices internes et externes, mais aussi par une validation guidée par un système de recommandation que nous avons proposé et dont la détection de communautés est sa tâche principale. Les résultats obtenus sur ces approches permet d'améliorer la qualité des communautés détectés en prenant en compte les informations sur les attributs du réseaux. De plus, nous offrons des outils d'analyse des réseaux attribués sous le langage de programmation R.

Mot clés : Détection de communautés, Réseaux Attribués, Réseaux multiplexes, Système de recommandation, Apprentissage non-supervisé.

Acknowledgements

First and foremost, I want to thank my director Prof. *Younès Bennani* for supervising my thesis and for giving me the opportunity to work on this domain. Many thanks also go to my supervisor Dr. *Rushed Kanawati* who has been a perfect guide during this research. I offer my sincere thanks to my co-supervisor Dr. *Nistor Grozavu* for being very kind and supportive throughout these years of research. He always took time for guiding and helping me whenever I needed. Nistor and Rushed were always available and were willing to help me every time I had a question about scientific matters and also with every document I wrote.

Thanks to all my friends and colleagues at LIPN, for all the interesting discussions and the support inside our group. I enjoyed working in such a friendly atmosphere. Special thanks to Dr. *Basarab Matei* who supported me and encouraged me, his guidance during the final preparations of this thesis is overwhelming.

In the end, I would like to thank my entire family for being with me through thick and thin.

So, thank you all, all you made this an unforgettable three years journey of academic and life experience.

Issam Falih
January, 2018

Contents

List of Figures

List of Tables

List of Symbols

$G(\mathcal{V}, E)$	Undirected graph.
$\mathcal{V}$	Set of vertices of G.
E	Set of links of G.
A	Adjacency matrix of G.
d_v	Degree of node v
$n = \|\mathcal{V}\|$	Nodes number
$m = \|E\|$	Edges number
$\Gamma(v) = \{u \in \mathcal{V} : (u, v) \in E\}$	Neighbor's of node v
$\mathcal{CC}$	Clustering coefficient of the graph G
$A^{[k]}$	Adjacency matrix of layer k
$d_u^{[k]}$	Degree of node u in layer k
$d_u^{tot} = \sum_{s=1}^{\alpha} d_u^{[s]}$	Total degree in all layers of node u
$m^{[k]}$	Edges number in layer k
$\Gamma(v)^{[k]} = \{u \in V : (u, v) \in E_k\}$	Neighbor's of node v in layer k
$\Gamma(v)^{tot} = \cup_{s \in \{1,\ldots,\alpha\}} \Gamma(v)^{[s]}$	Neighbors of node v in all α layers
$SPath^{[k]}(u, v)$	Shortest path length between nodes u and v in layer k
C_{uv}^{kl}	Inter-slice link weight between node u, v in slices k, l.
x, y	Patterns, observations, samples.
$\mathcal{T}$	Set of features.
$\mathcal{A}$	Attribute matrix.
$\mathcal{P}$	Partition
C_i	Cluster i of a partition
k	number of clusters
δ_G	Density of graph G

Dedicated to my mother . . .

Introduction

Contents

0.1 Context

A great amount of data is now available in science, business, industry, and many other areas, due to the rapid advances in computerization and digitalization techniques. Such data may provide a rich resource for knowledge discovery and decision support. In order to understand, analyse, and make use of the huge amount of data, a multidisciplinary approach, data mining, is proposed to meet the challenge. Data mining is the process of identifying interesting patterns from large databases [139].

Data mining is the core part of the Knowledge Discovery in Database (KDD) process as shown in figure 1. The KDD process may consist of the following steps: data selection, data cleaning, data transformation, pattern searching (data mining), finding presentation, finding interpretation, and evaluation. Data mining and KDD are often used interchangeably because data mining is the key part of the KDD process [53]. There exist several data mining tasks leading to different kinds of result patterns, e.g. clustering, classification or frequent pattern mining. In the last step, these patterns can then be evaluated and visualized in order to be more easily interpretable.

The aim of data mining is to extract knowledge from large data sets by combining methods from statistics and machine learning with database management. The data size can be measured in two dimensions, the number of features and the number of observations. Both dimensions can take very high values, which can cause problems during the exploration and analysis of the data set. Models and tools are therefore required to process data for an improved understanding.

Topological learning is a recent direction in Machine Learning which aims to develop methods grounded on statistics to recover the topological invariants from the observed data points. Most of the existed topological learning approaches are based on graph theory or graph-based clustering methods. The topological learning is one of the most known technique which allows clustering and visualization simultaneously. These clusters can be represented by more concise information than the brutal listing of their patterns, such as their gravity center or different statistical moments. As expected, this information is easier to manipulate than the original data points.

Many real-world systems are modeled as networks of interacting actors (e.g. users, authors, documents, scientific papers, items, proteins, etc.). Frequently cited examples include the cell described as a complex network of chemicals connected

by chemical reactions; the Internet is a complex network of routers and computers linked by various physical or wireless links; fads and ideas spread on social network, whose nodes are human beings and whose edges represent various social relationships as Friendship; the World Wide Web is an enormous virtual network of Web pages connected by hyperlinks [6]. However, real-world systems are often associated with additional information describing nodes i.e. actors and/or the relation between them. This gives rise to the *attributed networks*, i.e. networks where the nodes are associated with an number of attribute and nodes in the network are linked with different types of relations. For example, in social networks, edge attributes represent the different types of relationship (friendship, collaboration, family, etc) among people while vertex attribute describe the role or the personality of a person (age, gender, profession, etc.). Another example is, in a bibliography network, a vertex may represents an author, vertex attributes describe the author (such as the area of interest, number of publications, etc), while edge attribute represents relationships among authors (such as co-authorship, citation, etc.).

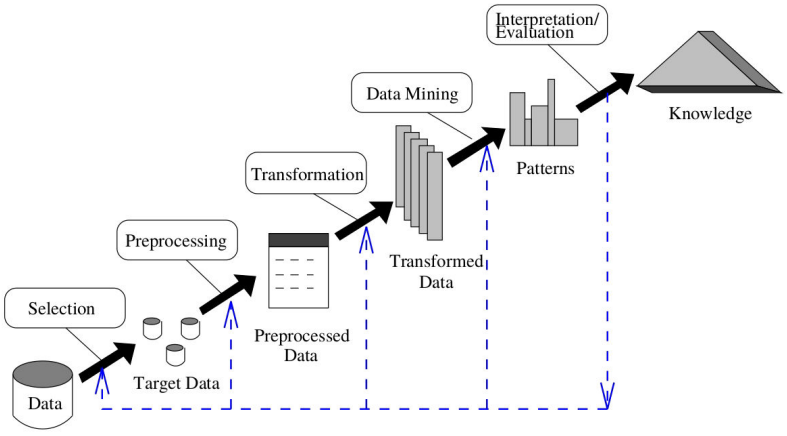


FIGURE 1: The knowledge discovery process [41]

This thesis focuses on the data mining task of clustering in the attributed network, i.e. grouping objects into clusters such that objects located in the same group are similar to each other, while objects located in different groups are dissimilar.

Most existing clustering methods were developed for vector data. In traditional vector clustering methods, the similarity between two objects is defined based on the similarity of the vertices in all the attributes/dimensions.

Besides the algorithms for vector data, clustering algorithms for graph data also exist. The basic aim of these approaches is to detect clusters of vertices in a graph such that the vertices in a cluster are densely connected in the graph. This task is often denoted as graph clustering or community detection. While various clustering approaches can handle either vector data or graph data, in many applications data of both types is available simultaneously. Graph clustering and community detection have traditionally focused on graphs without attributes, with the notable exception of edge weights. However, these models only provide a partial representation of real social systems, that are thus often described using node attributes, representing features of the actors, and edge attributes, representing different kinds of relationships among them. We refer to these models as attributed graphs. Consequently, existing graph clustering methods have been recently extended to deal with node and edge attributes.

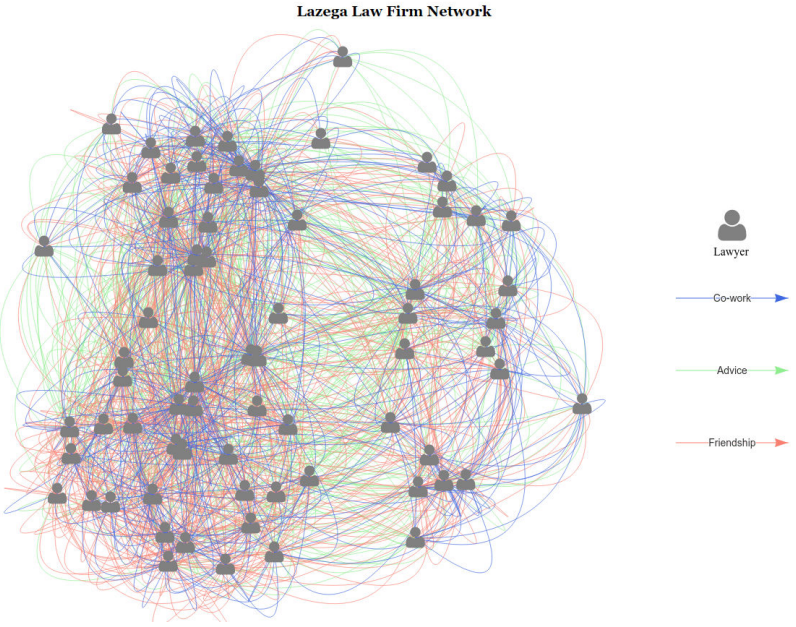


FIGURE 2: Example of attributed network

An increasing number of applications on the World Wide Web rely on combining link and content analysis (in different ways) for subsequent analysis and inference. For example, search engines, like Google, Bing and Yahoo! typically use content and link information to index, retrieve and rank web pages. Social networking sites like Twitter, Flickr and Facebook, as well as the aforementioned search engines, are increasingly relying on fusing content (pictures, tags, text) and link information (friends, followers, and users) for deriving actionable knowledge (e.g. marketing and advertising).

In this thesis, we introduce clustering approaches for graphs with vertex attributes and graphs with edges attributes. In the following, we give an overview over the contributions in section 1.1, we present the list of publications in section 1.2 and the structure of this thesis is given in the Section 1.3.

0.2 Contribution

The goal of this thesis is to propose new methods to improve cluster analysis of attributed network, by introducing new approaches for clustering of graphs with additional attribute data information. This section provides a short overview of the contributions and the structure of this thesis as well as information about preliminary publications of parts of the thesis content.

- **Clustering graphs with edges attributes** In first part of this thesis, we consider edge attributed network i.e network with different type of relations over the set of vertices. We have proposed *muxLicod* approach, a clustering algorithm that deal with this type of networks.

- **Clustering graphs with vertex attributes** The second part address the problem of node attributed network clustering. We proposed a new algorithm that learns the node attribute information and the topological structure of the network simultaneously.

- **Recommender system** The third proposition is a recommender system based on clustering and in which we validate the results of the proposed algorithms.

0.3 List of Publications

- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2017), « ANCA: Attributed Network Clustering Algorithm », in Proc. Complex Networks'17, The 6th International Conference on Complex Networks and Their Applications, November 29 - December 01 2017, Lyon, France.
- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2017), « ANCA: Attributed Network Clustering Algorithm », in Proc. MARAMI'17, 18-20 Octobre 2017, La Rochelle, France.
- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2017), « Multiplex Network Clustering based Collaborative Filtering », in Proc. ISI'17 : 61st International Statistical Institute World Statistics Congress, 16-21 July, Marrakech, Kingdom of Morocco.
- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2016), «Clustering dans les graphes attribués», in Proc. AAFD& SFC'16: Conférence Internationale Francophone sur la Science des Données 2016, 22-26 mai 2016, Marrakech, Kingdom of Morocco.
- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2016), «Attributed network clustering based recommendation approach», in Proc. MARAMI/JFGG'16 12-14 October 2016 Cergy, France.
- SELLAMI S., FALIH I., AUVRAY T., KANAWATI R. (2016), « Analyse des réseaux d'interaction entre entreprises cotées en bourses», in Proc MARAMI/JFGG'16 12-14 October 2016 Cergy, France.
- FALIH I., GROZAVU N., KANAWATI R., BENNANI Y. (2015), «A Recommendation System Based on Topological Learning», in Lecture Notes in Computer Science, LNCS Springer, Proc of ICONIP'15 : 22th International Conference on Neural Information Processing, 09-12 November 2015, Istanbul, Turkey.
- FALIH I., KANAWATI R. (2015), «MUNA: A Multiplex network analysis Library», in Proc. ASONAM'15: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 25-28 August 2015, Paris, France.
- FALIH I., KANAWATI R., GROZAVU N., BENNANI Y. (2015), «Approches de clustering pour la recommandation», in Proc. SFC'15: XXII rencontre de la société francophone de classification, 9-11 september 2015, Nantes, France.
- FALIH I., HMIMIDA M., KANAWATI R. (2015), «Une approache centrée graine pour la détection de communautés dans les réseaux multiplexes», in Proc, EGC'15: Conférence francophone sur l'Extraction et la Gestion de Connaissance, 27-30 Jan. 2015 Luxembourg, Luxembourg.
- FALIH I., HMIMIDA M., KANAWATI R. (2014), «Détection de communautés dans les réseaux multiplexes: étude comparative», in Proc. JFGG'14: 5ième Journée thématique: Fouille de grands graphs, 15-17 october 2014 Paris, France.

- FALIH I., HMIMIDA M., KANAWATI R. (2014), « Community detection in multiplex network: a comparative study», in Proc. ECCS'14: European conference on complex systems Satellite workshop on multiplex networks 24 september 2014, Lucca, Italie.

0.4 Reading guide

This report is organized as follows.

- **Chapter : Data Clustering** presents the context of this research work. It provides a description about complex networks clustering and vectorial data clustering. It also presents the different quality measures to validate the clustering results.
- **Chapter : multi-relational network clustering (or multiplex)** provides basic definition of the edge attributed network clustering problem and we provide a quick survey on existing approaches. Additionally, in this chapter is introduced and formalized the concept of multiplex network. It also includes our first contribution that uses edge attribute network in the clustering process.
- **Chapter : Attributed Network Clustering** presents a new community detection approach which uses the topological structural of the network and node attribute information to produce a partition with clusters of nodes using the topological structure and of their attribute information.
- **Chapter : Application: Recommender system** provides an new recommender system based on clustering attributed networks.
- **Chapter : Conclusion and outlook.** This chapter concludes the thesis and also provides future directions in this research area. The main contribution of the thesis is also highlighted in this chapter.

Some additional information is provided in the appendices.

Chapter 1

Data clustering : State of the art

Contents

In this chapter a relevant literature review of the various topics that fall under data clustering and community detection are discussed. The first section discusses classical clustering notations and formulations, different similarity criteria and finally some of the well known clustering algorithms. The following section discusses existed approaches for community detection and the section 1.4 focus on different methods to validate the performance of the clustering result.

1.1 Introduction

One of the most used techniques among many others in the data mining field is the clustering. The aim of thesis methods is to synthesize and summarize huge amounts of data by splitting it into small and homogeneous clusters such that the data (observations) inside the same cluster are more similar to each other compared to the observations which belongs to other clusters. This definition assumes that there exists a well defined clustering quality measure that quantifies how much homogeneous are the obtained clusters. Although there is no a consensus about what is a good quality measure, but instead it may vary from application to another and from a data set to another one.

The input data of the clustering methods is non-labeled, therefore the groups are formed using only the properties of each element of the data set.

In general, the data sets used for clustering contains points (i.e. observations) in $\mathbb{R}^n$, but, despite they may be composed of other types of information such as categorical data or nonnumeric values, the typical representation of each element, or pattern, is a vector. In [33], the following types of data for clustering are considered:

- Qualitative variables.
- Quantitative variables (continuous).
- Nominal and ordinal variables.

Along to each data type it is required to define a set of similarity or dissimilarity measures to test the quality of a partition. Those similarity measures must satisfy some properties in order to be used also as a distance measure. These properties, presented by [61], are:

Given x and y two observations from a data set, a proximity measure, denoted by $d(x, y)$ must satisfy:

1. (a) For dissimilarity: $d(x, x) = 0, \forall x$.

   (b) For similarity: $d(x, x) \geqslant max_y d(x, y), \forall x$
2. $d(x, y) = d(y, x), \forall x, y$
3. $d(x, y) \geqslant 0, \forall x, y$
4. $d(x, y) = 0$ iff, $x = y$.
5. $d(x, y) \leqslant d(x, z) + d(z, y), \forall x, y, z$

The proximity measures between data observations should be defined in function of the type of data, i.e., binary, quantitative, nominal and ordinal variables according to [33, 36].

♣ **Quantitative variables :** It is possible to define a distance matrix such that x and y are two data vectors and the attributes are numerical (continuous). x_{a_t} and y_{a_t} are the value of variable (feature) a_t for x and y respectively.

- **Minkowski distance:** is the most common proximity index [61], is defined by:

$$d(x, y) = \sqrt[\lambda]{\sum_{t=1}^{\mathcal{T}} |x_{a_t} - y_{a_t}|^\lambda} \tag{1.1}$$

  where $\lambda \geqslant 1$, $\mathcal{T}$ is the number of features and λ is a parameter for changing the way in which the measure is taken. This measure, and it's derivations, satisfy the properties 4 and 5 stated above.
- **Manhattan distance:** Is obtained when $\lambda = 1$ and is defined by:

$$d(x, y) = \sum_{t=1}^{\mathcal{T}} |x_{a_t} - y_{a_t}| \tag{1.2}$$

- **Euclidean distance:** this measure is obtained when $\lambda = 2$ and is defined by:

$$d(x,y) = \sqrt{\sum_{t=1}^{\mathcal{T}}(x_{a_t} - y_{a_t})^2} \tag{1.3}$$

- **Maximum:** measure is obtained when $\lambda \to \infty$ and is defined by:

$$d(x,y) = \max_{1 \leqslant t \leqslant \mathcal{T}} |x_{a_t} - y_{a_t}| \tag{1.4}$$

- **Canberra measure:** this metric is similar to the Manhattan distance, but each term is divided by the sum of the absolute values of each component. Consequently, this metric is sensible to values close to zero [33]. It is defined by:

$$d(x,y) = \sum_{t=1}^{\mathcal{T}} \frac{|x_{a_t} - y_{a_t}|}{(|x_{a_t}| + |y_{a_t}|)} \tag{1.5}$$

- **Squared Euclidean distance:** Is defined by

$$d(x,y) = \sum_{t=1}^{\mathcal{T}}(x_{a_t} - y_{a_t})^2 \tag{1.6}$$

- **Average Euclidean distance:** Is defined by

$$d(x,y) = \sqrt{\frac{1}{\mathcal{T}}\sum_{t=1}^{\mathcal{T}}(x_{a_t} - y_{a_t})^2} \tag{1.7}$$

♣ **Nominal and ordinal variables:** these are variables for which there are more than two states or categories [33]. If those categories are ordered, the variables are called ordinal, otherwise are nominal. One measure would consist in summing the contributions of each category over all the variables. This is done by defining disagreements indices [33] between each pair of categories as $\delta_{klm} \geqslant 0$, where l and m are categories of the $k-th$ variable. In the case of nominal variables, $\delta_{klm} = 1$ if $l \neq m$ and $\delta_{klm} = 1$ otherwise.

♣ **Binary variables:** these are variables can have only two states, e.g., (1, 0) or (TRUE, FALSE). Using the matrix representation proposed by [61, 36], the p possible values for two observations x_i and x_j are:

x_i \ x_j	1	0
1	S_{11}	S_{10}
0	S_{01}	S_{00}

Thus $p = S_{00} + S_{01} + S_{10} + S_{11}$. Note that S_{11} and S_{00} are the number of agreements between the two observations x_i and x_j.

Using the values of p it is possible to define the following measures:

- **Simple Matching Coefficient:** weights the number of agreements which is expressed as:

$$s(x,y) = \frac{S_{11} + S_{10}}{S_{11} + S_{01} + S_{10} + S_{00}} \tag{1.8}$$

- **Jaccard Coefficient:** weights only the value equals to 1 of the patterns. This means that, it only takes into account the values of the patterns which match 1 to 1, but discarding the 0 to 0 matches. This coefficient can be generalized as the size of intersection divided by the size of the union of the compared patterns:

$$s(x,y) = \frac{S_{11}}{S_{11} + S_{01} + S_{10}} \tag{1.9}$$

- **Dice-Sorensen Coefficient:** this coefficient was first used to compare the ecological association between species by [36]. But it can be generalized to different types of data. The coefficient is given by:

$$s(x,y) = \frac{S_{11}}{2S_{11} + S_{01} + S_{10}} \tag{1.10}$$

- **Yule Coefficient:**

$$s(x,y) = \frac{S_{11}S_{00} - S_{01}S_{10}}{S_{11}S_{00} + S_{01}S_{10}} \tag{1.11}$$

- **Pearson Coefficient:**

$$s(x,y) = \frac{S_{11}.S_{00} - S_{01}.S_{10}}{\sqrt{(S_{11} + S_{01})(S_{11} + S_{10})(S_{01} + S_{00})(S_{10} + S_{00})}} \tag{1.12}$$

- **Kulzinsky Coefficient:**

$$s(x,y) = \frac{S_{11}}{S_{01} + S_{10}} \tag{1.13}$$

- **Rogers-Tanimoto Coefficient:**

$$s(x,y) = \frac{S_{11} + S_{00}}{S_{11} + 2(S_{01} + S_{10}) + S_{00}} \tag{1.14}$$

In general, similarities can be translated into dissimilarities, or even be treated as distances, by doing $(1 - s(x,y))$, where $s(x,y) \in [0,1]$ is a similarity measure.

Clustering techniques are very diverse and they have been continuously developed for over a half century depending upon the optimization techniques, main methodology (statistical methods, system modeling, signal processing), and application areas. These algorithms are generally classified as partitional clustering and hierarchical clustering, based on the properties of the generated clusters ([39]; [54]; [63]; [62]). Partitional clustering divides data samples into a single partition, whereas a hierarchical clustering algorithm groups data with a sequence of nested partitions (figure 1.1). Some clustering methods are summarized below.

1.1.1 Partitional Clustering

These algorithms partition the data set into k groups and then assign each point to one group according to the distance to the group's center of the cluster. Techniques in this category are known for using computational resources efficiently [160]. However, one of their drawbacks is the selection of the initial k value and the initial centroids. In the following, we will describe the most known partitional clustering.

- k**-means:** This technique presented in [4], [94] assign each point from the data set to one of the k groups according to some similarity criterion. Nowadays, this algorithm is widely used due to its computational and space-use efficiency and to its simplicity of implementation. However, it has some drawbacks: the stability of the results and the initial selection of the number k of groups. The first disadvantage is related with the fact that every run of the algorithm may return different results, even when the number of groups is the same. The second one is related with the first selection of k. Assigning a priori the number of clusters, requires some knowledge about the data set, or, at least, make some assumptions about it. The algorithm begins randomly assigning a centroid to each of the k groups, then, assign each point to the nearest centroid. Once each point has been assigned, the centroids are recalculated according to the points which belong to each one and then, the algorithm is restarted. This is made until the cluster configuration remains stable (convergence of the algorithm).

- **Entropy based categorical clustering** This technique presented by [24, 91], begins by assigning each point in the data set to one of the k defined clusters. Then, using a Monte-Carlo approach, a node is randomly selected and put into some random cluster. If that change reduces the entropy of the set, then the node is assigned to the new group, if not, the node is returned to its original group. To calculate the entropy of the partition $\mathcal{P}$ the following expression is used:

$$H(P) = \sum_{i=1}^{k} H(C_i) \tag{1.15}$$

where $H(C_i)$ is the entropy of the group C_i of the partition $\mathcal{P}$, and its given by:

$$H(C_i) = \sum_{i=1}^{n-1} \sum_{j=1}^{n} s_{ij} ln s_{ij} + (1 - s_{ij}) ln(1 - s_{ij}) \quad s_{ij} \in [0, 1] \tag{1.16}$$

Where $0 < s_{ij} < 1$ is a similarity measure between the elements i and j. This approach can easily be applied to cluster various type of data by selecting the appropriate similarity measure.

1.1.2 Agglomerative clustering

Agglomerative clustering starts with n clusters, each of which includes exactly one data point. A series of merge operations is then followed that eventually forces all objects into the same group.

- **Hierarchical clustering :** Hierarchical clustering methods impose a hierarchical structure on the data objects and their step-wise clusters, i.e. one extreme of the clustering structure is only one cluster containing all objects, the other extreme is a number of clusters which equals the number of objects. To obtain a certain number of clusters, the hierarchy is cut at the relevant depth. Hierarchical clustering is a rigid procedure, since it is not possible to re-organize clusters established in a previous step.

  As it is shown on figure 1.1 there is two types of the hierarchical clustering methods: agglomerative approach and divide approach. Divisive hierarchical

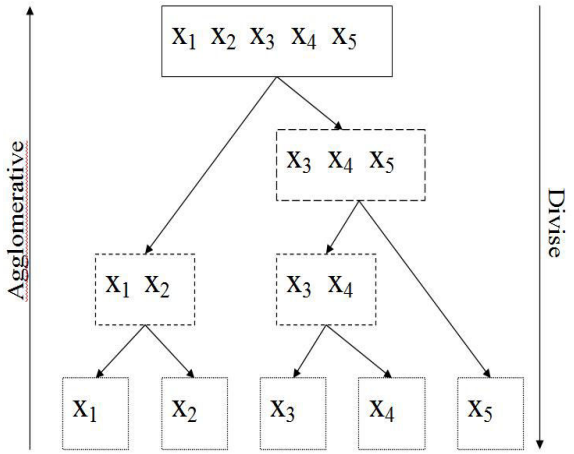


FIGURE 1.1: The principle of the Hierarchical Clustering Algorithm

clustering method starts from a cluster which contains all the data and divide this cluster until obtaining the desired clusters. Contrarily, agglomerative hierarchical clustering method starts from n clusters (n data) and will merge these clusters until obtaining a cluster containing the whole data.

The general agglomerative clustering method can be summarized by the algorithm 1.

Algorithm 1 Hierarchical Clustering Algorithm.

Input: Data set X, n - number of samples, k - number of clusters

for $i = 1$ to n **do**
  Compute the proximity matrix (usually based on the distance function) for the k clusters;
end for
for $j = 1$ to k **do**
  Compute/Search the minimal distance $d(C_i, C_j) = \min_{1 \leq m,l \leq k, m \neq l} d(C_m, C_l)$ where $d(.,.)$ is the distance function
end for

for $p = 1$ to k **do**
  Update the proximity matrix by computing the distances between the cluster C_{ij} and the other clusters;

end for
REPEAT steps 2 and 3 until only one cluster remains.

- **Self-Organizing Maps**

  The basic model proposed by Kohonen [80] consists of a discrete set $\mathcal{C}$ of cells called "map". This map has a discrete topology defined by an undirected graph, which usually is a regular grid in two dimensions.

For each pair of cells (j,k) on the map, the distance $\delta(j,k)$ is defined as the length of the shortest chain linking cells j and k on the grid. For each cell j this distance defines a neighbour cell; in order to control the neighbourhood area, we introduce a kernel positive function $\mathcal{K}$ ($\mathcal{K} \geq 0$ and $\lim_{|y|\to\infty} \mathcal{K}(y) = 0$). We define the mutual influence of two cells j and k by $\mathcal{K}_{j,k}$. In practice, as for traditional topological maps we use a smooth function to control the size of the neighbourhood as $\mathcal{K}_{j,k} = \exp(\frac{-\delta(j,k)}{T})$. Using this kernel function, T becomes a parameter of the model. As in the Kohonen algorithm, we decrease T from an initial value T_{max} to a final value T_{min}.

Let $\Re^d$ be the Euclidean data space and $E = \{\mathbf{x}_i; i = 1, \ldots, N\}$ a set of observations, where each observation $\mathbf{x}_i = (x_i^1, x_i^2, ..., x_i^d)$ is a vector in $\Re^d$. For each cell j of the grid (map), we associate a referent vector (prototype) $\mathbf{w}_i = (w_i^1, w_i^2, ..., w_i^d)$ which characterizes one cluster associated to cell i. We denote by $\mathcal{W} = \{\mathbf{w}_j, \mathbf{w}_j \in \Re^d\}_{j=1}^{|\mathcal{W}|}$ the set of the referent vectors. The set of parameter $\mathcal{W}$ has to be estimated iteratively by minimizing the classical cost function defined as follows:

$$R(\chi, \mathcal{W}) = \sum_{i=1}^{N} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{1.17}$$

where χ assigns each observation $\mathbf{x}_i$ to a single cell in the map $\mathcal{C}$. This cost function can be minimized using both stochastic and batch techniques [141].

The minimization of $R(\chi, \mathcal{W})$ is done by iteratively repeating the following three steps until stabilization. After the initialization step of prototype set $\mathcal{W}$, at each training step $(t+1)$, an observation $\mathbf{x}_i$ is randomly chosen from the input data set and the following operations are repeated:

- Each observation $(\mathbf{x}_i)$ is assigned to the closest prototype $\mathbf{w}_j$ using the assignment function defined as follows:

$$\chi(\mathbf{x}_i) = \underset{i \leq j \leq |w|}{\arg\min} \left(\|\mathbf{x}_i - \mathbf{w}_j\|^2 \right)$$

- The prototype vectors are updated using the gradient stochastic expression :

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \epsilon(t) \mathcal{K}_{j,\chi(\mathbf{x}_i)} \left(\mathbf{x}_i - \mathbf{w}_j(t) \right)$$

At the end of the learning process the algorithm provides a prototypes matrix (topological map), a neighbourhood matrix and the affectations of data to each cell (best matching unit). This information will be used in our approach in order to improve the computational time of the spectral clustering and to give more information to the obtained clusters.

- GTM: Generative Topographic Mapping GTM was proposed by Bishop et al. [11] as a probabilistic counterpart to the Self-organizing maps (SOM) [79]. GTM is defined as a mapping from a low dimensional latent space onto the

observed data space. The mapping is carried through by a set of basis functions generating a constrained mixture density distribution. It is defined as a generalized linear regression model:

$$y = y(z, W) = W\Phi(z) \tag{1.18}$$

where y is a prototype vector in the D-dimensional data space, Φ is a matrix consisting of M basis functions $(\phi_1(z), \ldots, \phi_M(z))$, introducing the non-linearity, W is a $D \times M$ matrix of adaptive weights w_{dm} that defines the mapping, and z is a point in latent space.

The standard definition of GTM considers spherically symmetric Gaussians as basis functions, defined as:

$$\phi_m(x) = \exp\left\{-\frac{\|x - \mu_m\|^2}{2\sigma^2}\right\} \tag{1.19}$$

where μ_m represents the centers of the basis functions and σ - their common width. Let $\mathcal{D} = (x_1, \ldots, x_N)$ be the data set of N data points. A probability distribution of a data point $x_n \in \Re^D$ is then defined as an isotropic Gaussian noise distribution with a single common inverse variance β:

$$\begin{aligned} p(x_n|z, W, \beta) &= \mathcal{N}(y(z, W), \beta) \\ &= \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|x_n - y(z, W)\|^2\right\} \end{aligned} \tag{1.20}$$

The distribution in x-space, for a given value of W, is then obtained by integration over the z-distribution

$$p(x|W, \beta) = \int p(x|z, W, \beta)p(z)\,dz \tag{1.21}$$

and this integral can be approximated defining $p(z)$ as a set of K equally weighted delta functions on a regular grid,

$$p(z) = \frac{1}{K}\sum_{i=1}^{K}\delta(z - z_k) \tag{1.22}$$

So, equation (1.21) becomes

$$p(x|W, \beta) = \frac{1}{K}\sum_{i=1}^{K}p(x|z_i, W, \beta) \tag{1.23}$$

For the data set $\mathcal{D}$, we can determine the parameter matrix W, and the inverse variance β, using maximum likelihood. In practice it is convenient to maximize the log likelihood, given by:

$$\begin{aligned}\mathcal{L}(W,\beta) &= \ln \prod_{n=1}^{N} p(x_n|W,\beta) \\ &= \sum_{n=1}^{N} \ln \left\{ \frac{1}{K} \sum_{i=1}^{K} p(x_n|z_i,W,\beta) \right\} \end{aligned} \tag{1.24}$$

The EM Algorithm

The maximization of (1.24) can be regarded as a missing-data problem in which the identity i of the component which generated each data point x_n is unknown. The EM algorithm for this model is formulated as follows:

The posterior probabilities, or responsibilities, of each Gaussian component i for every data point x_n using Bayes theorem are calculated in the E-step of the algorithm in this form

$$\begin{aligned} r_{in} &= p(z_i|x_n, W_{old}, \beta_{old}) \\ &= \frac{p(x_n|z_i, W_{old}, \beta_{old})}{\sum_{i'=1}^{K} p(x_n|z_i', W_{old}, \beta_{old})} \\ &= \frac{\exp\{-\frac{\beta}{2}\|x_n - W\phi(z_i)\|^2\}}{\sum_{i'=1}^{K} \exp\{-\frac{\beta}{2}\|x_n - W\phi(z_i')\|^2\}} \end{aligned} \tag{1.25}$$

As for the M-step, we consider the expectation of the complete-data log likelihood in the form

$$\mathbf{E}[\mathcal{L}_{comp}(W,\beta)] = \sum_{n=1}^{N}\sum_{i=1}^{K} r_{in} \ln\{p(x_n|z_i,W,\beta)\} \tag{1.26}$$

The parameters W and β are now estimated maximizing (1.26), so the weight matrix W is updated according to:

$$\Phi^T G \Phi W_{new}^T = \Phi^T R X \tag{1.27}$$

where, Φ is the $K \times M$ matrix of basis functions with elements $\Phi_{ij} = \phi_j(z_i)$, R is the $K \times N$ responsibility matrix with elements r_{in}, X is the $N \times D$ matrix containing the data set, and G is a $K \times K$ diagonal matrix with elements

$$g_{ii} = \sum_{n=1}^{N} r_{in} \tag{1.28}$$

The parameter β is updated according to

$$\frac{1}{\beta_{new}} = \frac{1}{ND} \sum_{n=1}^{N}\sum_{i=1}^{K} r_{in}\|x_n - W^{new}\phi(z_i)\|^2 \tag{1.29}$$

- Spectral clustering Given a set of data points $x_1, x_2, ..., x_n \ in\mathbb{R}^m$, spectral clustering first constructs an undirected graph $G = (V, E)$ represented by its adjacency matrix $W = (w_{ij})_{i,j=1}^n$, where $w_{ij} \geq 0$ denotes the similarity (affinity) between x_i and x_j. The degree matrix D is a diagonal matrix whose entries are column (or row, since W is symmetric) sums of W, $D_{ii} = \sum_j W_{ji}$. Let $L = D - W$, which is called Laplacian graph. Spectral clustering then use the top k eigenvectors of L (or, the normalized Laplacian $D^{-1/2}LD^{-1/2}$) corresponding to the k smallest eigenvalues as the low dimensional representations of the original data. Finally, k-means method is applied to obtain the clusters.

  For a data matrix $A = (a_{ij}) \in \mathbb{R}_+^{N\times M}$, the aim of the k-means clustering is to cluster the rows or the columns of A, so as to optimize the difference between $A = (a_{ij})$ and the clustered matrix revealing significant block structure. More formally, we seek to partition the set of rows $I = \{1, \ldots, N\}$ into K clusters $C = \{C_1, \ldots, C_K\}$. The partitioning naturally induce clustering index matrix $R = (r_{ik}) \in \mathbb{R}_+^{N\times K}$, defined as binary classification matrix such as $\sum_{k=1}^K r_{ik} = 1$. Specifically, we have $r_{ik} = 1$, if the row $i \in C_k$, and 0 otherwise. On the other hand, we note $S = (s_{kj}) \in \mathbb{R}_+^{K\times M}$ a reduced matrix specifying the cluster representation.

  The detection of homogeneous clusters of objects can be reached by looking for the two matrices R and S minimizing the total squared residue measure.

$$\mathcal{J}_{kmeans} = \mathcal{J}(A, RS) = ||A - RS||^2 \tag{1.30}$$

  The term RS characterizes the information of A that can be described by the cluster structures.

  Given a dataset A of P points in a space X and a $P \times P$ "similarity matrix" (or "affinity matrix") W that measures the similarity between the P points, the goal of clustering is to organize the dataset into disjoint subsets with high intra-cluster similarity and low inter-cluster similarity. Due to the high complexity of the graph construction $(O(n^2))$ and the eigen-decomposition$(O(n^3))$, it is not easy to apply spectral clustering on large-scale data sets.

- Topological Spectral Clustering (TSC) Spectral clustering method needs to construct an adjacency matrix and calculate the eigen-decomposition of the corresponding Laplacian matrix [27]. Both of these two steps are computational expensive. Then, it is not easy to apply spectral clustering on large-scale data sets. As the Spectral Clustering can not be used for large dataset due to the construction of the similarity matrix the Topological TSC use the topological clustering in order to reduce the dimension. This approach consists in two steps: i) Compute the map using the SOM algorithm (presented in section 2) and ii) use the spectral clustering on the W prototypes matrix weighted by the neighbourhood matrix (H).

  The basic idea of this method is to reduce the data size. The proposed method firstly performs SOM on the data set with a large number of p cells. Then, the traditional spectral clustering is applied on the p cells centers weighted by the neighbourhood function to give an topological view of the data distribution in clusters. The data point is assigned to the cluster as its nearest center.

  The proposed approach consists in two steps : i) Compute the map using the SOM algorithm (presented in section 2) and ii) use the spectral clustering on

Algorithm 2 Spectral Clustering trough Topological Learning

Input: n data points $x_1, x_2, ..., x_n \in \mathbb{R}^m$; Cluster number k ;
Output: k clusters;

1. Compute p prototype points ($W \in R^{p \times m}$) and neighbourhood matrix $H \in R^{p \times p}$ using SOM
2. Construct the affinity matrix $A \in R^{p \times p}$ defined by $A_{ij} = exp(-||w_i - w_j||^2/2\sigma^2)$ if $i \neq j$, and $A_{ii} = 0$
3. Construct the affinity matrix $S = A * H$
4. Define D to be the diagonal matrix whose (i,i) element is the sum of A's i-th row, and construct the matrix $L = D^{-1/2}SD^{-1/2}$
5. Find $u_1, u_2, ..., u_k$, the k largest eigenvectors of L (chosen to be orthogonal to each other in the case of repeated eigenvalues), and form the matrix $U = [u_1 u_2 ... u_k] \in R^{p \times k}$ by stacking the eigenvectors in columns
6. Form the matrix Y from U by re-normalizing each of U's rows to have unit length : $Y_{ij} = U_{ij}/(\sum_j U_{ij}^2)^{1/2}$
7. Cluster each row of Y into k clusters via k-means algorithm
8. Assign the prototypes w_i to cluster j if and only if row i of the matrix Y was assigned to cluster j.

the W prototypes matrix. This method is computationally simple and depends on the number of cells of the map and the number of clusters.

1.2 Community detection approaches

Graphs as an expressive data structure is popularly used to model structural relationships between objects in many application domains such as the web, social networks, sensor networks and telecommunication, etc. In graph representation, a community structure consists of several nodes which shows dense internal connections compared to the rest of the network. The identification of communities hidden within the structure of large network is a challenging problem which has attracted a considerable amount of interest. It has been widely studied in the literature, and there have been significant advancements with contributions from different fields. Different community detection methods are developed from various applications of specific needs which establish its own definition of community. This means that the definition of a community depends on the application domain and the properties of the graph under consideration.

We focus in this study on approaches that aim to compute a partition, or disjoint communities of a complex network. The variety of methods that have appeared in literature for detecting communities is even larger, since for each community definition there are more than one method claiming to detect the respective communities. Recent interesting survey studies on this topic can be found in [43, 142, 113, 69]. The existing approaches of clustering in simple graph can be classified into four classes:

1. Group based approaches

2. Network based approaches
3. Propagation based approaches
4. Seed-centric based approaches

We briefly review below each of these main approaches.

1.2.1 Group-based approaches

These are approaches based on identifying groups of nodes that are highly connected or share some strong connection patterns. Some relevant connection patterns are the following:

- *High mutual connectivity:* a community can be assimilated to a maximal clique or to a γ-quasi clique. Finding maximal cliques in a graph is known to be a NP-hard problem. Generally, cliques of reduced size are used as seeds to find larger communities. Consequently, such approaches are relevant for networks that are rather dense.
- *High internal reachability:* One way to relax the constraint of computing cliques or quasi-cliques is to consider the internal reachability of nodes within a community. Following this, a community core can be approximated by a maximal k-clique, k-club or k-core subgraph. A k-clique (resp. k-club) is a maximal subgraph in which the shortest path between any nodes (resp. the diameter) is $\leqslant \gamma$. A k-core is a maximal connected subgraph in which each node has a degree $\geqslant k$. In [149], authors introduce the concept of k-community which is defined as a connected subgraph $G\prime = \langle \mathcal{V}\prime \subset \mathcal{V}, E\prime \subset E \rangle$ of a graph G in which for every couple of nodes $u, v \in V\prime$ the following constraint holds : $\mid \Gamma(v) \cap \Gamma(u) \mid \geqslant k$. The computational complexity of k-cores and k-communities is polynomial. However, these structures do not correspond to all the community, but are rather used as seeds for computing communities. An additional step for adding non-clustered nodes should be provided. In [114] authors propose to compute k-cores as mean to accelerate computation of communities using standard algorithms, but on size-reduced graphs.

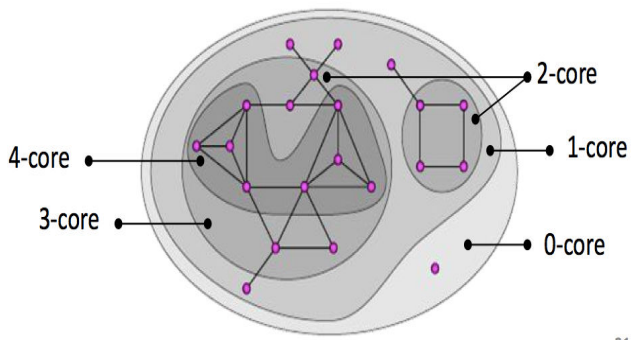


FIGURE 1.2: Example of k-core in a graph [113].

1.2.2 Network-based approaches

These approaches consider the whole connection patterns in the network. Historical approaches include classical clustering algorithms. The adjacency matrix can be used as a similarity one, or topological similarity between each couple of nodes can also be computed. Hierarchical clustering approaches can also then be used [119]. Usually the number of clusters to be found should be provided as an input for the algorithm. Some distributed implementations of these approaches are proposed to provide efficient implementations [146]. More popular network-based approaches are those based on optimizing a quality metric of graph partition. Different partition quality metrics have been proposed in the scientific literature. The modularity is the most widely used one [146]. This is defined as follows. Let $\mathcal{P} = \{C_1, ..., C_k\}$ be a partition of the node's set $\mathcal{V}$ of a graph. The modularity of the partition $\mathcal{P}$ is given by:

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{i=1}^{k} \sum_{u,v \in C_i} \left(A_{uv} - \lambda \frac{d_u d_v}{2m} \right) \tag{1.31}$$

The computing complexity of Q is $O(m)$ [47]. Some recent work has extended the definition to bipartite and multipartite graphs [112, 92, 103, 102] and even for multiplex and dynamic graphs [82, 101]. Different heuristic approaches have been proposed for computing partitions that maximize the modularity. These can be classified into three main classes:

- Agglomerative approaches: These implement a bottom-up approach where an algorithm starts by considering each single node as a community. Then, it iterates by merging the communities guided by a quality criteria. The Louvain algorithm [12] is one very known example of such approaches. The algorithm is composed of two phases. First, it looks for small communities by optimizing modularity in a local way. Second, it aggregates nodes of the same community and builds a new network whose nodes are the communities. Two adjacent communities merge if the overall modularity of the obtained partition can be enhanced. These steps are repeated iteratively until a maximum of modularity is reached. The computing complexity of the approach is empirically evaluated to be $O(n.log(n))$.

- Divisive approaches: These implement a top-down approach, where an algorithm starts by considering the whole network as a community. It iterates to select ties to remove splitting the network into communities. Different criteria can be applied for tie selection. The Girvan and Newman algorithm is the most known representative of this class of approaches [47]. The algorithm is based on the simple idea that a tie linking two communities should have a high betweenness centrality. This is naturally true since an inter-community tie would be traversed by a high fraction of shortest paths between nodes belonging to these different communities. Considering the whole graph G, the algorithm iterates for m times, cutting at each iteration the tie with the highest betweenness centrality. This allows to build a hierarchy of communities, the root of which is the whole graph and leafs are communities composed of isolated nodes. Partition of highest modularity is returned as an output. The algorithm is simple to implement and has the advantage to discover automatically the best number of communities to identify. However, the computation

complexity is rather high: $O(n^2.m + (n)^3 log(n))$. This is prohibitive to apply to large-scale networks.

- Other optimization approach: Other classical optimization approaches can also be used for modularity optimization such as applying genetic algorithms [65, 90, 117], evolutionary algorithms [58] or multi-objective optimization approaches [118].

  Another interesting work has been proposed by P. Pons et al. [120] which is based on computing nodes similarity using Random Walk approach. The distance is the probability that a random walker moves from one node to another in a fixed number of steps. The numbers of steps should be large enough to cover a significant portion of the network. The nodes are grouped into communities through an agglomerative hierarchical clustering and modularity is used to find the best partition in the resulting dendrogram. The algorithm runs with a time complexity of $O(n^2 d)$, where d is the depth of dendrogram. d being often small for real graphs which are sparse, the practical computational complexity is $O(n^2.log(n))$ [44]. Last but not the least, is the method of Infomap partitioning algorithm proposed by M. Rosvall et al. [128]. With greedy modularity optimization, this method produces a partitioning of the network which are generally of very high quality. This algorithm attempts to identify a coarse-grained representation of how information flows through a network. The goal is to optimally compress information needed to describe the process of information diffusion across the graph. Each cluster has a name or a number associated to it and each node inside a cluster has a proper local name or number. Nodes in different clusters may have same local names. A random walk on the network is then given by: [name of cluster C_i − local name of node v_i − local name of noder v_j − ... − local name of node v_k − a code indicating a link outside − name of cluster C_j].

  An example of this is given in figure 1.3. The goal of the algorithm is to find a partitioning and labelling of nodes in the network in order to minimize the expected length of a random walk's description.

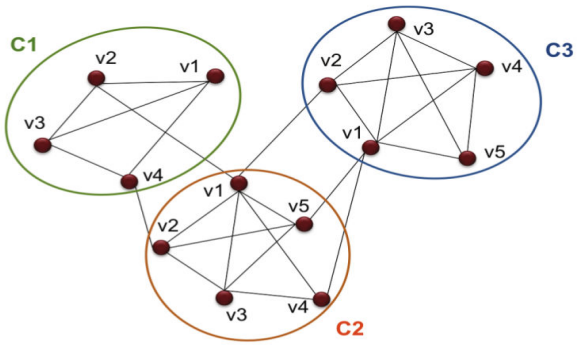


FIGURE 1.3: An example of Infomap execution.
The nodes can have same local names inside communities. A random path between node v1 of community C1 and node v5 of community C3 is [C1 - v1 - v2 - C2 - v1 - C3 - v2 - v3 - v5] [121]

All modularity optimization approaches make implicitly the following assumptions:

- The best partition of a graph is the one that maximize the modularity.
- If a network has a community structure, then it is possible to find a precise partition with maximal modularity.
- If a network has a community structure, then partitions inducing high modularity values are structurally similar.

Recent studies have showed that all three above-mentioned assumptions do not hold. In [50], authors show that the modularity function exhibits extreme degeneracies: it namely accepts an exponential number of distinct high scoring solutions and typically lacks for a clear global maximum. In [83], it has been shown that communities detected by modularity maximization have a resolution limit. These serious drawbacks of modularity-guided algorithms have boosted the research for alternative approaches. Some interesting emerging approaches are label propagation approaches [124] and seed-centric methods [70]

1.2.3 Propagation-based approaches

These approaches have the advantage of fast execution time.Large-scale networks networks can be composed of millions of nodes as it is frequently the case when considering online social networks for example. In addition, usually the real complex networks are very dynamic, and consequently, mining this networks become difficult. A low complexity incremental approaches for community detection are then needed. Label propagation approaches constitute a first step in that direction [124]. The underlying idea is simple: each node $v \in \mathcal{V}$ in the network is assigned a specific label l_v. All nodes update in a synchronous way their labels by selecting the most frequent label in the direct neighborhood. In a formal way, we have:

$$l_v = \underset{l}{\text{argmax}} \mid \Gamma^l(v) \mid \tag{1.32}$$

where $\Gamma^l(v) \subseteq \Gamma(v)$ is the set of neighbors of v that have the label l. Ties situations are broken randomly. The algorithm iterates until reaching a stable state where no more nodes change their labels. Nodes having the same label are returned as a detected community. The complexity of each iteration is $O(m)$. Hence, the overall computation complexity is $O(k.m)$ where k is the number of iterations before convergence. Study reported in [89] shows that the number of iterations grows in a logarithmic way with the growth of n; the size of the target network. In addition to its low complexity, the label propagation algorithm can readily be distributed allowing hence handling very large-scale networks [133, 159].

While the algorithm is very fast, it suffers from two serious drawbacks:

- First, there is no formal proof of the convergence to a stable state.
- Secondly, it lacks for robustness, since different runs produce different partitions due to random tie breaking.

Different approaches have been proposed in the literature to cope with these two problems. Asynchronous, and semi-synchronous label updating have been proposed to hinder the problem of oscillation and improve convergence conditions [32, 124]. However, these approaches by creating dependencies among nodes, increase randomness in the algorithm making the robustness even worse. Different other approaches have been developed to handle the problem of label propagation

robustness. These include balanced label propagation [138], label hop attenuation [89] and propagation preference-based approaches [93]. Another interesting way to handle the instability of label propagation approaches consists simply on executing the algorithm k times and apply an ensemble clustering approach on the obtained partitions [74, 85, 110, 132].

1.2.4 Seed-centric approaches

The basic idea underlying seed-centric approaches is to identify some particular nodes in the target network, called seed nodes, around which local communities can be computed [134, 48, 152, 71]. Algorithm 3 presents the general outlines of a typical seed-centric community detection algorithm.

A sees-centric algorithm is composed from three principal steps:

1. Seed computation.
2. Seed local community computation.
3. Community computation out from the set of local communities computed in the previous step.

Figure 1.4 shows a simple example of such approaches. Leader-driven algorithms constitute a special case of seed centric approaches where the nodes of the network are classified into two (eventually overlapping) categories: leaders and followers. An assignment step is applied to assign followers nodes to most relevant communities, the leaders represent the communities. Different algorithms apply different node classification approaches and different node assignment strategies. Three different Leader based community detection algorithms have been proposed almost simultaneously in three different works [67, 78]. Next, we present briefly the first two cited algorithms.

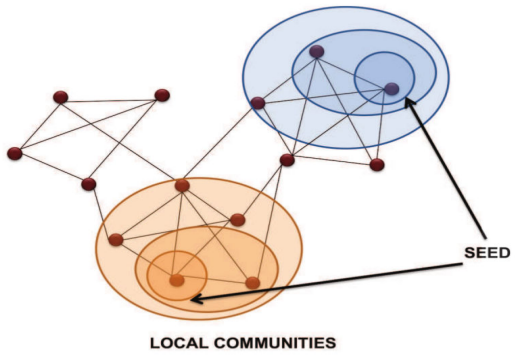


FIGURE 1.4: Seed centric local communities in a network [121].

In [78] authors propose an approach directly inspired from the k-means clustering algorithm [55]. The algorithm requires as input the number k of communities to identify. This is clearly a major disadvantage of the approach that authors of the approach admit. k nodes are selected randomly. Unselected nodes are labelled as followers. Leaders and followers form hence exclusive sets and each leader node represents a community. Each follower nodes is assigned to the most nearby leader node.

Algorithm 3 General seed-centric community detection algorithm

Require: $G =< V, E >$: a connected graph
1: $\mathcal{C} \leftarrow \emptyset$
2: $S \leftarrow$ **compute_seeds(G)**
3: **for** $s \in S$ **do**
4: $C_s \leftarrow$ **compute_local_community(s,G)**
5: $\mathcal{C} \leftarrow \mathcal{C} + C_s$
6: **end for**
7: **return compute_community($\mathcal{C}$)**

Different levels of neighborhood are allowed. If no nearby leader is found the follower node is labelled as outlier. When all followers nodes are handled, the algorithm computes a new set of k leaders. For each community, the most central node is selected as a leader. The process is iterated with the new set of k leaders until stabilization of the computed communities.

The convergence speed depends on the quality of initially selected k leaders. Different heuristics are proposed to improve the selection of the initial set of leaders. The best approach according to experimentation is to select the top k nodes that have the top degree centrality and that share little common neighbors.

Another interesting work is that of Licod algorithm proposed in [155]. The different steps of the algorithm are described below:

1. Search for the leader nodes. This can be done using ranking of nodes based on various criteria. Classical metrics of centrality are very useful for this.

2. The list of found leaders is further reduced by grouping leaders that have higher probability of being in the same communities.

3. For each node in the network (leaders/followers), membership degrees to all communities (represented by the leaders) is computed. A ranked list of communities based on membership degree is obtained for each node. The communities with highest membership degree are ranked on the top.

4. Each node will update its community preference list by merging it with those of its direct neighbors. Different rank aggregation techniques can be used for this purpose. This step will be repeated until stabilization.

   At the end, each node is assigned to the top community in its final ranked list of membership.

1.3 Clustering evaluation

The problem of clustering validationhas long been recognized as one of the vital issues to the success of clustering applications [61]. External clustering validation and internal clustering validation have been considered as the two main categories of clustering validation. Task driven clustering validation, which evaluates the clustering result based on a specific task, can also be considered.

1. **External clustering validation** Evaluation on data for which a ground-truth decomposition into clusters is known.

2. **Internal clustering validation** Evaluation in function of the internal features of computed clusters.

3. **Task-driven evaluation** Evaluate the clustering results in a problem context, i.e. recommendation system, link prediction, etc.

Next, we detail these three different approaches.

1.3.1 External clustering validation

External clustering validation is used when the clustering ground-truth is known. To our knowledge there is no attributed network with ground-truth partitions. However, network with ground truth decomposition can be obtained by one of the following ways:

- **Annotation by experts :** For some data representing, experts in the system field have been able to define the segmentation. In general, these data are rather very small (allowing hence to be handled by experts) and the defined ground-truth segmentation is usually given by a partition of the studied data without considering the attribute information, for example, the Zachary's karate club [158], presented on figure 1.5.

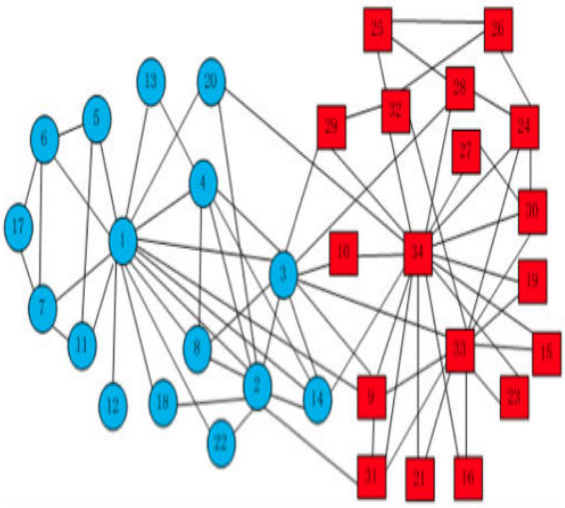


FIGURE 1.5: Zachary's karate club network is a social network of friendships between 34 members of a karate club at a US university in 1970 [158]. Following a dispute, the network was divided into two groups between the club's administrator and the club's instructor. The dispute ended that the instructor created his own club and taking about half of the initial club with him. The network can hence be divided into two main communities.

- **Data generators:** The idea here is to generate artificial data with predefined clustering. Some early work in this area proposed to generate artificial network without attribute as the Girvan-Newman benchmark graph [47] and the LFR benchmark [86]. These generator are deigned to simple graph without considering the attribute information.
  Recently, a sophisticated generator for node attributed network is proposed by

[87] where the user can control different parameters of the network including the size, the number of node's attribute and its distribution Note that the node attribute are numerical. While the approach is interesting, generated networks are not guaranteed to be similar enough to real complex networks observed in real-world applications.

When a ground-truth clustering is available, classical external clustering evaluation indices can be used to evaluate and to compare clustering algorithms. Different clustering comparison or similarities functions have been proposed in the literature [2]. Next, we present the most used external validation indexes.

1. **Simplified Silhouette (SS)**
   A well-known index that is based on geometrical considerations about compactness and separation of clusters is the Silhouette Width Criterion [129]. The original index depends on the computation of distance between objects and cluster centroids, originally the index called Simplified Silhouette [22]. In order to define this index, let's consider that the object x_j is the j^{th} object of the data set and it belongs to the cluster $C_i \in \{C_1, C_2, ..., C_k\}$, where k is the number of clusters in a given partition. Next, let the similarity between the object x_j and the centroid of its cluster C_i be denoted by $s_{x_j,i}$. Also, let $s_{\bar{x_j},i}$ be the dissimilarity between the object x_j and the centroid of its closest neighboring cluster. Then, the simplified silhouette of the individual object x_j is defined as follow:

$$S_x = \frac{s_{\bar{x_j},i} - s_{x_j,i}}{max\{s_{x_j,i}, s_{\bar{x_j},i}\}} \tag{1.33}$$

   where the denominator is just a normalization term. The higher S_x, the better the assignment of the object x_j to cluster C_i. If C_i is a singleton, i.e., if it is composed only the object x then it is assumed by convention that $S_{x_j} = 0$ [76]. The SS index, defined as the average of S_{x_j} over all object of the dataset .

$$SS = \frac{1}{n}\sum_{j=1}^{n} S_{x_j} \tag{1.34}$$

   The best partition is expected to be selected when SS is maximized, which implies minimizing the intra-group distance $(s_{x_j,i})$ while maximizing the inter-group distance $(s_{\bar{x_j},i})$.

2. **Alternative Simplified Silhouette (ASS)**
   A variant of the Simplified Silhouette criterion can be obtained by replacing Eq (1.33) with the following alternative definition of the silhouette of an individual object [148]:

$$S_{x_j} = \frac{s_{\bar{x_j},i}}{s_{x_j,i} + \epsilon} \tag{1.35}$$

   where ϵ is a small constant $(e.g. 10^{-6}$ for normalized data) used to avoid division by zero when $s_{x_j,i} = 0$. Note that the rationale behind Eq (1.35) is the

same as that of Eq (1.33), in the sense that both favor larger values of $s_{\bar{x_j},i}$ and lower values of $s_{x_j,i}$. The difference lies in how they favor, linearly in Eq (1.33) and non-linearly in Eq (1.35).

3. **Calinski-Harabasz (VRC)**
The variance Ratio Criterion [21] evaluates the quality of a data partition as:

$$\text{VRC}(\mathcal{P}) = \frac{\text{trace } (\mathbf{B})}{\text{trace } (\mathbf{W})} \times \frac{n-k}{k-1} \tag{1.36}$$

where **W** and **B** are the $\|\mathcal{T}\| \times \|\mathcal{T}\|$ within-group and between-group dispersion matrices [1] respectively, defined as:

$$\mathbf{W} = \sum_{i=1}^{k} \mathbf{W}_i$$

$$\mathbf{W}_i = \sum_{x_j \in C_i} (x - \overline{x_i})(x_j - \overline{x_i})^T$$

$$\mathbf{B} = \sum_{i=1}^{k} \|C_i\|.(\overline{x_i} - \overline{x})(\overline{x_i} - \overline{x})^T$$

where $\|C_i\|$ is the number of objects assigned to the $i-$th cluster (C_i), $\overline{x_i}$ is the $\mathcal{T}$-dimensional vector of sample means within that cluster (cluster centroid) and $\overline{x}$ is the $\mathcal{T}$-dimensional vector of overall sample means (data centroid or grand mean of the data). As such, the within-group and between-group dispersion matrices sum up to the scatter matrix of the data set, i.e., **T=W+B**, where :

$$T = \sum_{i=1}^{n} (x_i - \overline{x})(x_i - \overline{x})^T$$

The trace of matrix **W** is the sum of the within-cluster variances (its diagonal elements). Analogously, the trace of **B** is the sum of the between-cluster variances. As a consequence, compact and separated clusters are expected to have small trace $(\mathbf{W})$ values and large trace $(\mathbf{B})$ values. Hence, the better the data partition the greater the value of the ratio between trace $(\mathbf{B})$ and trace $(\mathbf{W})$. The normalization term $(n-k)/(k-1)$ prevents this ratio to increase monotonically with the number of clusters, thus making **VRC** an optimization (maximization) criterion with respect to k.

4. **PBM**
Another criterion, named PBM [111], is also based on the within-group and between-group distances:

$$PBM(\mathcal{P}) = (\frac{1}{k}\frac{S_1}{S_k}D_k)^2 \tag{1.37}$$

[1] Note that $\|\mathcal{T}\|$ is the number of attributes that describe the data objects

where S_1 is a constant that donates the sum of distances between the objects and the grand mean of the data, i.e.,

$$S_1 = \sum_{j=1}^{n} \|x_j - \overline{x}\|, S_k = \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - \overline{x_i}\|$$

represents the sum of within-group distances, and $D_k = \max_{i,l=1,2,\ldots,k} \|\overline{x_i} - \overline{x_l}\|$ is the maximum distance between group centroids. According to this equation, the best partition should be indicated when PBM is maximized, which implies maximizing D_k while minimizing E_k.

5. **Davies-Bouldin(DB)**
The Davies-Bouldin index [35] is somewhat related to VCR, since it is also based on a ratio involving within-group and between-group distances. Specifically, the index evaluates the quality of a given data partition as follows:

$$DB(\mathcal{P}) = \frac{1}{k} \sum_{i=1}^{k} D_i \tag{1.38}$$

where $D_i = \text{Max}_{i \neq l}\{D_{i,l}\}$. Term $D_{i,l}$ is the within-to-between cluster spread for the $i-$ th and $l-$ th clusters, given by $D_{i,l} = (\overline{d_i} + \overline{d_l})/d_{i,l}$, where $\overline{d_i}$ and $\overline{d_l}$ are average within-group distances for the $i-$ th and the $l-$ th clusters, respectively, and $d_{i,l}$ is the inter-group distance between these clusters. These distances are defined as $\overline{d_i} = (\frac{1}{\|C_i\|}) \sum_{x_j \in C_i} \|x_j - \overline{x_i}\|$ and $d_{i,l} = \|\overline{x_i} - \overline{x_l}\|$.
The term D_i represents the worst case within-to-between cluster spread involving the $i-$ th cluster. Minimizing D_i for all clusters clearly minimize the Davis-Bouldin index. Hence, good partitions, composed of compact and separated clusters, are distinguished by small values of DB in (1.38).

6. **Dunn (DN)**
The Dunn index [37] is another validity criterion based on geometrical measure of cluster compactness and separation. It is defined as follows:

$$\textbf{DN}(\mathcal{P}) = \text{Min}_{i,l \in \{1,\ldots,k\}_{i \neq l}} \{ \frac{\delta_{C_i,C_l}}{\text{Max}_{p \in \{1,\ldots,k\}} \Delta_{C_p}} \} \tag{1.39}$$

where Δ_{C_p} is the diameter of the cluster p and δ_{C_i,C_l} is the set distance between clusters C_i and C_l. The original definitions of *diameter* and *set distance* in (1.39) were generalized in [10], giving rise to several variants of the original Dunn index. One of the most used of the index is where the set distance across clusters C_i and C_l is defined as $\delta_{C_i,C_l} = \|\overline{x_i} - \overline{x_p}\|$, whereas the diameter Δ_{C_p} of a given cluster p is calculated by $\Delta_{C_p} = \frac{2}{\|C_p\|} \sum_{x \in C_p} \|x - \overline{x_p}\|$. Note that the definitions of Δ_{C_l} and δ_{C_p,C_q} are directly related to the concepts of within-group and between-group distances, respectively. Bearing this in mind, it is easy to verify that partitions composed of compact and separated clusters are distinguished by large values of **DN** in (1.39).

7. **Jaccard index (J)**
In this index [59], which has been commonly applied to assess the similarity between different partitions of the same dataset, the level of agreement between a set of class labels $\mathcal{L}$ and a clustering result $\mathcal{P}$ is determined by the number of pairs of points assigned to the same cluster in both partitions:

$$J(\mathcal{L},\mathcal{P}) = \frac{a}{a+b+c} \tag{1.40}$$

where a denotes the number of pairs of points with the same label in $\mathcal{L}$ and assigned to the same cluster in $\mathcal{P}$, b denotes the number of pairs with the same label, but in different clusters and c denotes the number of pairs in the same cluster, but with different class labels. The index produces a result in the range $[0,1]$, where a value of 1 indicates that $\mathcal{L}$ and $\mathcal{P}$ are identical.

8. **F-measure**
The F-measure combines the precision and recall concepts from information retrieval. We then calculate the recall and precision of that cluster for each class as:

$$Recall(i,j) = \frac{n_{ij}}{n_i} \tag{1.41}$$

and

$$Precision(i,j) = \frac{n_{ij}}{n_j} \tag{1.42}$$

where n_{ij} is the number of objects of class i that are in cluster j, n_j is the number of objects in cluster j, and n_j, is the number of objects in class i. The F-measure of cluster j and class i is given by the following equation:

$$F(i,j) = \frac{2Recall(i,j)Precision(i,j)}{Precision(i,j)+Recall(i,j)} \tag{1.43}$$

The F-measure values are within the interval [0,1] and larger values indicate higher clustering quality.

9. **Variation of Information (VI)**
The variation of information or shared information distance is a measure of the distance between two clustering (partitions of elements). It is closely related to mutual information. The Variation of Information is based on the information theory. This coefficient establishes the quantity of information is included in each partitions, and how much information one partition gives about the other [96].

$$VI = -2\sum_{ij}\frac{n_{ij}}{n}log\frac{n_{ij}n}{n_i n_j} - \sum_i \frac{n_i}{n}log\frac{n_i}{n} - \sum_j \frac{n_j}{n}log\frac{n_j}{N} \tag{1.44}$$

with n the total number of objects, n_i the number of objects belonging to the cluster C_i of $\mathcal{P}1$, n_j the number of objects belonging to the cluster C_j of $\mathcal{P}2$ and n_{ij} the number of object in cluster C_i in $\mathcal{P}1$ and C_j in $\mathcal{P}2$.

10. **Rand index**
The Rand index or Rand measure (named after William M. Rand) [125] is a measure of the similarity between two clustering. From a mathematical standpoint, Rand index is related to the accuracy, but is applicable even when class

labels are not given. It is defined as follow:

$$Rand(\mathcal{P}_1, \mathcal{P}_2) = \frac{S_{00} + S_{11}}{S_{00} + S_{01} + S_{10} + S_{11}} \tag{1.45}$$

we note $\mathcal{P}1$ and $\mathcal{P}2$ the two partitions we wish to compare. We define S_{11} as the number of object pairs belonging to the same cluster in $\mathcal{P}1$ and $\mathcal{P}2$, S_{10} denotes the number of pairs that belong to the same cluster in $\mathcal{P}1$ but not in $\mathcal{P}2$, and S_{01} denotes the pairs in the same cluster in $\mathcal{P}2$ but not in $\mathcal{P}1$. Finally, S_{00} denotes the number of object pairs in different clusters in $\mathcal{P}1$ and $\mathcal{P}2$.

11. **Purity**
Purity is very similar to entropy calculated for a set of clusters. First, we compute the purity in each cluster by

$$p_i = \frac{1}{n_i} Max_j(n_i^j) \tag{1.46}$$

It represents the number of objects in cluster C_i with class label j. In other words, p_i is a fraction of the overall cluster size that the largest class of objects assigned to that cluster represents. The overall purity of the clustering solution is obtained as a weighted sum of the individual cluster purity and given as:

$$Purity(\mathcal{P}_a, \mathcal{P}_b) = \sum_{i=1}^{k} \frac{n_i}{n} p_i \tag{1.47}$$

Where n_i is the size of cluster i, k is the number of clusters, and n is the total number of objects.

12. **Entropy**
Entropy measures the purity of the clusters compared to class labels. Thus, if all clusters consist of objects with only a single class label, the entropy is 0. However, as the class labels of objects in a cluster become more varied, the entropy increases. To compute the entropy of a data set, we need to calculate the class distribution of the objects in each cluster as follows:

$$Entropy_i = \sum_i p_{ij} log(p_{ij}) \tag{1.48}$$

Where the sum is taken over all classes. The total entropy for a set of clusters is calculated as the weighted sum of the entropy of all clusters, as shown in the next equation

$$Entropy = \sum_{i=1}^{k} \frac{n_i}{n} Entropy_i \tag{1.49}$$

Where n_i is the size of cluster i, k is the number of clusters, and n is the total number of objects.

13. **Adjusted Rand Index (ARI)** The ARI index is based on counting the number of pairs of elements that are clustered in the same clusters in both compared partitions. Let $\mathcal{P}_a = \{{C_1}^a, {C_2}^a, ..., C_{k_a}^a\}$ and $\mathcal{P}_b = \{{C_1}^b, {C_2}^b, ..., C_{k_b}^b\}$ be two

partitions of a set of nodes $\mathcal{V}$ It is defined as follows:

$$ARI(\mathcal{P}_a, \mathcal{P}_b) = \frac{\sum_{i=1}^{k_a}\sum_{j=1}^{k_b}\binom{n_{ij}}{2} - t_3}{1/2(t_1+t_2) - t_3} \tag{1.50}$$

where

$t_1 = \sum_{i=1}^{k_a}\binom{n_{ia}}{2}$, $t_2 = \sum_{j=1}^{k_b}\binom{n_{bj}}{2}$, and $t_3 = \frac{2t_1t_2}{n(n-1)}$

This index has expected value zero for independent clustering and maximum value 1 for identical clustering

14. **Normalized Mutual Information (NMI)**

$$NMI(\mathcal{P}_a, \mathcal{P}_b) = \frac{-2\sum_{i=1}^{k_a}\sum_{j=1}^{k_b} n_{ij} log(\frac{n.n_{ij}}{n_{ia}.n_{bj}})}{\sum_{i=1}^{k_a} n_{ia} log(\frac{n_{ia}}{n}) + \sum_{j=1}^{k_b} n_{bj} log(\frac{n_{bj}}{n})} \tag{1.51}$$

where, $\mathcal{P}_a = \{{C_1}^a, {C_2}^a, ..., C_{k_a}^a\}$ and $\mathcal{P}_b = \{{C_1}^b, {C_2}^b, ..., C_{k_b}^b\}$ with k_a and k_b clusters respectively, are two clustering on data set with n samples (observations); n_{ij} signifies the number of common objects in cluster C_i^a in clustering $\mathcal{P}_a$ and in cluster C_j^b in clustering $\mathcal{P}_b$; n_{ia} denotes the number of objects in cluster C_i^a that belong to the partition $\mathcal{P}_a$; and n_{bj} stands for the number of objects in cluster C_j^b that correspond to the clustering $\mathcal{P}_b$.

1.3.2 Internal clustering validation

Without the ground-truth, the quality of a clustering is computed using different internal indexes which evaluate the clustering quality. Three types of topological measures can be used to evaluate the quality of a computed community structure:

- Measures that evaluates the quality of a partition based on internal connectivity.
- Measures that evaluates the quality of a partition based on external connectivity.
- Measures that evaluates the quality of a partition by combining both internal and external connectivity.

In this section, we give an overview of internal validity graph data clustering indexes. Given a set of nodes $\mathcal{V}$, we consider a function $f(C_i)$ that characterizes the connectivity of a community i. Let $G = (\mathcal{V}, E)$ be an undirected graph with $n = \|\mathcal{V}\|$ nodes and $m = \|E\|$ edges. Let C_i be the set of nodes, where n_i is the number of nodes in C_i, $n_i = \|C_i\|$; m_i is the number of edges in the clusters C_i, $m_i = \|(u, v) \in E : u \in C_i, v \in C_i\|$, b_i is the number of edges that pointed outside the cluster C_i and d_u is the degree of node u.

(A) Measures based on internal connectivity

1. **Density** The density function denoted δ allows to obtain information about connection among vertices. It is the ratio between number of edges presented in a cluster C_i and the total number of edges in the whole graph m. The ratios get accumulated for all clusters to evaluate the overall impact. Density values lie in the interval of $[0, 1]$. High density refers to strong connection among these vertices. Formally, we have

$$\delta(\mathcal{P}) = \frac{1}{m} \sum_{C_i \in \mathcal{P}} \|E(C_i)\| \tag{1.52}$$

2. **Internal density:**
Is the edge density of the cluster C_i [123].

$$ID(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{m_i}{n_i(n_i - 1)/2} \tag{1.53}$$

3. **Edges inside:**
Is the number of edges between the nodes in the cluster C_i [123].

$$ED(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} m_i \tag{1.54}$$

4. **Average degree:**
Is the average internal degree of nodes in cluster C_i[123].

$$AD(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{2m_i}{n_i} \tag{1.55}$$

5. **Fraction over median degree (FOMD):**
Is the fraction of nodes of C_i that have internal degree higher than $median(m_i)$, where $median(m_i)$ is the median value of d_u in $\mathcal{V}$.

$$FOMD(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{\|\{u, v \in C_i, \|\{(u, v)\}\| > median(m_i)\}\|}{n_i} \tag{1.56}$$

6. **Triangle Participation Ratio (TPR):**
Is the fraction of nodes is C_i that belongs to a triad.

$$TPR(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{\|\{u \in C_i : \exists v, w \in C_i, (u, v), (v, w), (w, u) \in E\}\} \neq \phi\}\|}{n_i} \tag{1.57}$$

(B) Measures functions based on external connectivity:

1. **Expansion**
It measures the number of edges per node that pointed outside the cluster

(community) [123]:

$$E(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{bi}{n_i} \tag{1.58}$$

2. **Cut Ratio**
   Is the fraction of existing edges (out of all possible edges) leaving the cluster C_i [46]:

$$CR(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{b_i}{n_i(n - n_i)} \tag{1.59}$$

(C) Measures functions that combine internal and external connectivity:

1. **Conductance:**
   It mesures the fraction of total edges volume that points outside the cluster C_i [135].

$$C(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{b_i}{2m_i + b_i} \tag{1.60}$$

2. **Normalized Cut**:[135]

$$NC(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{b_i}{2m_i + b_i} + \frac{b_i}{2(m - m_i) + b_i} \tag{1.61}$$

3. **Maximum-ODF (Out Degree Fraction)**:[42]

$$MODF(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} max_{u \in C_i} \frac{\|\{(u, v) \in E : v \notin C_i\}\|}{d_u} \tag{1.62}$$

4. **Average-ODF:**
   Is the average fraction of edges of nodes in cluster C_i that point out of C_i [42].

$$AODF(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{1}{n_i} \sum_{u \in C_i} \frac{\|\{(u, v) \in E : v \notin C_i\}\|}{d_u} \tag{1.63}$$

5. **Flake-ODF:**
   Is the fraction of nodes in C_i that have fewer edges pointing inside than to the outside of the cluster [42].

$$FODF(\mathcal{P}) = \frac{1}{\|C_i\|} \sum_{C_i \in \mathcal{P}} \frac{\|\{u \in C_i, \|\{(u, v) \in E : v \notin C_i\}\| < d_u/2\}\|}{n_i} \tag{1.64}$$

6. **Modularity:**
   Modularity reflects the concentration of vertices within clusters compared with a random distribution of edges between all vertices regardless of clusters. The value of modularity falls within the range of $[-1, 1]$. A positive value indicates that the number of intra-cluster edges exceeds the number expected on a random basis [109].

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{i=1}^{k} \sum_{u,v \in C_i} (A_{uv} - \lambda \frac{d_u d_v}{2m}) \tag{1.65}$$

1.3.3 Task-driven evaluation

Few networks of large sizes for which a ground-truth partition is known are available. The limitations of topological criteria for evaluating communities and limitations of generators models of artificial benchmark networks, are some factors that motivated the search for new approaches to evaluation partitions detected by the various community detection algorithms. Task-driven evaluation seems to be a promising alternative.
The principle is simple: Let T be a task where community detection can be applied. Let $per(T, Algo^x_{com})$ be a performance indicator of the performance of the task T using community detection algorithm $Algo^x_{com}$. We can compare the performance of two different algorithms based on the indicators per $(T, Algo^x_{com})$ and $per(T, Algo^y_{com})$. The performance of these algorithms can be computed using different external indexes depending on the used Task.

In [113], authors propose to use the recommendation task for evaluating purposes. In [154], authors propose to use the data clustering as an evaluation task.

1.4 Conclusion

In this chapter the state of art in data clustering and community detection were presented.Several concepts of data clustering are inherited by the graph clustering approaches, including quality and proximity measures, algorithms structure and the principle of intra-group density versus inter-group sparsity. One of the important conclusion is that there is no clustering algorithm which solves all problems for all the data types.

Existing algorithms of community detection can be divided into four main family of approaches according to the manner used to calculate the partitions. In general the community detection algorithms have been designed to find groups of nodes based just on the topological configuration of the graph. However, as some authors pointed out, real world networks contain more information, nodes are linked by different types of edges and over those nodes and edges may lay other nodes attribute information which can be useful to identify more accurate clusters.

The state of the art shows that the data clustering and the community detection are very distinct domains which each have their approaches and their optimization criteria. These areas have evolved in parallel, but largely independently. Whether for data clustering or community detection, it seems important to point out that the evaluation of results remains complex today. The number of benchmark-type evaluation data sets is low and often raises issues of relevance in the way the ground truth was built. In the case where the evaluation is made with respect to a precise partition, we have seen that a whole range of measures, from the simplest to the most complex, are available. The internal criteria raise other questions, which join those encountered in the evaluation of community detection in a graph or in data clustering.

Chapter 2

Edge Attributed Network Clustering

Contents

2.1 Introduction

In many application, real-world graph data are often associated with additional information, vertices of a graph are associated with a number of attribute that describe the vertex and they are linked by different types of relationship. Indeed, there are two sources of data that can be used to perform the clustering task. The first is the data about the nodes and their attributes and the second source of data comes from the different kind of connections among vertices considered in this chapter. For example, a graph can contain different types of edges that can occur when we combine information from several data networks: e.g. combining a co-author network with a citation network. In the first graph, authors are connected if they have papers in common. In the second graph, they are connected if a paper of one author cited a paper of the other author. Thus, we will get two edges type: *co-authorship* and *citation*. Furthermore, each type edge might also be associated with a label, e.g. the number of co-authored (or cited) papers or the year the collaboration between the authors started. We denote such a graph with multiple edge types as a multi-relational network and can be represented as a *multiplex graph*. It is defined as a set of graphs, called *layers*, where each graph is composed by the same set of vertices and it represents the edges of one type. Accordingly, in each layer, a different edges set is given. Edge labels can represent characteristics of the relations. For example, a co-author network might contain information about the collaboration between two

authors, such as the begin or end time of the collaboration, research topics, conferences/journals where the joint papers were published etc. For example, figure 2.1 illustrates an example of a multi-relational network which can be represented by a $3-$layer social network where layers represent *advice friendship* and *co-work* relationships among partners and associates of a corporate law company [88].

The community detection problem has gained much of attention for the case of monoplex networks i.e simple graphs, where all edges are of the same type. A large number of different algorithms have been proposed in the scientific literature [43, 142]. Only few works have attempted to adapt algorithms developed for monoplex networks to the case of multiplex networks. One example is the work presented in [101] that apply a greedy optimization of an extended modularity criteria devised for multiplex networks. A multi-objective optimization algorithm has been proposed in [3]. A unified approach, ranging from layer aggregation to partition aggregation is also described in [142].

For this issue, we propose an original approach based on adapting a seed-centric approach to the case of multiplex networks. The approach is mainly based on the *Licod* algorithm proposed in [155]. Seed-centric approaches are mainly based on local computations that makes them suitable to handle large-scale networks. The adaptation to the multiplex case requires redefining basic metrics usually used in such algorithms in order to define seeds: such as the different node centrality metrics, neighborhood definition as well as shortest path measuring between two nodes across different layers.

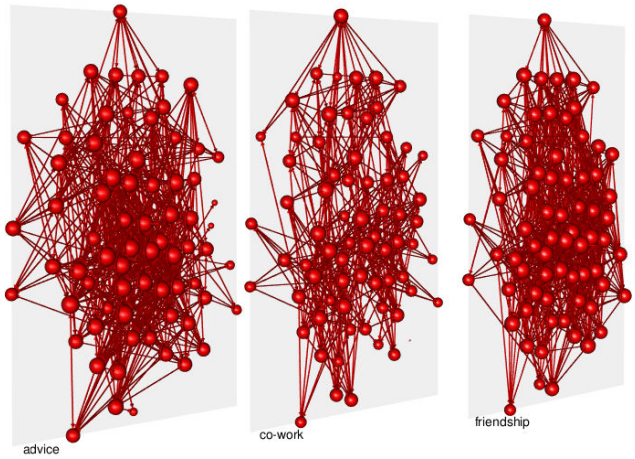


FIGURE 2.1: Lazega Law Firm Network

The remainder of this chapter is organized as follows. In section 3, we present an overview on community detection in multiplex networks after introducing definitions and notations in section 2. Section 4 introduce the proposed *muxLicod* algorithm. Performances, on both benchmark and real networks, of our algorithm are compared to state-of-the-art methods in section 5. Finally, we conclude in section 6.

2.2 Multiplex Network: Definitions and Notation

Formally, a multiplex network is defined as a triplet: $G =< \mathcal{V}, \mathbb{E}, \mathbb{C} >$, where $\mathcal{V}$ is a set of nodes, $\mathbb{E} = \{E_1, \ldots, E_\alpha\}$ is a set of α types of edges between nodes in $\mathcal{V}$ and $E_k \subseteq \mathcal{V} \times \mathcal{V}, \quad \forall k \in \{1, \ldots, \alpha\}$. α denotes the number of layers in the multiplex network. $\mathbb{C}$ is the set of coupling links that represent links between a node and itself across different layers. We have $C = \{(v, v, l, k) : v \in \mathcal{V}, \quad l, k \in [1, \alpha], l \neq k\}$, where (v, v, l, k) denotes a link from node v in the layer l to node v in the layer k. Different coupling schemes can be applied.

- *Ordinal coupling:* where a node in one layer is connected to itself in adjacent layers. In other words, $(v, v, l, k) \in \mathbb{C}$ if $|l - k| = 1$. This is the default coupling when using multiplex networks to model dynamic networks.
- *Categorical coupling:* where a node in one layer is connected to itself in each other layer. This is the default coupling when representing multi-relationnal networks.

Other coupling schemes can also be considered as discussed in [104]. Table 2.1 reminder the notations used later in this chapter.

TABLE 2.1: Multiplex networks: Notations reminder

Notation	Description
$A^{[k]}$	Slice k Adjacency matrix
$d_i^{[k]}$	Degree of node i in layer k
$d_i^{tot} = \sum_{s=1}^{\alpha} d_i^{[s]}$	Total degree of node i
$m^{[k]}$	edge number in slice k
$\Gamma(v)^{[k]} = \{u \in V : (u, v) \in E_k\}$	Neighbor's of v in layer k
$\Gamma(v)^{tot} = \cup_{s \in \{1,\ldots,\alpha\}} \Gamma(v)^{[s]}$	Neighbors of v in all α layers
$SPath^{[k]}(u, v)$	Shortest path length between nodes u and v in slice k
C_{ij}^{kl}	Inter-slice link weight between node i, j in slices $k, l \in 1...\alpha$

2.3 Community Detection in Multiplex graph

Few work have addressed the problem of community detection in multiplex networks. We classify the exiting methods into two main categories detailed below:

1. *Applying monoplex approaches*: the basic idea is to transform the problem of clustering a multiplex into a problem of community detection in simple network [140, 9].
2. Extending existing algorithms to deal directly with multiplex networks [82, 3].

2.3.1 Applying monoplex approaches

One first naive approach consists on aggregating layers of a multiplex network in one layer as illustrated on figure 2.2 [140]. Classical community detection algorithms can then be applied.

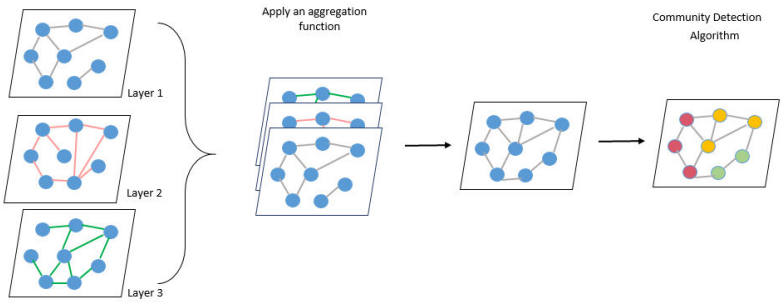


FIGURE 2.2: Layer Aggregation

Different aggregation schemes can be applied. In general, the layer aggregation approach consists on transforming a multiplex network into a weighted monoplex graph $G =< \mathcal{V}, E, W >$ where W is a weight matrix. Different weights computations approaches can be applied and some of the most frequent functions are the follows:

Binary weights: two nodes u, v are linked in the aggregated simple graph if there is at least one layer in the multiplex where these nodes are linked. Formally we have:

$$w_{ij} = \begin{cases} 1 & \text{if } \exists (i,j) \in E_s \quad 1 \leq s \leq \alpha : \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

Frequency-based weighting: in [143], authors propose to weight a link (u, v) by the average of weights in all layers in the multiplex. Formally we have:

$$w_{ij} = \frac{1}{\alpha} \sum_{k=1}^{\alpha} A_{ij}^{[k]} \tag{2.2}$$

A similar weighting scheme is proposed in [8] where a link is weighted by its redundancy over multiplex layers:

$$w_{ij} = \parallel \{d : A_{ij}^{[d]} \neq 0\} \parallel \tag{2.3}$$

Similarity-based weighting scheme: the weight of a link (u, v) is expressed by the similarity of two nodes computed in the multiplex graph. Practically, one can apply temporal dyadic similarity measures to compute a multiplex dyadic similarity score of two nodes (layers in the multiplex are considered as time stamps). In [8], authors propose to use the clustering coefficient to compute the weight of link in the aggregated simple graph.

Linear combination: In [20], authors propose to consider differently the different layers of a multiplex. The weight of a link is the resulting aggregated graph which takes into account the difference of layers contributions. A linear combination schemes is used in this case:

$$A = \sum_{k=1}^{\alpha} w_k A^{[k]} \tag{2.4}$$

where the weights w_k can also be learned based on user defined constraints on the clustering of some nodes into communities.

More recently, another transformation approach has been proposed [29]. It consists on mapping a multiplex to a $3-$uniform hypergraph $H = (\mathcal{V}, E)$ such that the node set in the hypergraph is $\mathcal{V} = \mathcal{V} \cup 1, ..., \alpha$ and $(u, v, i) \in Eif \exists l : A^l_{uv} \neq 0, u, v \in \mathcal{V}, i \in 1, ..., \alpha$. Once we get the hypergraph, we can compute the community detection in hypergraph using for example the tensor factorization approach.

Another monoplex approach consists on applying a community detection algorithm to each layer and then apply an ensemble clustering approach in order to combine all obtained partitions (see figure 2.3).

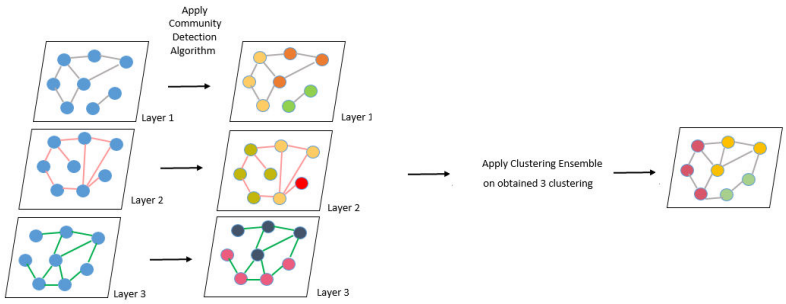


FIGURE 2.3: Partition aggregation approach

From figure 2.3, it can be seen, that after the first step, we obtain four partitions. By ensemble the resulting partitions using ensemble learning ([145], [18], [49]), we obtain a single partition.

2.3.2 Extending monoplex approaches to the multiplex case.

Few studies have addressed the problem of simultaneous exploration of all layers of a multiplex network for the detection of communities. [142] is among the first studies that have tried to extend existing approaches to multiplex setting. The leading role that modularity and its optimization have played in the context of community detection in simple graphs has naturally motivated work to generalize the modularity to the case of multiplex networks. A generalized modularity function is proposed in [101]. This is given as:

$$Q_{multiplex}(P) = \frac{1}{2\mu} \sum_{c \in P} \sum_{\substack{i,j \in c \\ k,l:1 \to \alpha}} \left(\left(A^{[k]}_{ij} - \lambda_k \frac{d^{[k]}_i d^{[k]}_j}{2m^{[k]}} \right) \delta_{kl} + \delta_{ij} C^{kl}_{ij} \right) \tag{2.5}$$

where $\mu = \sum_{\substack{j \in V \\ k,l:1 \to \alpha}} m^{[k]} + C_{jkl}$ is a normalization factor, and λ_k is a resolution factor as introduced [126] in order to cope with the modularity resolution problem.Approaches based on optimizing the multiplex modularity are likely to have the same drawbacks of those optimizing the original modularity function for monoplex approaches [51]. This motivates exploring other approaches for community detection. Seed-centric approaches constitute an interesting option. Actually these approaches are mainly based on local computations that makes these suitable to apply to large-scale networks [71].

2.4 Proposed approach : muxLicod

Seed-centric algorithms present an emerging trend in the area of community detection in complex networks. The basic idea underlying these approaches consists on identifying special nodes in the target network, called seeds, around which communities can then be identified.

Algorithm 4 General seed-centric community detection algorithm

Require: $G =< V, E >$ a connected graph,
1: $\mathcal{C} \leftarrow \emptyset$
2: $S \leftarrow$ **compute_seeds(G)**
3: **for** $s \in S$ **do**
4: $C_s \leftarrow$ **compute_local_com(s,G)**
5: $\mathcal{C} \leftarrow \mathcal{C} + C_s$
6: **end for**
7: **return compute_community($\mathcal{C}$)**

Seed-centric approaches often relay on local computations allowing then to handle large-scale networks. In this work we propose an extension of a seed centric approach to cope with the problem of community detection in a multiplex network. Algorithm 4 presents the general outlines of a typical seed-centric community detection algorithm.

We recognize three principal steps:

1. Seed computation.
2. Seed local community computation
3. Community computation out from the set of local communities computed in step 2.

Each of the above mentioned steps can be implemented applying different techniques. A survey on seed-centric approaches is presented in [71]. In this work, we mainly extend a previous algorithm, the Licod algorithm presented in [155], to be applied to multiplex networks. Algorithm 5 gives the outlines of the approach.

1. Seeds computation : The seeds set computation is done in two steps. First we identify nodes that are likely to play a central role in a community called *leader nodes*. Next, the set of leader nodes are reduced to a set of *seeds* based local connectivity patterns of involved leaders. Leaders that share a lot of *common neighbors* are grouped into one seed. This is simply made by a constructing an ϵ-similarity graph over the set of identified leaders. In such a graph, two leader nodes are linked in their topological similarity is above a given threshold $\epsilon \in [0, 1]$. The applied similarity function is the classical Jaccard similarity (i.e. $Jaccard(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$). The connected components of the obtained ϵ-similarity graph form the set of seeds. Classical centrality functions (ex. degrees of centrality, closeness centrality, pagerank, . . . , etc.) can be used to identify leader nodes. The applied heuristics is the following: any node that have a centrality metric higher than its directe neighbors is a leader node. This can be simply understood in terms of centrality interpretation. A node that have more links that all its neighbors is likely to be a local leader. A node that has a high closeness centrality than its neighbors is a node that neighbors should

pass through in order to reach other nodes in the networks. We select to use the degree centrality function that has a low computational complexity compared to other centralities. This allows to handle large-scale graphs more efficiently. Notice that, applying this heuristics gives automatically the number of communities to detect. In algorithm 5 these two steps are achieved respectively by function *isLeader()* (line 3) and function *computeComumunitiesLeader()*. $\mathcal{L}$ is the set of leader nodes and $\mathcal{C}$ is the set of seeds. We note also that a seed is defined as a set of nodes not necessarelly connected.

2. Local Seed community computation : Once the set of seeds computed, we need to compute the local community of each of the seeds. Different approaches can be applied to compute local communities (called also ego-centered communities). Classical approaches apply an expansion strategy around seeds. A greedy optimization algorithm guided by a given quality function is applied to compute the set of nodes forming the seed community [73]. However, applying an expansion strategy around the set of seed does not guarantee that all nodes of the network will be covered by the set of identified communities. For this reason we propose an agglomerative approach that operates as follows: Each node in the network (a leader or not) computes its membership degree to each community in $\mathcal{C}$; then a ranked list of communities can be obtained, for each node, where communities with highest *membership degree* are ranked first (lines 9-13 in). Next, each node will adjust its community membership preference list by merging it with preference lists of its direct neighbors in the network. Different strategies borrowed form the social choice theory can applied here to merge the different preference lists [38], [26]. This step is iterated until stabilization of obtained ranked lists at each node.

3. Community computation : Finally, each node will be assigned to top ranked communities in its final obtained membership preference list.

Algorithm 5 muxLicod algorithm

Require: $G =< V, E >$ a connected Multiplex graph
$\mathcal{L} \leftarrow \emptyset$ {set of leaders}
/ Leaders identification */*
for $v \in V$ **do**
  if $isLeader(v)$ **then**
    $\mathcal{L} \leftarrow \mathcal{L} \cup \{v\}$
  end if
end for
$\mathcal{C} \leftarrow computeComumunitiesLeader(\mathcal{L})$
/ compute membership vector */*
for $v \in V$ **do**
  for $c \in \mathcal{C}$ **do**
    $M[v, c] \leftarrow membership(v, c)$
  end for
  $P[v]$ = **sortAndRank(**$M[v]$**)**
end for
/ merging membership vector */*
repeat
  for $v \in V$ **do**
    $P^*[v] \leftarrow \mathbf{rankAggregate}_{\mathbf{x} \in \{\mathbf{v}\} \cap \mathbf{\Gamma_G(v)}} \mathbf{P[x]}$
    $P[v] \leftarrow P^*[v]$
  end for
until Stabilization of $P^*[v] \forall v$
/ community computation */*
for $v \in V$ **do**
  / assigning v to communities */*
  for $c \in P[v]$ **do**
    if $|M[v, c] - M[v, P[0]]| \leq \epsilon$ **then**
      $COM(c) \leftarrow COM(c) \cup \{v\}$
    end if
  end for
end for
return $\mathcal{C}$

Applying te above described algorithm to multiplex networks requires to redefine basic metrics usually applied to simple networks: nodes neighborhood, node's degree and shortest path between two nodes [7, 17]. Next we introduce the multiplex version of three basic concepts that are used later in this Chapter.

Neighborhood: Different options can be considered to define the neighborhood of a node in a multiplex. One simple approach is to make the union of all neighbors across all layers. Another more restrictive definition is to compute the intersection of node's neighbors sets across all layers. In [17, 77], authors define a multiplex neighborhood of a node by introducing a threshold on the number of layers in which two nodes are linked. Formally we have:

$$\Gamma_m(v) = \{u \in V \text{ such that } count(i) > m : A_{uv}^{[i]} > 0\}$$

We extend further this definition by proposing a similarity-guided neighborhood: neighbors of a node v are computed as a subset of $\Gamma(v)^{tot}$ composed of nodes

having a similarity with v exceeding a given threshold δ. Using the classical Jaccard similarity function this can be formally written as follows:

$$\Gamma^{mux}(v) = \{x \in \Gamma(v)^{tot} : \frac{\Gamma(v)^{tot} \cap \Gamma(x)^{tot}}{\Gamma(v)^{tot} \cup \Gamma(x)^{tot}} \geq \delta\} \tag{2.6}$$

$\delta \in [0, 1]$ is the applied threshold.

The threshold δ allows to fine-tune the neighborhood size ranging from the most restrictive definition (interaction of neighborhood sets across all layers) to the most loose definition (the union of all neighbors across all layers).

Node degree: The degree of a node is defined as the cardinality of the set of direct neighbors. By defining the multiplex neighborhood function we can define directly a multiplex node degree function. Another interesting multiplex degree function has been proposed in [7]. It defines the multiplex degree of a node as the entropy of node's degrees in each layer. Formally, we can write the following expression:

$$d_i^{multiplex} = -\sum_{k=1}^{\alpha} \frac{d_i^{[k]}}{d_i^{[tot]}} log\left(\frac{d_i^{[k]}}{d_i^{[tot]}}\right) \tag{2.7}$$

The basic idea underlying this proposition, is that the node should be involved in more than one layer in order to be qualified; otherwise its value is zero. The degree of a node i is null if all its neighbors are concentrated in a single layer. However, it reaches its maximum value if the number of neighbors is the same in all layers. This can be useful if we have no prior information about the importance of each layer in the studied multiplex but we want to assume that all layers are important to the target analysis task.

Shortest path : two approaches can be applied to define multiplex dyadic measures (including shortest path). For instance, let $X^{[k]}(u, v)$ be a simple dyadic measure involving nodes u and v in slice k. Two different multiplex versions of the dyadic metric X can then be defined:

$$X^{multiplex}(u, v) = \mathcal{F}(X^{[1]}(u, v), \ldots, X^{[\alpha]}(u, v)) \tag{2.8}$$

Where $\mathcal{F}$ is an aggregation function (ex. average, min, max, ...). Another definition is the one based on the entropy of the metric for both involved nodes across the different layers [122]:

$$X^{multiplex}(u, v) = -\sum_{k=1}^{\alpha} \frac{X(u, v)^{[k]}}{X^{[tot]}} \log(\frac{X(u, v)^{[k]}}{X^{[tot]}}) \tag{2.9}$$

where, $X^{[tot]}(u, v) = \sum_{k=1}^{\alpha} X(u, v)^{[\alpha]}$.

2.5 Experiments

2.5.1 Evaluation criteria

The problem of evaluating community detection algorithms still to be an open problem despite the great amount of work conducted in this field [155]. Since we do not have multiplex networks with ground truth partitions into communities we have opted to evaluate the quality of obtained communities using *unsupervised* evaluation

metrics, namely the multiplex modularity ($\mathcal{Q}$) (see 2.5), the redundancy (ρ) criteria and the complementarity (γ) criteria introduced in [8].

Redundancy criteria (ρ) **[8] :** The redundancy ρ computes the average of the redundant links of each intra-community in all multiplex layers. The intuition is that the link intra-community should be recurring in different layers. To compute this indicator, we firstly denote:

- P the set of couple (u, v) which are directly connected to at least one layer.
- $\bar{\bar{P}}$ the set of couple (u, v) which are directly connected in at least two layers.
- $P_c \subset P$ represents all links in the community c
- $\bar{\bar{P}}_c \subset \bar{\bar{P}}$ the subset of $\bar{\bar{P}}$ and which are also in c.

The redundancy of the community c is then given by:

$$\rho(c) = \sum_{(u,v)\in\bar{\bar{P}}_c} \frac{\parallel \{k : \exists A_{uv}^{[k]} \neq 0\} \parallel}{\alpha\times \parallel P_c \parallel} \tag{2.10}$$

And, the quality of a given multiplex partition is defined as follows:

$$\rho(\mathcal{P}) = \frac{1}{\parallel \mathcal{P} \parallel} \sum_{c\in\mathcal{P}} \rho(c) \tag{2.11}$$

$$\gamma(P) = \frac{1}{\|P\|} \sum_{c\in P} \gamma(c) \tag{2.12}$$

Complementarity criteria (γ) **[8] :** The complementarity γ is the conjunction of three measures:

- Variety $\mathcal{V}_c$: this is the proportion of occurrence of the community c across layers of the multiplex defined as follows:

$$\mathcal{V}_c = \sum_{s=1}^{\alpha} \frac{\|\exists (i,j) \in c / A_{ij}^{[s]} \neq 0\|}{\alpha - 1} \tag{2.13}$$

- Exclusivity ε_c : represents the number of pairs of nodes, in community c, that are connected exclusively in one layer.

$$\varepsilon_c = \sum_{s=1}^{\alpha} \frac{\|\overline{P_{c,s}}\|}{\|P_c\|} \tag{2.14}$$

  with P_c : is the set of pairs (i, j) in community c that are connected at least in one layer and $\overline{P_{c,s}}$ is the set of pairs (i, j) in community c that are connected exclusively in layer s.

- Homogeneity $\mathcal{H}_c$: this measure captures how uniform is the distribution of the number of edges, in the community c, per layer. The idea is that intra-community links must have a uniform distribution among all layers.

The homogeneity is computed using the following expression:

$$\mathcal{H}_c = \begin{cases} 1 & if \quad \sigma_c = 0 \\ 1 - \frac{\sigma_c}{\sigma_c^{max}} & otherwise \end{cases} \tag{2.15}$$

with

$$avg_c = \sum_{s=1}^{\alpha} \frac{\|P_{c,s}\|}{\alpha}$$

$$\sigma_c = \sqrt{\sum_{s=1}^{\alpha} \frac{(\|P_{c,s}\| - avg_c)^2}{\alpha}}$$

$$\sigma_c^{max} = \sqrt{\frac{(max(\| P_{c,d} \|) - min(\| P_{c,d} \|))^2}{2}}$$

The higher the complementarity the better is the partition. The complementarity is then obtained by the following formula:

$$\gamma(c) = \mathcal{V}_c \times \varepsilon_c \times \mathcal{H}_c$$

The above described three criteria i.e. modularity, redundancy and complementarity, capture different proprieties of a multiplex partition and there is no clear indication to know which criteria is better to use. This is why we choose to apply them all: algorithms whose results are in the Pareto-front - outperforms the others.

2.5.2 Datasets

We have selected four different multiplex networks on which we computed the performances of community detection algorithms. These datasets are the following:

CKM Physicians Innovation Network This data set was presented in [28]. It describes relationships between physicians in the context of new drug adaptation. Observed relationships include: advice, discussion and friendship.

Lazega Law Firm Network This data set comes from a network study of corporate law partnership reported in [88]. The network describes relationships between employees in gems of three different relationships: co-working, advice and friendship.

Vickers Chan 7th Graders Network The data were collected by Vickers from 29 seventh grade students in a school in Victoria, Australia [150]. Students were asked to nominate their classmates on a number of tree relationships: Who do you get on with in class ? Who are your best friends class ? Who would you prefer to work with ?

DBLP Network Dblp is a bibliographical database referencing a huge amount of scientific papers mostly related to computer science. We extracted from the publicly available database, a subset corresponding to publications covering the time period 1980 to 1985. A 3-layer multiplex network is constructed out from this dataset: nodes of the multiplex are authors. The first layer encodes a co-authorship relationship. The second one gives co-citation relationships between authors while the third layer gives co-venue relation between authors (participating to the same conference edition).

TABLE 2.2: Multiplex networks

Network	#nodes	#layers	#edges
CKM Physicians Innovation	246	3	1552
Lazega Law Firm Network	71	3	2224
Vickers Chan 7th Graders	29	3	741
DBLP Network	2809	3	293115

2.5.3 Results

To each of the selected networks we have applied the muxLicod algorithm, GenLouvain algorithm [101], an algorithm that apply a greedy optimisation of the multiplex modularity function, and both layer-aggregation and partition aggregation approaches. For layer aggregation we apply a basic aggregation scheme consisting on computing the union of linked in all layers of the multiplex. Both layer aggregation and partition aggregation approaches have been tested with the following classical community detection algorithms: Edge Betweenness [47], Walktrap [119], Louvain [13] and infomap [128]. Next figures show obtained results, in terms of redundancy and multiplex modularity on each dataset.

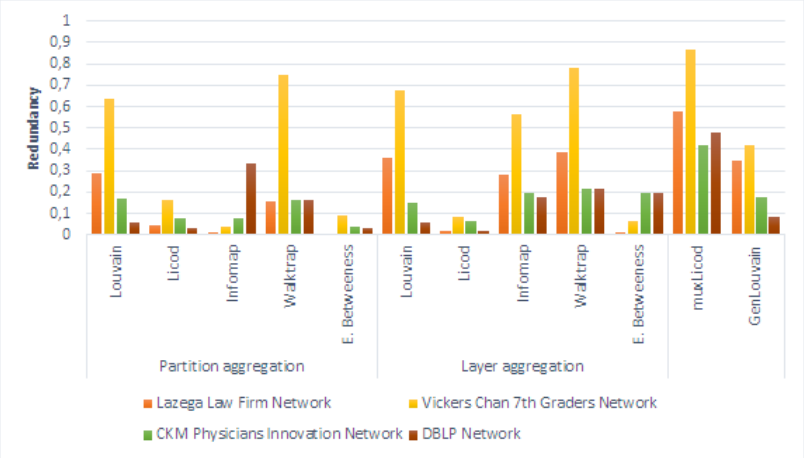


FIGURE 2.4: Result in terms of redundancy

Results show that the muxLicod outperforms GenLouvain algorithm and other approaches in terms of redundancy coefficient. Communities detected by muxLicod seems to be more dense across layers of the multiplex. However in terms of modularity, the results are more mitigated. In a number of situations, the others approaches, namely those based on modularity optimization such as the GenLouvain, Louvain and Walktrap approaches outperform muxLicod. A more careful investigation about the topological characteristics of the multiplex should be done to explore if the similarity between layers plays a role in the success of modularity optimization-based algorithms when applied to multiplex networks. The contrast in results between those obtained in terms of redundancy and modularity raises also a question about the limits of the modularity in estimating the quality of good partitions as it is the case for monoplex networks [45, 51].

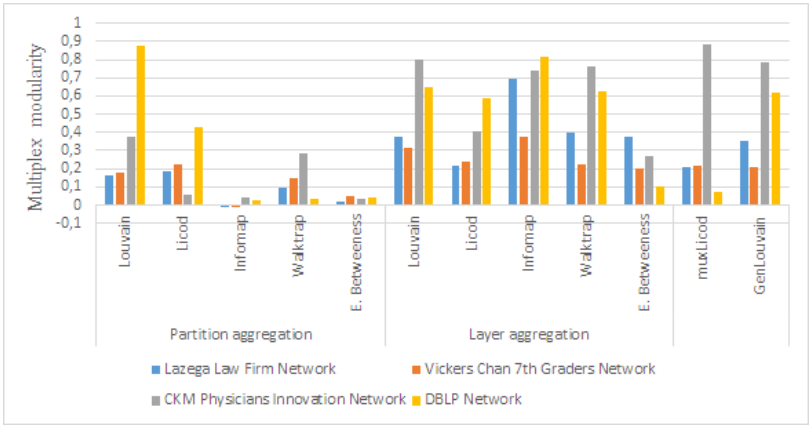


FIGURE 2.5: Result in terms of multiplex modularity

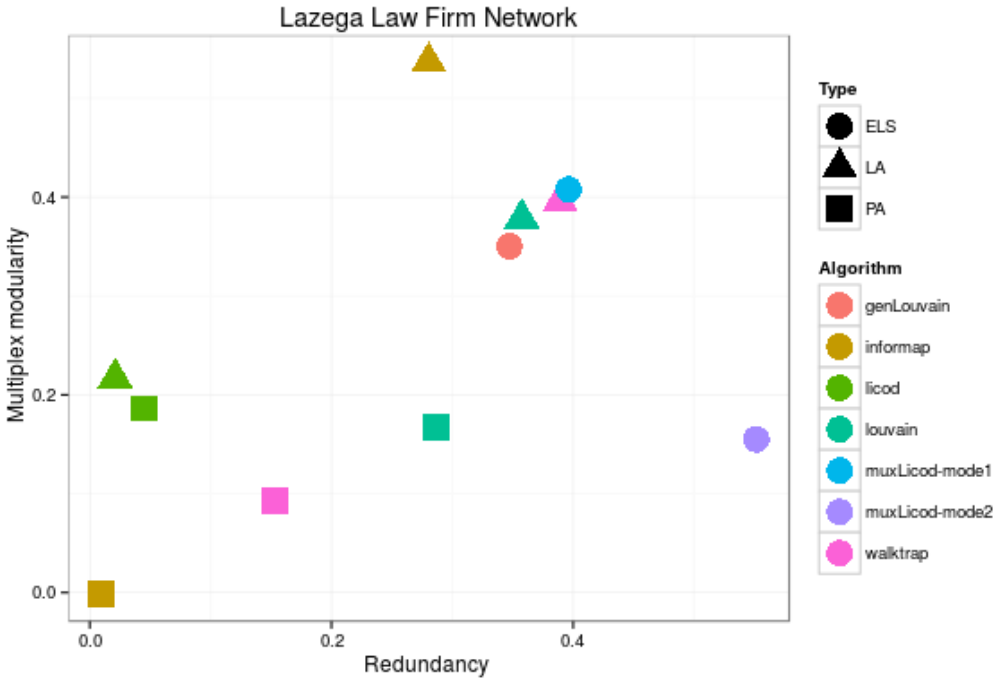


FIGURE 2.6: Pareto Front on Lazega Law Firm Network

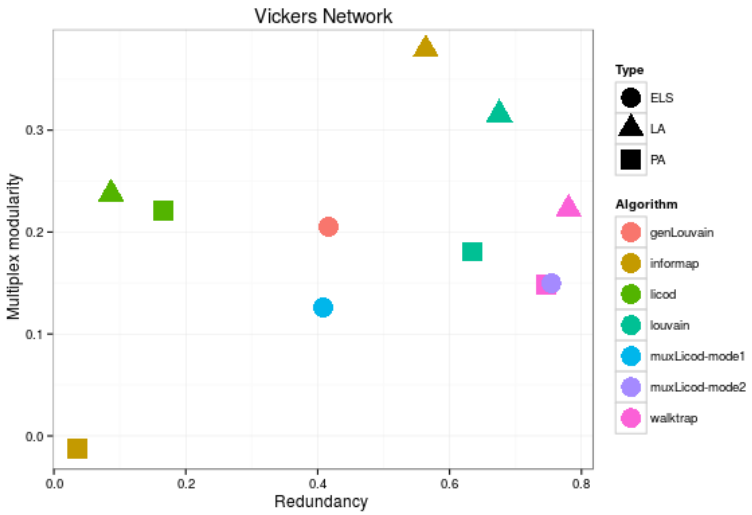


FIGURE 2.7: Pareto Front on Vickers Chan 7th Graders Network

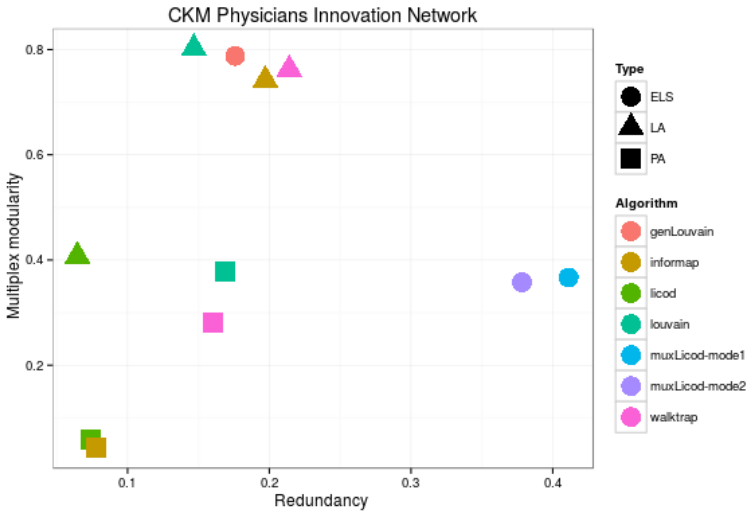


FIGURE 2.8: Pareto Front on CKM Physicians Innovation Network

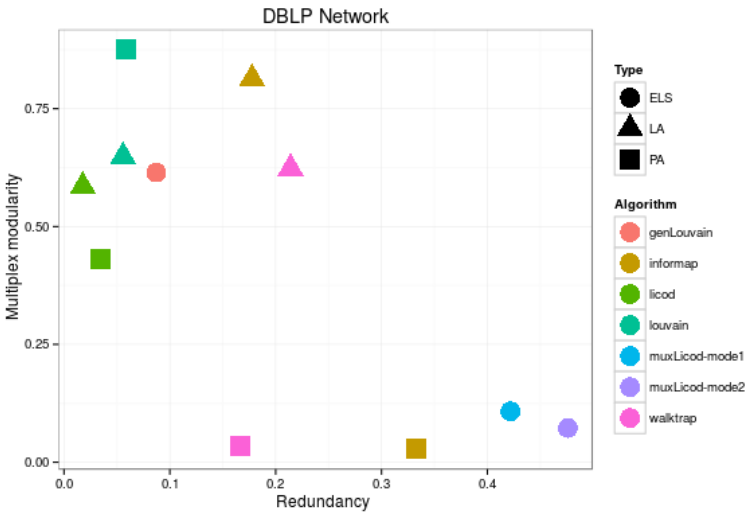


FIGURE 2.9: Pareto Front on DBLP Network

2.6 Conclusion

Few work are designed for multiplex networks and usually these approaches consist on transforming, in a way or another, the multiplex clustering problem into the classical problem of community detection in a monoplex network. One naive approach consists on aggregating layers of a multiplex network in one layer [8, 140]. Another approach consists on applying a community detection algorithm to each layer; and then use an ensemble clustering approach in order to combine all obtained partitions [137].

In this chapter, we presented a new approach of community detection based on seed centered algorithm that takes into account the different type of relationships between the nodes of different layers in the multiplex network. This allows having a good clustering as we have seen from the values of redundancy of our approach compared to GenLouvain, which explores layers simultaneously like muxlicod, and other types of algorithm based on monoplex transformation. For future works, we should test the proposed algorithm on large-scale networks and on a recommendation task in order to better validate the approach.

Chapter 3

Node-attributed Network Clustering

Contents

3.1 Introduction

Most of the state-of-art community detection methods focus only on the topological structure of the graph. However, real-world networks are usually attributed networks i.e. networks with additional information describing either the object and/or the relation between objects. For example, in social networks, edge attributes represents the relationship (friendship, collaboration, family, etc) among people while vertex attribute describe the role or the personality of a person. An other example, is bibliographical network, a vertex may represents an author and vertex properties

describe attributes of the author (i.e. the area of interest, the number of publications, etc), while the topological structure represents relationships among authors (co-authorship, etc).

Thus, it is important to consider both sources of information simultaneously and consider network communities as sets of nodes that are densely connected, but which also share some common attributes. Node attributes can complement the network structure, leading to more precise detection of communities; additionally, if one source of information is missing or is noisy, the other will be used.

However, considering both node attributes and network topology for community detection is also challenging, as the approach have to combine two types of information [156]. Recently, only few recent studies have addressed the problem of clustering in attributed networks.

We summarize the main contributions of this Chapter as follows:

- We summarize the existing methods that deals with attributed clustering problem into different main approaches.
- We propose ANCA, an attributed network clustering algorithm that groups vertices with similar connectivity into clusters that have high attribute homogeneity.
- We evaluate our method on a collection of synthetic data and real data. Experimental results shows that ANCA successfully groups vertices into meaningful clusters. A performance evaluation of ANCA against the state-of-the-art competitors is conducted, which attests its efficacy.

We present a formalization of the problem in the next section. Then a classification review state-of-art methods mining nodes attributed networks will be listed in section 4.3. The description of the proposed approach will be presented in section 4.4. Section 4.5 provides the experimental evaluation of n-ANCA and its comparison against the state-of-the-art methods. Finally, Section 4.6 concludes the present chapter.

3.2 Definition & Problem statement

An attributed graph G is defined as a 4-tuple (V, E, A, F), where $V = \{v_1, v_2, ..., v_n\}$ is a set of n vertices, $E = \{(u, v) : u, v \in |V| \times |V|, u \neq v\}$ is a set of edges, $A = \{a_1, a_2, ..., a_T\}$ is a set of T attributes, $F = \{f_1, f_2, ..., f_T\}$ is a set of T attributes functions and each function $f_t : V \rightarrow dom(a_t)$ assigns to each vertex in V an attribute value in the domain $dom(a_t)$ of the attribute a_t (for $t : 1 \leq t \leq T$). In the attributed graph G, a vertex $v \in V$ is essentially associated with an attribute vector of length T, where the element t in the vector is given by the function $f_t(v)$.

Given an attributed graph $G = (V, E, A, F)$ and the number of clusters k, the clustering problem is to partition the vertex set V of G into k disjoint subsets $P = \{C_1, C_2, ..., C_k\}$, such that :

1. $C_i \cap C_j = \emptyset \; \forall i \neq j$ and $\cup_i C_i = |V|$
2. The vertices within clusters are densely connected, while vertices in different clusters are sparsely connected.
3. Nodes in the same clusters are expected to have homogeneous attributes.

The produced partition integrates the topological structure and nodes attribute information, each cluster within this partition may have this two main dimensions.

3.3 Related work

Compared to the wide range of work on graph clustering, there are much less approaches for attributed graphs, only few recent studies have addressed the problem of clustering node attributed networks. The main existing approaches can be classified into different types of approaches based on their methodological principles :

- Edge weighting based approaches
- Unified distance based approaches
- Augmented graph based approaches

In the following, we describe these approaches.

3.3.1 Edge weighting based approaches

The main used approaches to cluster attributes networks is to integrate the attribute information in the clustering process. This is done by defining a similarity measure between node's attributes that will be used to weight the existing edges. The similarity between nodes is determined by examining each of T attribute values they have in common. Then, any algorithm for weighted graphs clustering can be applied. The change of weights will influence the clustering algorithm to privilege the creation of communities in which vertices are not only well connected but also similar. Algorithm 6 shows the principal outlines of this family of approaches.

Algorithm 6 Edge-weighting based approach

Require: $\mathcal{G}(\mathcal{V}, E, \mathcal{A}, F)$: attributed graph.
$\mathcal{S}$: similarity function.
$clustAlgo_w$: clustering algorithm for weighted graph.
Ensure: Partition of $\mathcal{V}$.
$G_w = (\mathcal{V}, E, w)$; $w : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$
for $(u, v) \in E(G_w)$ **do**
$w(u, v) = S(f_{1..T}(u), f_{1..T}(v))$
end for
$\mathcal{P} = clustAlgo_w(G_w)$
return $\mathcal{P}$

We report in the following main work adopting this strategy. In [106], authors propose the *matching coefficient* similarity function that consists on counting, for two connected vertices, the number of attribute values they have in common. Formally, the *matching coefficient* over two vertices (u, v) is given by :

$$\mathcal{S}(u, v) = \begin{cases} \sum_{t=1}^{T} s_{a_t}(u, v) & if (u, v) \in E \quad or \quad (v, u) \in E \\ 0 & otherwise \end{cases} \tag{3.1}$$

Where

$$s_{a_t}(u, v) = \begin{cases} 1 & if f_t(u) = f_t(v) \\ 0 & otherwise \end{cases}$$

Once weights are changed, authors use classical unsupervised learning algorithm as Karger's Min-Cut [75] or spectral clustering [144] on the weighted adjacency matrix. Note, that community detection algorithms that deals with weighted graph as Louvain [13], Licod [155] can also be used.

The *matching coefficient* similarity measure deals only with categorical attributes. It was extended by Steinhaeuser and Chawla [136] in *NAS* algorithm to handle, at the same time, categorical and continuous attributes. For continuous attributes, first each attribute is normalized in the range $[0, 1]$ by adding a normalizing parameter denoted α and then, the arithmetic difference between the pairs of attribute value is used to obtain a similarity score. This similarity metric is expressed as:

$$s'_{a_t}(u, v) = \begin{cases} 1 \quad if \quad a_t \quad categorical \quad \& \quad f_t(u) = f_t(v) \\ 1 - \alpha_t |f_t(u) = f_t(v)| \quad if \quad a_t \quad continuous \\ 0 \quad otherwise \end{cases}$$

Where α_t is a normalizing parameter that corresponds to the attribute a_t. It allows to normalize the attribute value of a_t in the range $[0, 1]$.

Then communities are obtained using a simple thresholding method. Given threshold in the range $(0, 1)$, any pair of nodes u and v whose edge weight exceeds the threshold are placed in the same community. Edge weighting based approaches produces a new edge weights according to node attribute similarity. If the the original graph is weighted, the two weight can be merged by multiplying them. This family of techniques are simple to implement but their disadvantage is that they take in consideration only vertices that are directly connected. Vertices that are not directly connected in the graph have a similarity equal to zero regardless their attribute value which can decrease the communities accuracy when the graph contains many not directly connected vertices.

3.3.2 Unified distance based approaches

Unified distance based approaches store the networks structural information into a similarity or a distance function between vertices. Generally this unified distance is defined as a linear combination between structural distance function and node-attribute distance. Once this function is defined, classical distance-based clustering methods can be applied.

Formally we have:

$$dis(u, v) = \alpha dis_T(d_u, d_v) + (1 - \alpha) dis_S(u, v) \tag{3.2}$$

where :

- $dis_T(d_u, d_v)$: represents the topological distance between vertices u and v. Different topological distance can be used as the shortest path, the neighborhood random walk distance, etc.
- $dis_S(d_u, d_v)$: is an attribute distance between vertices u and v.
- $\alpha \in [0, 1]$: is a parameter introduced to control the influence of both similarity aspects. The importance of structure and context similarity can change and depends on the application domain. Therefore, the choice of appropriate value for this parameter is critical. Usually, social networks often exhibit dense regions and follow power-law degree distribution. The higher value for α seems effective for these networks because nodes in dense regions are expected to have similar attributes. However, for non-scale free networks, e.g. road networks, we need to treated this parameter with a balanced ratio, which can be hard to determine.

As an example, authors in [30], define an unified distance as a linear combination of two distances, each corresponding to a type of data: cosine distance on textual information and geodesic distance on the network structure. Then a hierarchical agglomerative clustering is applied with the unified distance matrix. Another similar unified distance function is proposed by [34] that is used to build a k-nearest neighbor graph. Communities will be found using the Louvain algorithm.

In [105] authors proposed a unified distance called Collaborative Similarity Measure (CSM) that is computed through structural and contextual similarity inspired by Jaccard similarity coefficient. Authors distinguish between three basic scenarios for vertex pair connectivity: connected, indirectly connected, or disconnected. Formally this measure is given by the following expression:

$$CSM(u,v) == \begin{cases} \frac{1}{CSIM(u,v)} & if \quad (u,v) \in E \\ \sum\limits_{z \in Path(u,v} \frac{1}{CSIM(z,z+1)} if & \exists \quad Path(u,v) \quad \& \quad (u,v) \not\in \quad E \\ \infty \quad if \quad \nexists \quad Path(u,v) \end{cases} \tag{3.3}$$

$$CSIM(u,v) = \alpha * SIM(u,v)_{struct} + (1-\alpha) * SIM(u,v)_{context}$$

$$SIM(u,v)_{struct} = \begin{cases} \frac{w_{uv}}{\sum\limits_{z \in \Gamma(u)} w_{uz} + \sum\limits_{z \in \Gamma(v)} w_{vz} - w_{uv}} & if \quad (u,v) \in E \\ \prod\limits_{z \in Path(u,v)} SIM(z,z+1)_{struct} & if \exists Path(u,v) \& (u,v) \not\in E \\ 0 \quad if \quad \nexists \quad Path(u,v) \end{cases}$$

$SIM(u,v)_{struct}$ represents the structural similarity between two vertices, u and v. It is defined as the weighted ratio of common neighbors to all the neighbors of both vertices. A directly connected pair of vertices employ direct neighborhood information to estimate the similarity value. However, the similarity value for indirectly connected vertices, by following a path, is calculated through linear product of direct structural similarity values. The similarity value becomes zero for disconnected vertices. $SIM(u,v)_{context}$ represents the attribute similarity between two vertices, u and v.

$$COMMON(u,v,a_t) = \begin{cases} 1 & if \quad \& \quad f_t(u) = f_t(v) \\ 0 & otherwise \end{cases}$$

and

$$SIM(u,v)_{context} = \begin{cases} \frac{\sum\limits_{t=1}^{T} COMMON(u,v,a_t) * w_{a_t}}{\sum\limits_{t=1}^{T} w_{a_t}} & if \quad (u,v) \in E \quad or \quad \nexists \quad Path(u,v) \\ \prod\limits_{z \in Path(u,v)} SIM(z,z+1)_{context} if & \exists Path(u,v) \quad \& \quad (u,v) \not\in E \end{cases}$$

Vertices are then grouped together based on computed similarity under the k-Medoid method.

3.3.3 Augmented graph based approaches

This type of approaches seek to combine the topological structure and the attribute information through an augmented graph. The initial topological structure of the original graph is augmented by new vertices called *attribute vertices* and new edges called *attribute edges*. An *attribute vertex* $v_{a_{ti}}$ represents an attribute-value pair (a_t, a_{ti}) where $a_{ti} \in dom(a_t$ is a value of the attribute a_t. If a vertex v has the value a_{ti} on the attribute a_t, an *attribute edge* is added between the vertex v and the *attribute vertex* $v_{a_{ti}}$. With such graph augmentation, the attribute similarity is expressed as the vertex neighborhood in the augmented graph: two vertices sharing the same attribute value are connected by a common *attribute vertex*. Since each vertex v_i has T attribute values, there are totally $|\mathcal{V}| \times T$ attribute edges added to the original graph.

In the augmented graph, two vertices are close either if they are connected through many other original vertices, or if they share many common *attribute vertices* as neighbors, or both. Once the augmented graph is created, distance measure which estimate the pairwise vertex closeness or a community detection algorithm can be applied to find out the set of clusters. Next, in algorithm 7, we formally present the principle of attributed network algorithm based on augmented graph.

Algorithm 7 Augmented graph based approach

Require:
$\mathcal{G}(\mathcal{V}, E, \mathcal{A}, F)$: attributed graph.
$clust$: clustering algorithm
Ensure: A partition of $\mathcal{V}$.
1: $\mathcal{V}' = \mathcal{V} \cup \mathcal{V}_a$; $\mathcal{V}_a = \{(a_t, a_{ti})\}$ with $t \in \{1..T\}$ and $i \in dom(a_t)$
2: $E' = E \cup E_a$ with $E_a \subseteq \mathcal{V} \times \mathcal{V}_a$
3: $G' = (\mathcal{V}', E')$
4: $\mathcal{P} = clust(G')$
5: **return** $\mathcal{P}$

Authors in [161], [25] propose to use the neighborhood random walk distance to compute a unified distance between vertices on the augmented graph. The random walk distance between two vertices is based on the paths consisting of both structure and attribute edges. In this way, it combines the structural closeness and attribute similarity through the random walk distance measure. Then, the random walk distance is used as pairwise similarity measure in the clustering process by K-Medoids clustering approach to partition the graph into k clusters.

Approaches based on augmented graph can handle only categorical attributes but it can be easily extended to handle both categorical and continuous attributes. Usually, for continuous attributes, the values are transformed in different intervals value. The disadvantage of these approaches is that they are limited to small networks with few attribute values.

3.3.4 Quality function optimization based approaches

This family of approaches extend the well-know graph based methods to consider both attributes information and topological structure. The existing approaches mainly extend the Louvain algorithm [13] as linear combination of the Newman [107] modularity and new measure that computes the attribute similarity.

Cruz & al. include the entropy optimization as an intermediate step between modularity optimization and community aggregation. This is done to minimize semantic disorder of nodes by moving nodes among the clusters found during the modularity optimization. These steps are iterated until the modularity is not longer improved. [34] proposed an extension of the Louvain algorithm [13] with a modification of modularity by including the similarity of the attributes given by:

$$Q^{+} = \sum_{C_i \in P} \sum_{v,u \in C_i} \alpha.[\frac{1}{2m}(A_{vu} - \lambda \frac{d_v d_u}{2m})] + (1-\alpha).S(u,v)$$

Where $S(v,u)$ is a similarity function based on type of attributes of v and u and it can be adapted according to how the attributes are represented. $\alpha \in [0,1]$ is a weighting factor which represents the degree of contribution of structural and attribute information.

Another extension of Louvain is proposed by [31] called ILouvain algorithm which uses the inertia based modularity combined with the Newman's modularity.

Modularity optimization approaches make assumption that the best partition of a graph is the one that maximizes the modularity, but [51], [84] have shown that this assumption can not be satisfied if the modularity is not a pertinent measure for some graphs.

3.4 n-ANCA : nodes Attributed Network Clustering Algorithm

Our challenge is to use the topological information of the network and the nodes attribute information simultaneously during the learning process to detect the communities. Instead of changing the topological structure of the network as edge weighting based approaches, or by adding new attribute vertices and attribute edge as done by the approaches based on the augmented graph, it is possible to represent the topological structure of the network by a set of new features. These features represents a set of nodes in the network that will be used to characterize each vertex. The position of each vertex in the network will be characterized by its topological relationship over these set of *seeds* i.e. landmarks in the network. We propose a node attributed network clustering algorithm (n-ANCA) that is based on this principle.

The proposed method, nodes attributed network clustering algorithm *n-ANCA*, can operate on undirected, weighted or un-weighted, and multi-attributed graph, which requires one parameter as prior knowledge - the number of clusters to be found even though it's unsupervised. At any stage, it does not require to alter the structure of the graph. Systematically, it consists of two main components similarity matrix calculation and clustering.

Given an attributed graph $G(V,E,A,F)$ and k number of clusters, the main steps of the n-ANCA algorithm are:

1. Seed selection: Select a set of central nodes.
2. Node's characterization: Characterize each node of V by its relationship with seed nodes S using a topological distance.
3. Incorporating both type of informations:
   Compute a similarity measure over node's attribute.

Apply matrix factorization techniques after incorporating seeds features and node's attribute features.

4. Clustering Process: Apply an unsupervised learning technique to cluster the data.

Encapsulating topological information with attributes enriches the dimension space. To this end, we will use the matrix factorization techniques reliable for this kind of problem. Each of the above step can be implemented using different techniques and we detail hereafter each of these steps.

3.4.1 Seed selection

Various techniques can be used to select the *seeds* set [72]. A simple selection technique is to pick randomly a set of qualified seeds i.e, select the k-top central nodes in the graph [78]. An advanced selection as described in [68] consists on selecting nodes with higher centrality compared to their direct neighbors. In our approach, we look to characterize the position of each node in the network i.e. seeds must occupy diverse positions in the network. Each seed will provide a part of the network from its own position. Our strategy is to select nodes playing central role in the major part of the network and also nodes that are in the periphery of the network. For this purpose, we can use the following proprieties to select the *seeds* set :

- Seeds are the union of k-top and l-lower central nodes in the graph, using linear centrality i.e. page rank centrality, eigenvector centrality and degree centrality
- Seeds are the nodes with higher degree centrality compared to their direct neighbors.
- Seed are the set of articulation nodes to which we add the top central nodes in the biconnected core.

3.4.2 Node's characterization

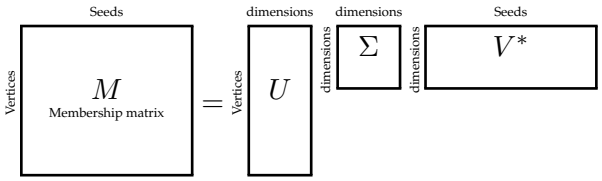
Once *seeds* set is selected, a topological distance/similarity metric will be used to outline the relationship between the set of *seeds* and each nodes of the network. Topological distance/similarity metrics are divided into two main classes: neighborhood based metrics and path based metrics. Neighborhood based metrics are based on the idea that two nodes are similar if they are connected to the same or similar neighbors. Similarity measures for binary vectors as *Jaccard Index* can be used in this case. However, path based metrics use either the length of the path, i.e, the shortest path length, Kat'z Index, or the time required to reach from one node to another i.e. the Matrix Forest Index, etc. The topological similarity will be used to compute the membership matrix $|\mathcal{V}| \times |S|$ between graph nodes and the *Seeds*. Several distance or similarity measure can be used to compute the pair-wise nodes attribute similarity matrix $|\mathcal{V}| \times |\mathcal{V}|$ and it depends on the type of attribute data (numeric, binary, categorical, etc) for each attribute [36].

3.4.3 Incorporating both type of information

Before concatenating nodes attribute similarity matrix and membership matrix, we apply a single value decomposition to both matrices in order to reduce the dimension size of the matrix. The singular value decomposition (SVD) is a matrix factorization technique the most used for this purpose. For example, the membership matrix M is an $|\mathcal{V}| \times |S|$ matrix, then we may write M as a product of three factors:

$$M = U\Sigma V^* \ , \tag{3.4}$$

where U is an orthogonal $|\mathcal{V}| \times |\mathcal{V}|$ matrix, V is an orthogonal $|\mathcal{V}| \times |S|$ matrix, V^* is the transpose of V, and Σ is an $|\mathcal{V}| \times |S|$ matrix that has all zeros except for its diagonal entries, which are nonnegative real numbers. If σ_{ij} is the i, j entry of Σ, then $\sigma_{ij} = 0$ unless $i = j$ and $\sigma_{ii} = \sigma_i \geq 0$. The σ_i are the "singular values" and the columns of u and v are respectively the right and left singular vectors.



Then use the top k eigenvectors of M (resp. M') corresponding to the k smallest eigenvalues as the low dimensional representations of membership matrix (resp. nodes attribute similarity matrix).

3.4.4 Clustering process

At the last stage, the partitioning of the graph vertices is done by employing the k-means clustering approach. This clustering is strongly influenced by initial centroid selection. The blind strategy is usually used to select centroids randomly to avoid local optimum. In addition to this, we have also used the guided mechanism, i.e. centroids are selected based on degree centrality of the vertices, to analyze the quality of the resultant clusters. Consequently, at the beginning, k vertices are appointed either randomly or based on their degree as centroid (expected center points) or initial seeds to represent each cluster. The clustering module implicitly selects the centroids based on their edge degree ranking unless stated using the term Random. Then we associate neighboring vertices to the nearest centroids based on their distance values to make a partition.

The k-means algorithm divides a set of n elements into k disjoint groups. The main idea is to assign each element to one of k centroids. In general, the problem solved by this algorithm is stated as: given a set P of observations, where each observation is represented as a p-dimensional vector, find a partition $C = C_1, C_2, ..., C_k$ according to:

$$argmin_{C_i} \Sigma_{i=1}^{k} \Sigma_{v \in C_i} |v - \mu_i| \tag{3.5}$$

where μ_i is the centroid of the group i. Thus, the algorithm searches for a partition configuration that minimizes the distances of the points to their respective group centroids. Solving this optimization problem is known to be NP-Hard [95] however several heuristic approaches have been developed to find, good enough, local minima solutions.

K-means algorithm [94] requires to set the final number of groups k beforehand. This is not always an easy task, and even more, an incorrect choice of k may lead to poor results; this is why in many cases a preprocessing step is required to determine a suitable value for k. This preprocessing depends on the type of data and the particular application. A general rule of thumb [66] to select k is:

$$k \approx \sqrt{\frac{n}{2}} \tag{3.6}$$

where n is the number of elements in the data set.

Algorithm 8 presents the details of the proposed approach using the shortest path as topological distance and the Euclidean distance as attribute similarity.

Algorithm 8 Attributed Network Clustering Algorithm (ANCA)

Require:
$G(V, E, A, F)$: attributed graph.
Ensure: Partition of V.

1: Let $S \subseteq V$ be a set of nodes.
2: Form the membership matrix $M \in \mathbb{R}^{|V|\times|S|}$ defined by $M(u,s) = SPath(u,s)$ $\forall u \in V, \forall s \in S$
3: Compute the attribute similarity matrix M' defined by $M'(u,v) = \sqrt{f_{a_t}(u) - f_{a_t}(v)}$ $\forall u, v \in V, t \in \{1, ..., T\}$
4: Find $x_1, x_2, ..., x_l$, the l largest eigenvectors of M and find $x'_1, x'_2, ..., x'_{l'}$, the l' largest eigenvectors of M'.
5: Form the matrix $X \in \mathbb{R}^{|V|\times(l+l')}$ by stacking the eigenvectors of M and M' in columns.
6: Form the matrix Y from X by normalizing each of X's rows to have unit length defined by $Y_{ij} = X_{ij}/\sqrt{\sum_j X_{ij}^2}$.
7: Cluster each row of Y into k-cluster via K-means algorithm.

After clustering step, we receive the desired output as a set of vertices in k groups and each vertex is assigned to a single cluster.

3.5 Experiments

3.5.1 Experimental setup and baseline approaches

In this section, we performed experiments to evaluate the performance of our proposed *ANCA* algorithm that was implemented using R language. The results were compared to several state-of-art methods i.e. *SA-Cluster* [161], *SAC* [34], *IGC-CSM* [105], *NAS* [136], ILouvain [31]. The *ANCA* source code, the other approaches cited earlier, evaluation measures and the datasets used for the experiments in the chapter are available on the following web page :[1] as well as the R library.

3.5.2 Datasets

We have analyzed the proposed strategy on three real and several synthetic datasets.

[1] lipn.univ-paris13.fr/~falih/packages/ANCL/

Synthetic data

We evaluate the parameters of the proposed approach *n-ANCA* on a set of artificial data generated using attributed network generator [87] for which the ground truth decomposition into communities is known. We consider six attributed networks with two continuous attributes and where the number of edges $|E| = 3 \times |V|$ and $|V| \in \{100, 500, 1000, 5000, 10.000, 20.000\}$. We evaluate also the run-time of n-ANCA on these networks.

Real-world data

- *Political Blogs :* The dataset has $1,490$ vertices and $19,090$ edges. Each vertex represents a web blog on US politics and each directed edge represents a hyperlink from one web blog to another. Each vertex is associated with an attribute, indicating the political leaning of the web blog, liberal or conservative. Since we only consider undirected graphs in this work, we ignore the edge directions in this dataset, which results in $16,715$ undirected edges.

- *DBLP10K :* The dataset is a co-author network extracted from the DBLP Bibliography data. Each vertex represents a scholar and each edge represents a co-author relationship between two scholars. The dataset contains $10,000$ scholars who have published in major conferences in four research fields: database, data mining, information retrieval, and artificial intelligence. Each scholar is associated with two attributes, prolific and primary topic. The attribute "prolific" has three values: "highly prolific" for the scholars with ≥ 20 publications, "prolific" for the scholars with ≥ 10 and < 20 publications, and "low prolific" for the scholars with < 10 publications. The domain of the attribute "primary topic" consists of 100 research topics extracted by a topic model [56] from a collection of paper titles from the scholars. Each scholar is then assigned a primary topic out of the 100 topics. This dataset was given by [105].

- *Emails :* due to emails privacy issues, there is no public corpus from a real organization available except for a huge Anonymized Enron email corpus.It contains vast collection of emails covering a time span of 41 months, and also uniquely depicts the ups and downs of the energy giant Enron. It provides an opportunity to determine related mailbox users based on their unique communication and relationship in the email network. We have considered the Enron email data set 5, which contains all the emails of 161 users, managed separately, to infer the community structure from partial information available in terms of personalized emails. Two users, Sally Beck and Louise Kitchen, email network is exploited from this data set for all the experiments in this paper to infer the community structure. The email interactions with the individuals outside the Enron Corporation are explicitly ignored to reflect factual associations.

Table 3.1 summarize the information about the graphs.

Figures 3.1, 3.2 and 3.3 show the distribution of each nodes attribute in respectively Polblogs network, Emails network and DBLP network.

Network	$\| \mathcal{V} \|$	$\| E \|$	$\| \mathcal{A} \|$	δ_G	$\mathcal{CC}$
Political Blogs	1 490	16 715	1	0.0086	0.226
Emails	4 256	1 0139	6	0.0011	0.029
DBLP10K	10 000	27 867	2	0.0006	0.224

TABLE 3.1: Main features of the used dataset. The number of vertices $\| \mathcal{V} \|$, the number of edges $\| E \|$, the number of nodes attributes $\| \mathcal{A} \|$, δ_G is the density of the network and $\mathcal{CC}$ is the clustering coefficient.

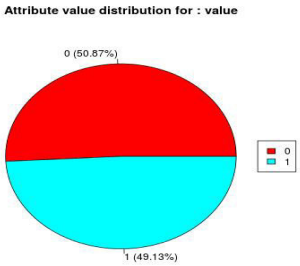


FIGURE 3.1: nodes attribute value distribution for Polblogs network.

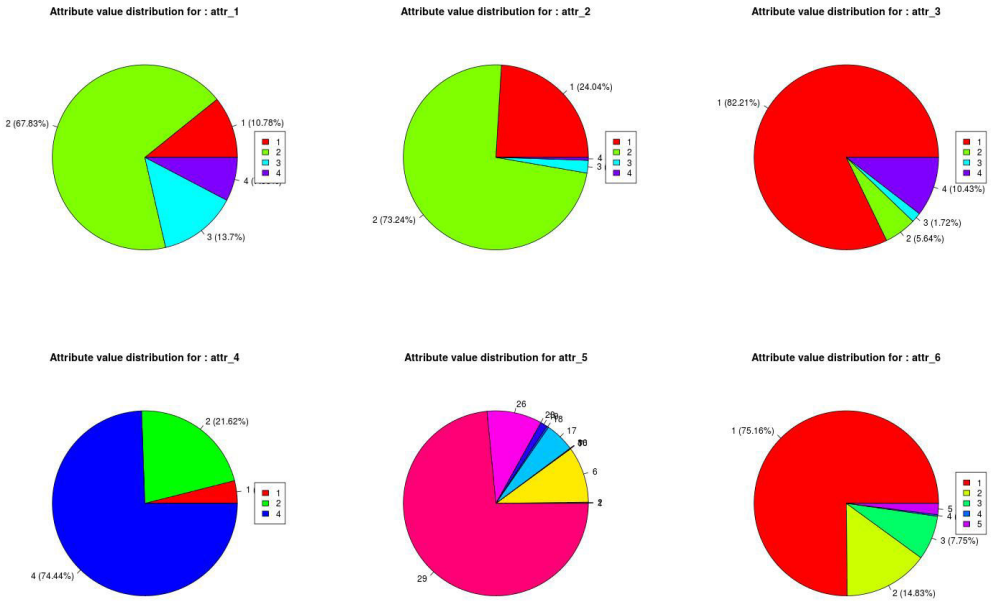


FIGURE 3.2: nodes attribute value distribution for Political blogs network.

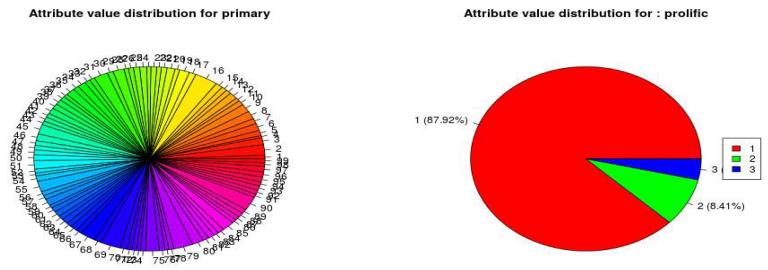


FIGURE 3.3: nodes attribute value distribution for DBLP10k network.

As it can be noticed, the distribution is very different for these networks, in order to test ANCA on networks with different problems.

3.5.3 Study of the effect of n-ANCA parameters

Our first experiments aim to evaluate ANCA's parameters as seeds selection and topological similarity metrics. We use three types of seeds selection:

- *Central*: Seeds are the union of 15% top and 5% lower central nodes in the graph combining between three different centrality which are Page rank centrality, Eigenvector centrality and degree centrality.
- *LeaderC*: Seeds are the nodes with higher degree centrality compared to their direct neighbors.
- *BiCC*: Seed are the set of articulation nodes to which we add the top central nodes in the biconnected core.

We pick a topological similarity measure based on neighborhood and another based on length of the path which are:

- Neighborhood based metric: Jaccard similarity matric *SimJa*.
- Path based metric: Shortest path *SPath*.

The combination of the parameters create six different variants of n-ANCA as following:

- ANCA : BiCC + SPath
- ANCA : BiCC + SimJa
- ANCA : Central + SPath
- ANCA : Central + SimJa
- ANCA : LeaderC + SPath
- ANCA : LeaderC + SimJa

Figure 3.4 present the scalability of n-ANCA using different parameters sets. The selection methods influence the run-time of n-ANCA algorithm as it can be noted that n-ANCA with a percentage of top and low central node is faster than the one using other selection method.

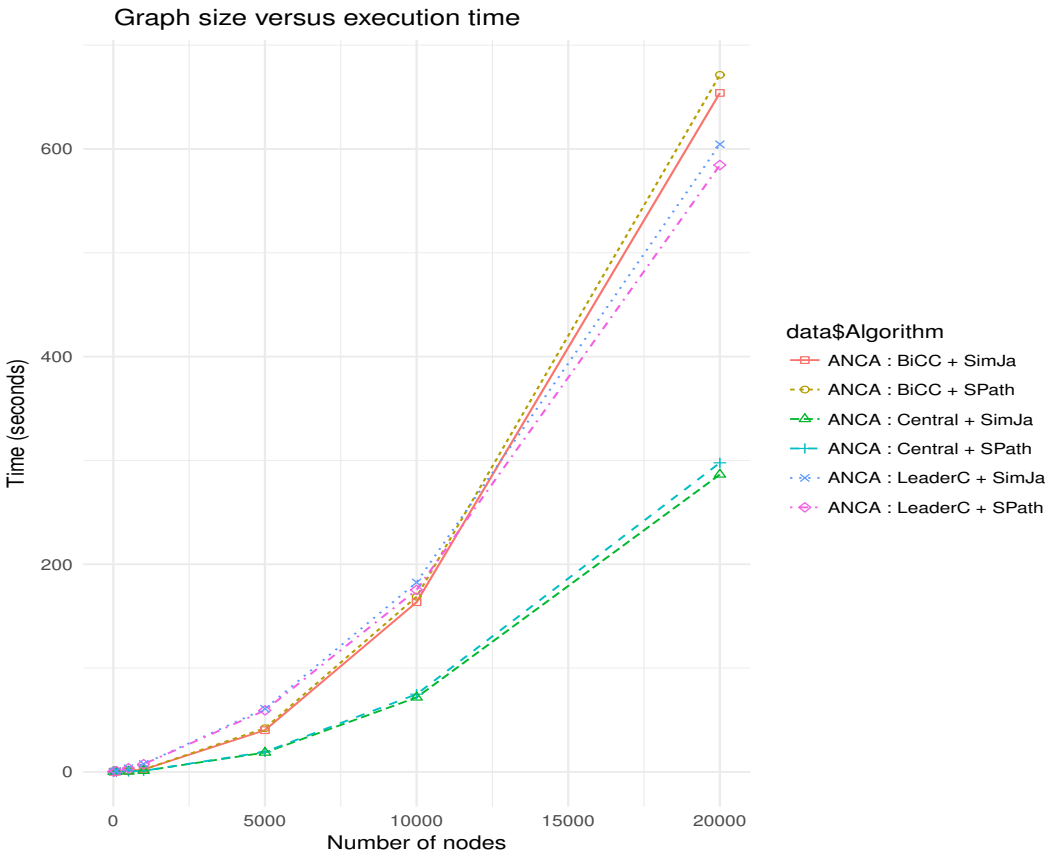


FIGURE 3.4: Scalability of n-ANCA on synthetic data.

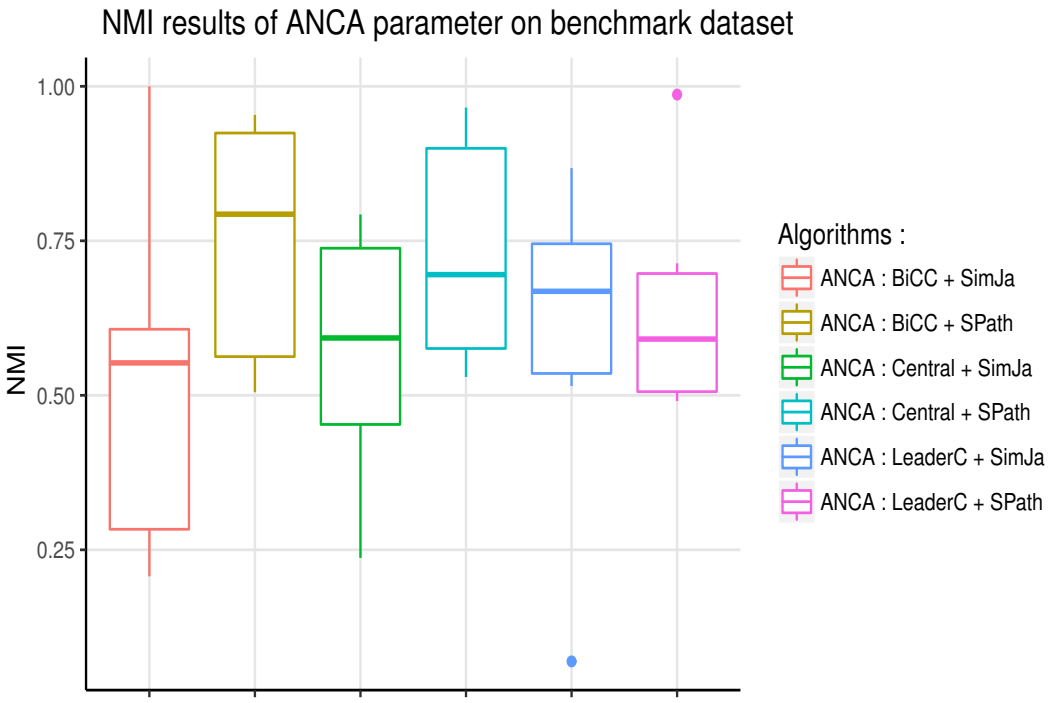


FIGURE 3.5: Evaluation of n-ANCA parameters according to normalize mutual information (NMI).

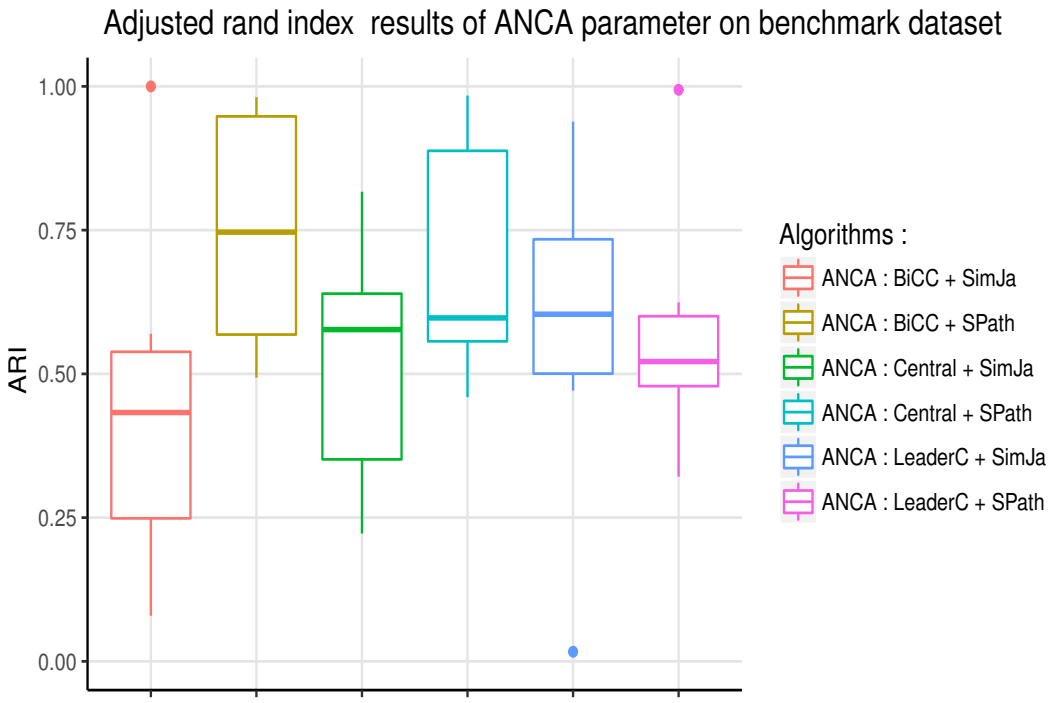


FIGURE 3.6: Evaluation of n-ANCA parameters according to Adjusted Rand Index (ARI).

Figures 3.5, 3.6 present the Normalize Mutual Information (NMI) idnex and the ARI index comparison result between n-ANCA selection methods and n-ANCA topological similarity measure on 6 synthetic data present in 3.5.2. It can be notice that n-ANCA using topological distance metrics based on path achieves a much higher NMI and ARI result compared to n-ANCA using topological distance based on neighborhood on these set of synthetic data.

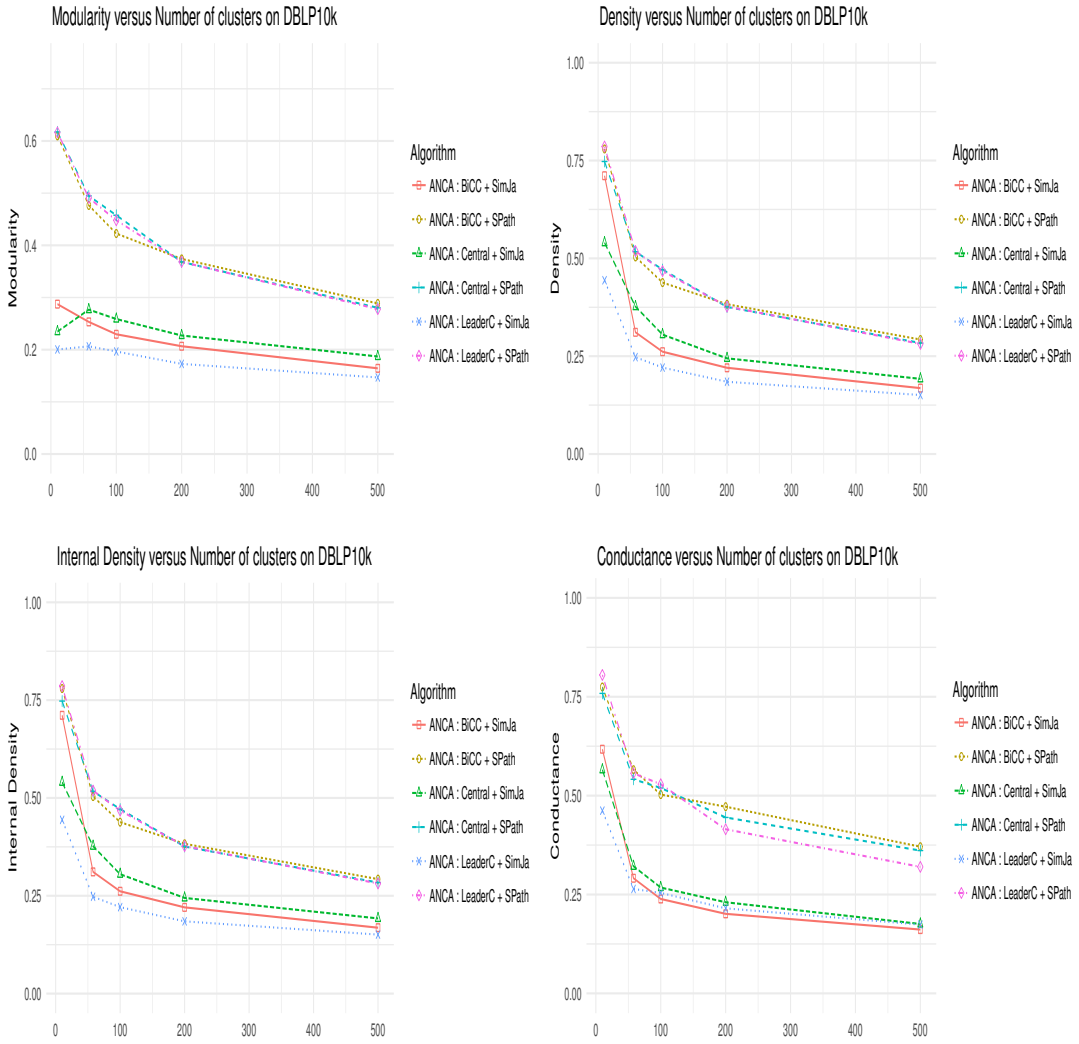


FIGURE 3.7: Cluster topological quality comparison of ANCA parameters on DBLP10k dataset

The clustering results should preserve at the same time the dense connectivity among vertices in the original graph. The density, modularity, and conductance measures are designed to analyze the topological structure of the clusters. However, the entropy is designed to analyze the attribute value homogeneity of clusters. We computed all these measures on the obtained results in order to have a better view of the obtained communities.

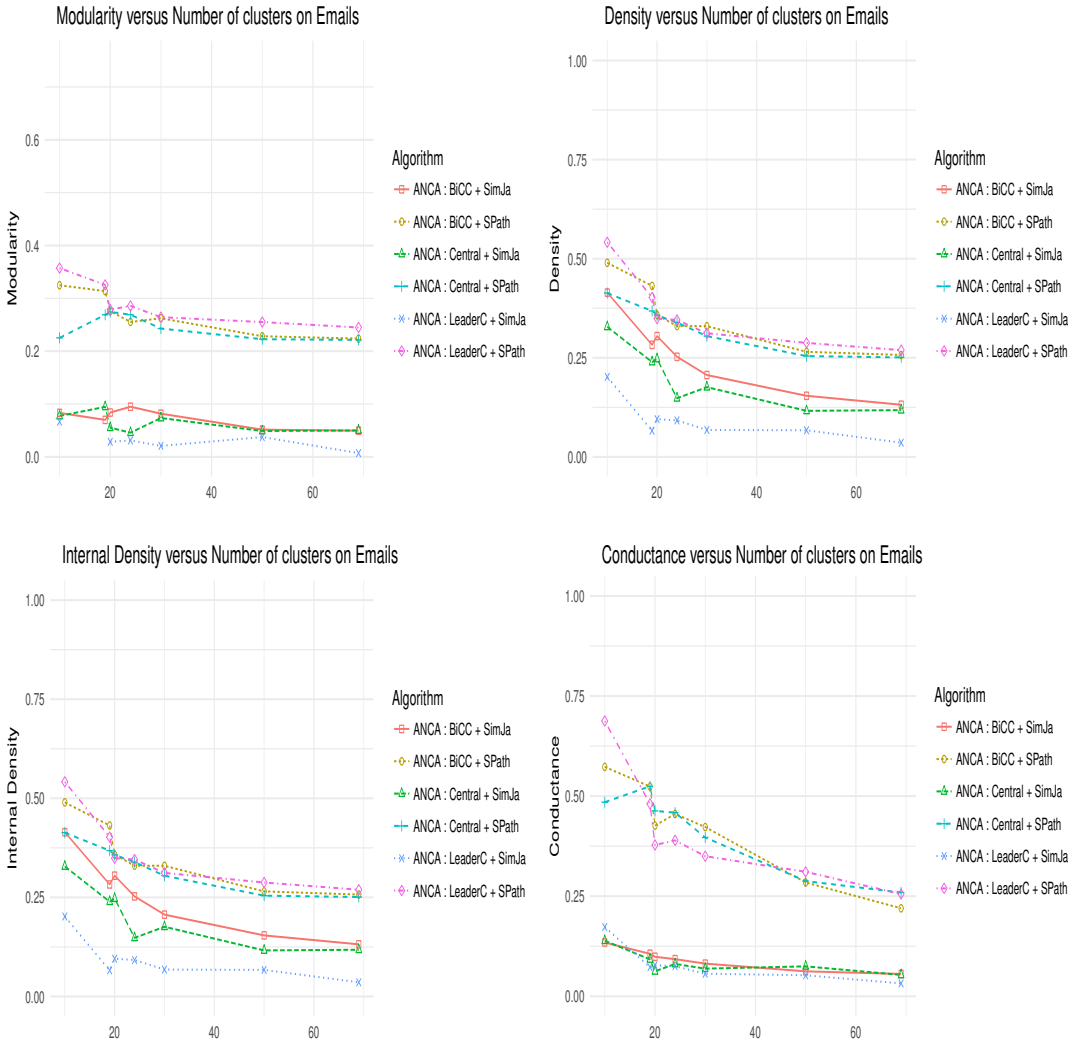


FIGURE 3.8: Cluster topological quality comparison of ANCA parameters on DBLP10k dataset

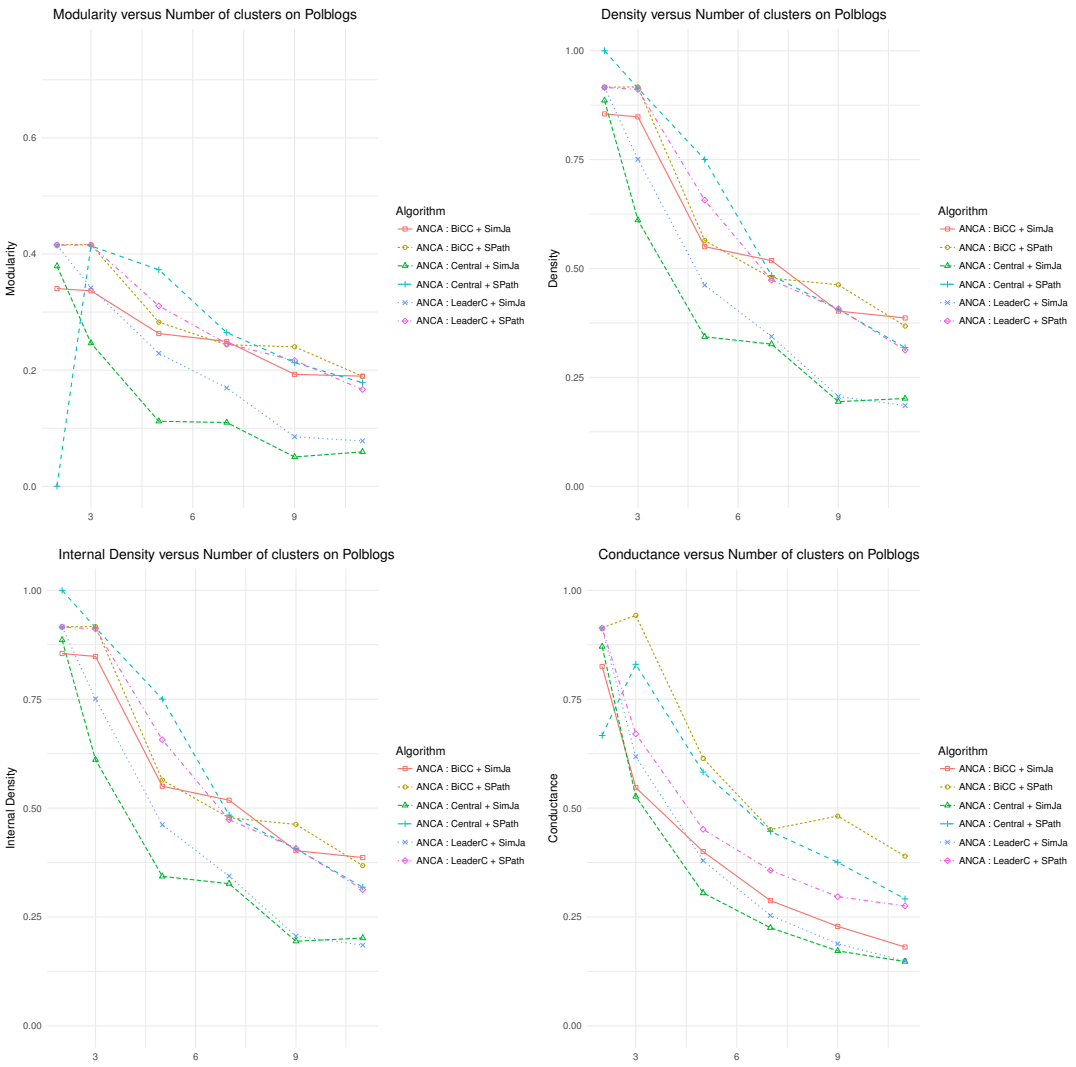


FIGURE 3.9: Cluster topological quality comparison of ANCA parameters on Polblogs dataset

Topological structure of the clustering results presented in figure 3.7, 3.9 confirm the result found previously on synthetic data that n-ANCA using topological distance metric based on path length gives a dense connectivity among vertices belonging to the same cluster.

Attribute similarity results presented in figure 3.10, show that n-ANCA using Jaccard similarity as topological distance gives a low entropy when the number of cluster k increase in Polblogs data. However, the seeds selection methods results are approximately the same with topological quality indexes or with attribute similarity.

For all next experiments, we set n-ANCA parameters as follows:

- Seeds selection methods will be the union of 15% top and 5% lower central nodes in the graph combining three different centrality which are Page rank centrality, Eigenvector centrality and degree centrality, due to the low computational time of this choice.

- Topological distance will be the Shortest Path distance.

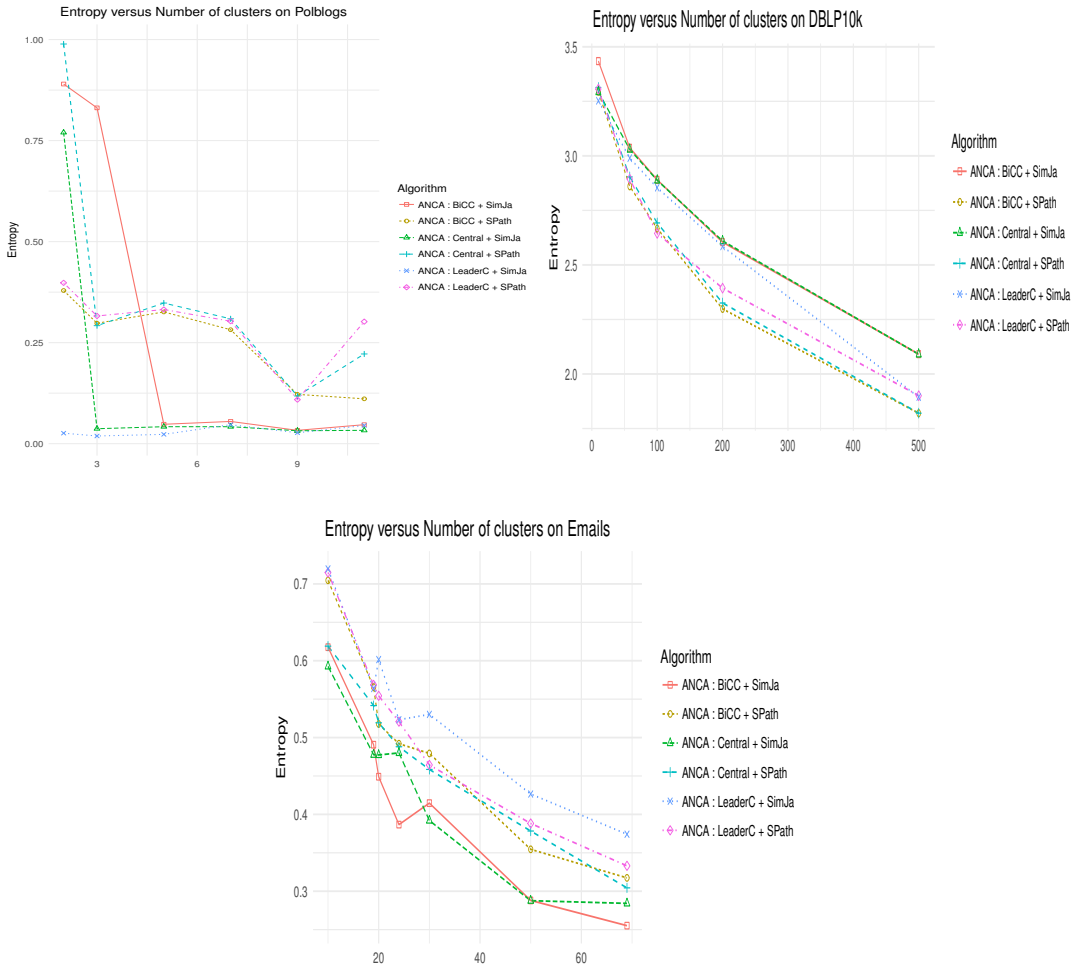


FIGURE 3.10: Entropy comparison of ANCA parameters on real datasets.

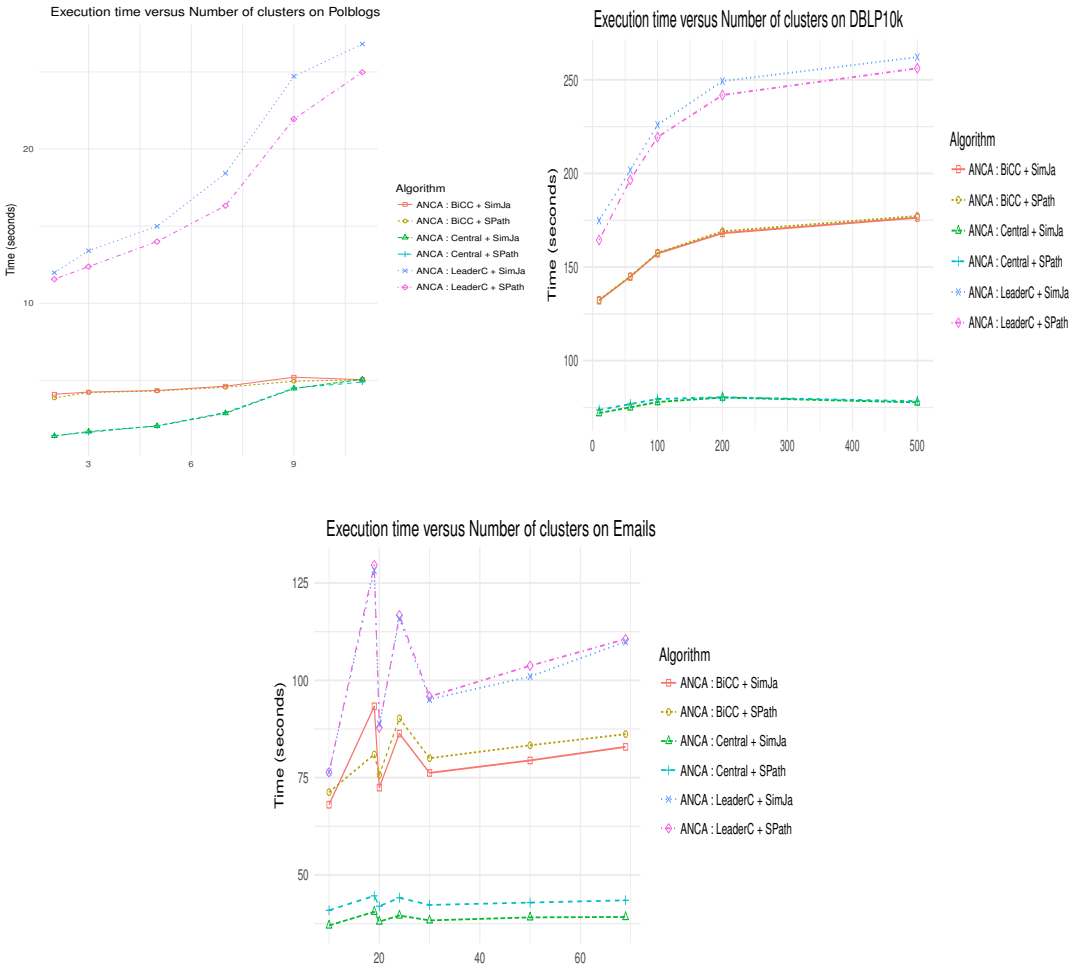


FIGURE 3.11: Execution time comparison of ANCA parameters on real datasets.

3.5.4 Comparison of ANCA with other methods on artificial data

In this second set of experiments, we compare the robustness of our method compared to other clustering algorithms applied on 100 artificial networks. These networks was taken from [31] for which the ground truth decomposition into communities is known and the results of ILouvain[2] algorithm is also given. The results of clustering using n-ANCA are compared to those found by ILouvain algorithm, Louvain algorithm, k-means algorithm and Walktrap algorithm. Figures 3.12, 3.13 summarize the comparative result respectively based on normalize mutual information (NMI) and on the adjusted rand index (ARI). The results confirm the interest of combining both kind of information in the clustering process, ILouvain and n-ANCA outperforms other approaches that take in consideration only one type of information. The proposed n-ANCA method produces better results, higher than 0.75% compared to ILouvain algorithm which use also the topological structure of the network and nodes attribute information.

[2]http://bit.ly/ILouvain

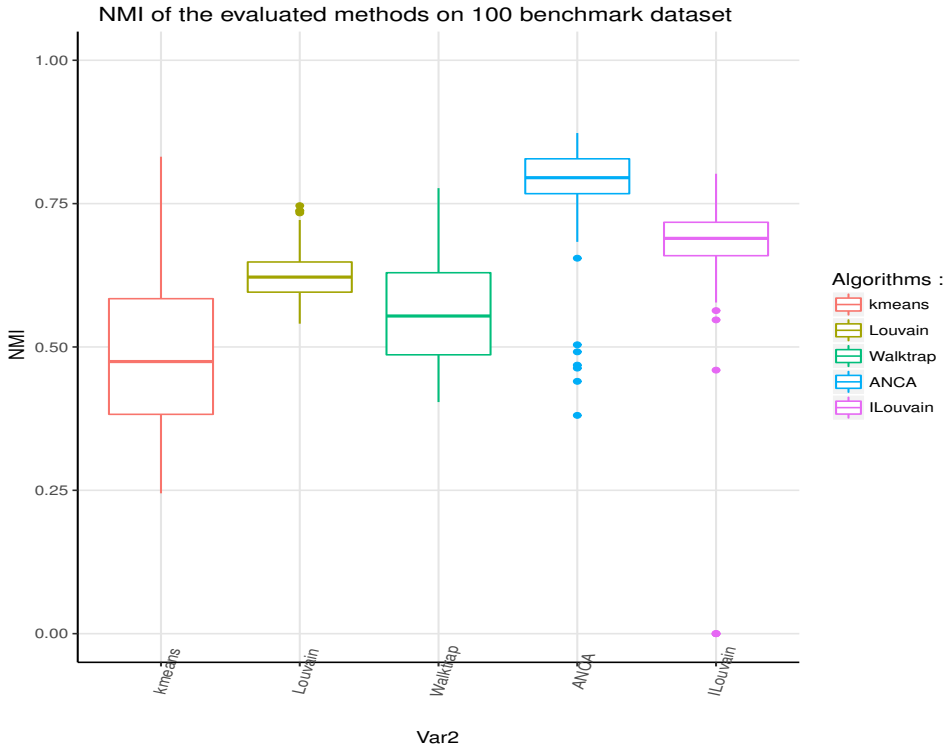


FIGURE 3.12: Cluster quality comparison on 100 synthetic data according to normalize mutual information (NMI).

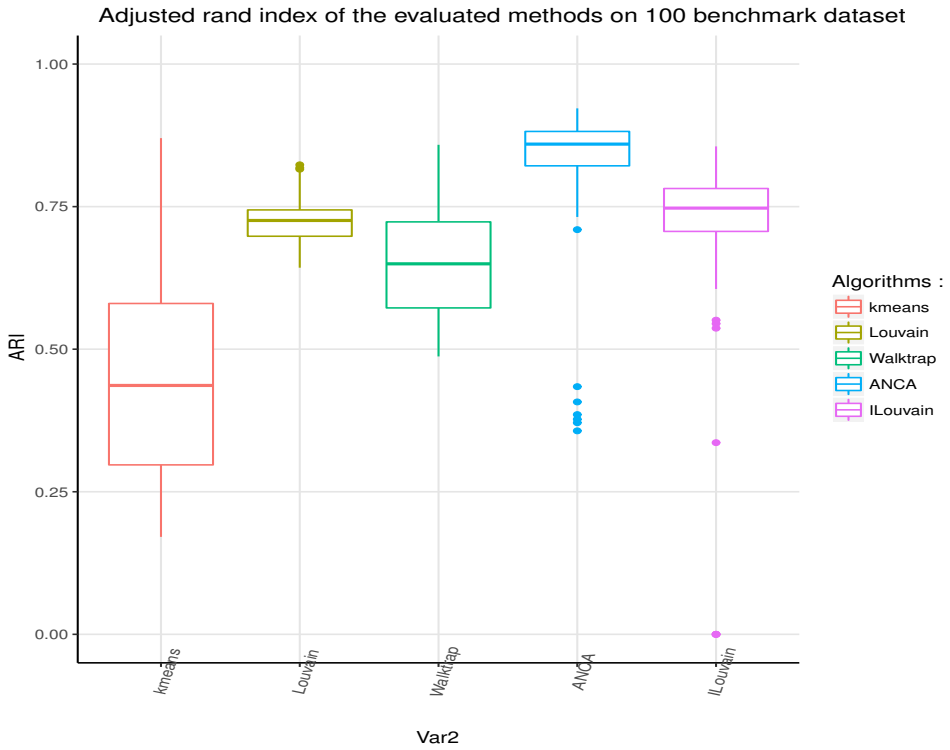


FIGURE 3.13: Cluster quality comparison on 100 synthetic data according to adjusted rand index (ARI).

3.5.5 Comparison of ANCA with other methods on real world network

For real world data, usually the ground truth is not available. However, we can use internal validity indexes to compare the clusters quality. Thus, we analyze in Figure 3.14 different properties of the clustering results determined by n-ANCA, k-means, Louvain, SAC, SA-Cluster. We set the cluster number k = 10, 100, 200 for SA-Cluster and k-means. Louvain and SAC method don't need the number of clusters as input and we use the same obtained number of clusters as input for n-ANCA algorithm in order to be able to compare these approaches.

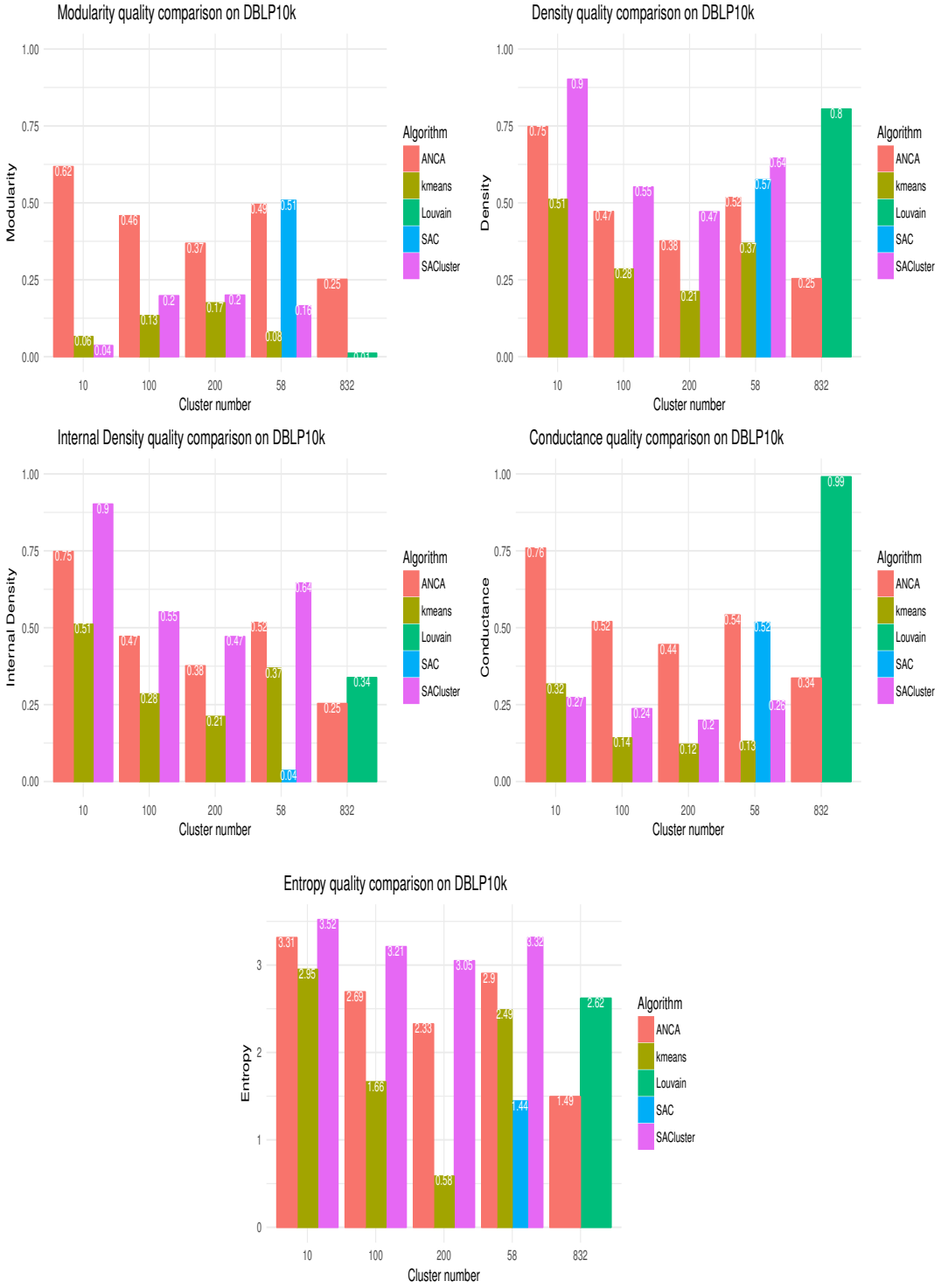


FIGURE 3.14: Cluster quality comparison on DBLP

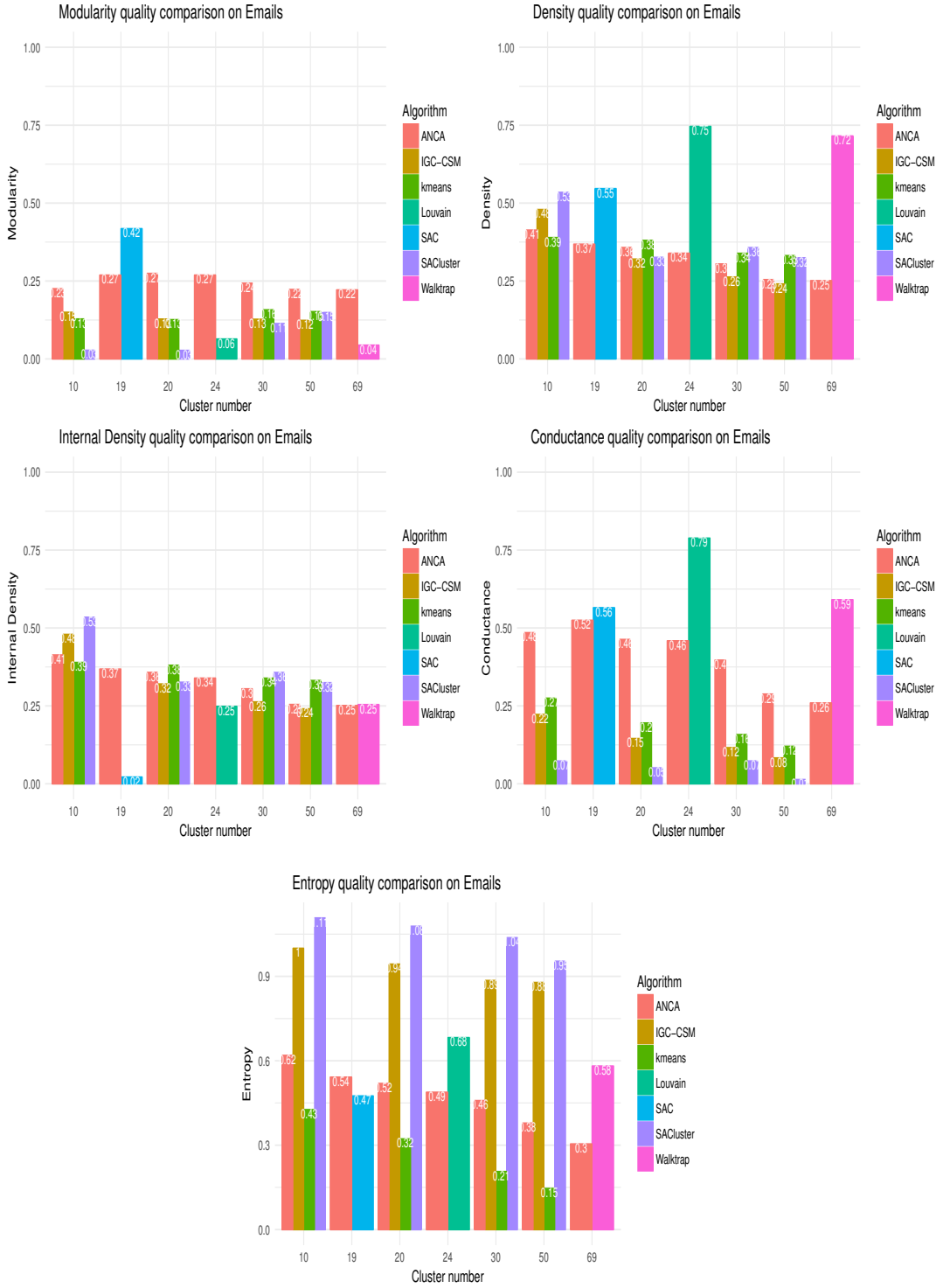


FIGURE 3.15: Cluster quality comparison on Emails

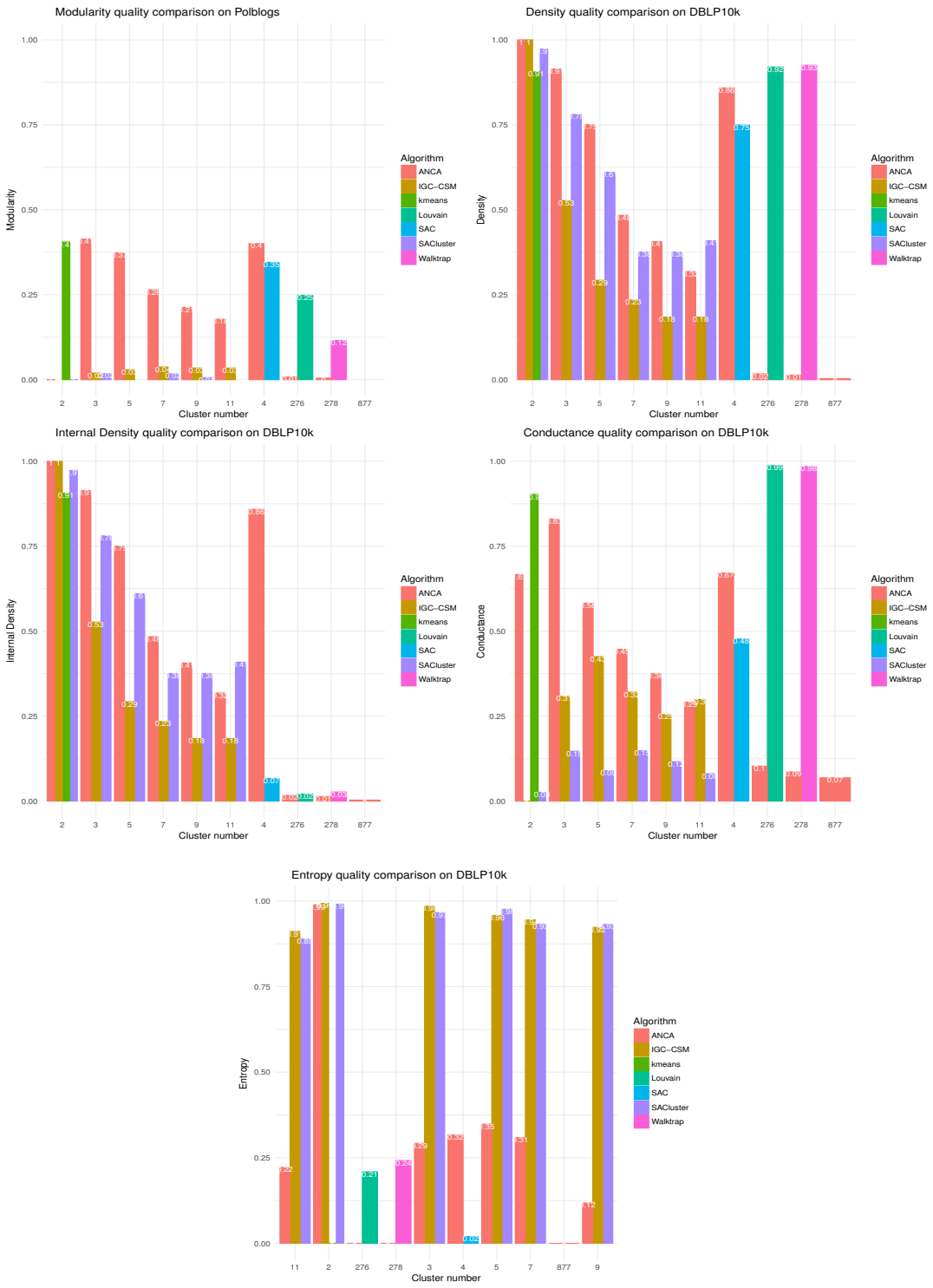


FIGURE 3.16: Cluster quality comparison on Polblogs

The best methods should preserve the dense connectivity among vertices in the original graph and low entropy values simultaneously, i.e. density, conductance and modularity which measures the connectivity around vertices. For ANCA, these topological quality measures are correlated and they decrease when k increase. The density values shows that the clusters found by SA-Cluster are more dense that then those found by ANCA. However, the modularity and the conductance values by SA-Cluster gives an opposite view. By analyzing the clusters distribution, SA-Cluster

find a large cluster and few vertices in other clusters. As shown in figure 3.14, ANCA achieves a much lower entropy than SACluster and SAC achieves an even lower entropy around 1.44.

3.6 Conclusion

In this chapter, we provide an overview of the emerging topic of clustering in attributed graph. Only few works have been proposed in the literature and their aim is to partition attributed graphs into dense clusters with vertices having similar attributes. The proposed ANCA algorithm takes into account the topological structure and attribute information simultaneously during the clustering process. We evaluate our algorithm on both synthetic data and real data, and we demonstrated that ANCA outperforms the state-of-art methods.

Chapter 4

Recommender system

Contents

4.1 Introduction

A recommendation system attempts to present items (such as movies, music, web sites, news) that are likely of interest to the user [157]. Intuitively, a recommendation system builds up a user's profile based on his/her past records, and compares it with some reference characteristics. Then, it seeks to predict the rating that a user would give to an item he/she had not yet evaluated.

Collaborative filtering is a well-known technique in recommender systems. Collaborative filtering use the known preferences of a group of users to make recommendations or predictions of the unknown preferences for other users. The fundamental assumption of collaborative filtering is that if users rate items similarly, or have similar behaviours (e.g., buying, watching, listening), then they will rate or act on other items similarly. The disadvantage of the collaborative filtering is that they suffer from the data sparsity problem when users only rate a small set of items which makes the computation of users similarity imprecise and reduce consequently the accuracy of the recommended items.

Methods in collaborative filtering can be either memory-based and model-based. Memory-based algorithms operates on the entire user-item rating matrix and generates recommendation by identifying the neighborhood of the target user to whom the recommendation will be made, based on user's past ratings. Model-based techniques use the rating data to train a model and then the model will be used to derive the recommendations.

Memory-based techniques are quite successful in real-world applications because they are easy to implement. However, there are some problems which limit the application of memory-based techniques, especially in the large-scale applications. The most problem is the sparsity of user-item rating matrix where each user only rates a small set of a large database of items. It has to compute the similarity between every pair of users (or items) to determine their neighborhoods which are based on the few overlapping ratings [14].

To overcome the weaknesses of memory-based techniques, a line of research has focused on model-based clustering techniques with the aim of seeking more efficient methods [116]. Based on ratings, these techniques group users or items into clusters, thus give a new way to identify the neighborhood.

Clustering techniques include community detection can be used to deal with this problem. In this chapter, we propose a collaborative filtering system based on clustering that predict the rate value that a user would give to an item. This approach looks, in a first step, for users having the same behavior or sharing the same characteristics. Then, use the ratings from those similar users found in the first step to predict other ratings. This chapter is structured as follow: In section 2 we provide some definitions and expose the problem statement. Section 3 describes the proposed collaborative filtering system and in section 4 we present the experimental results. We conclude the chapter in Section 5.

4.2 Definition & problem statement

Collaborative filtering techniques use a database of preferences for items by users to predict additional topics or products that a new user might like. In a typical collaborative filtering scenario, there is a list of n users $\{u_1, u_2, ..., u_n\}$ and a list of m items $\{i_1, i_2, ..., i_m\}$, and each user has a list of items which the user has rated. The ratings can either be implicit indications, such as purchases, or explicit indications, on a numerical five-star scale for example, where one and two stars represent negative ratings, three stars represent ambivalence, while four and five stars represent positive ratings. The list of ratings is converted to a user-item ratings matrix $\mathcal{R} \in \mathbb{R}^{|u|\times|i|}$. There are lots of missing values in the Users-Items matrix where users did not give their preferences for certain items and the sparsity of $\mathcal{R}$ is often larger than 99% in commercial systems [131]. Table 4.1 illustrates an example of user-item rating matrix concerning 5 users (denoted as u_1 to u_5) and 10 items (denoted as i_1 to i_{10}). Each user rates a list of items as to express his interest on each item. The goal of a recommendation system is to predict the missing rating in the matrix, and to recommend an item to a user if its predicted rating is high. In the following we denote by $\widehat{R}$ the matrix of the predicted rating.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}
u_1	1		4			2	2		3	
u_2		3	1		2			5	4	
u_3	4		2			5		5		5
u_4	1		4		2	2	2		2	
u_5		3		5		5			3	3

TABLE 4.1: User - Item matrix

The clustering techniques (i.e graph clustering) can be used to treat the problem of recommendation system. Indeed, the clustering can be seen as a generalization of the principle of collaborative filtering which we can recommend to someone the good items assessed by members of cluster which belongs to. Items can also be grouped into clusters as the reasons for their purchases allowing a customer to recommend similar products to what he liked in the past.

4.3 Proposed recommender system

The proposed recommender system is based on two step: the learning phase and the prediction step i.e. the recommendation. We detail hereafter the both steps.

4.3.1 Learning system

The goal of the proposed recommendation system is to predict the missing ratings in the Users-Items matrix. The learning step consist to find the clusters related to items and the clusters related to users based on Users-items matrix. It takes the users-items matrix as input then we apply an unsupervised learning method in order to find the clusters for users and for items. The clusters partition depend on the choice of the clustering approach.

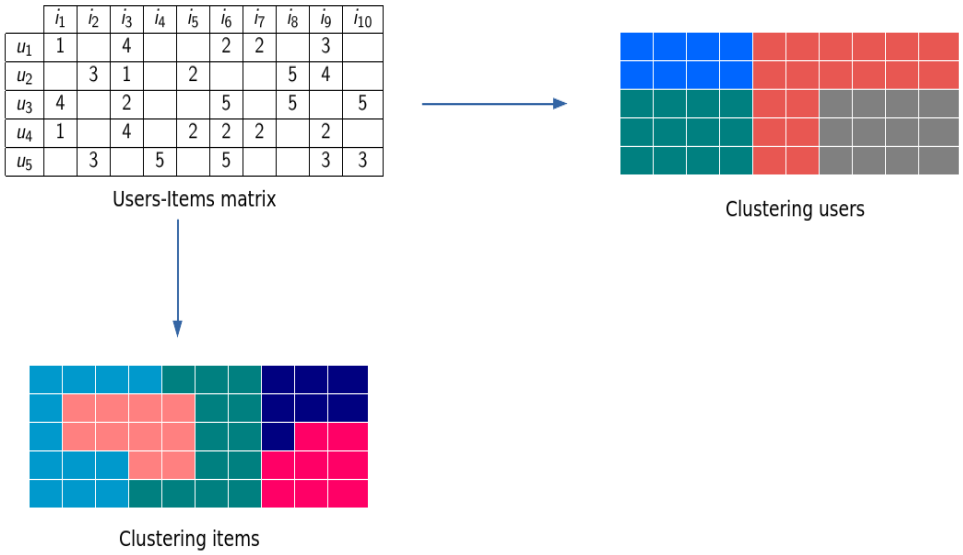


FIGURE 4.1: Learning system: Apply an unsupervised learning approach on Users and on Items.

In order to apply our clustering algorithms presented in chapter 2(resp. 3), the first step is to create the multiplex graphs (resp. node-attributed graph) related to user and the one related to items. Once created, we can apply the known clustering algorithm. We describe next how we create the multiplex graph and the node attributed graph.

Creating the multiplex network

Users-Items matrix can be seen as an adjacency matrix of valued bipartite graph. This bipartite graph is composed by items set and users set linked by different valued edges which represent user's rate preference of some items.

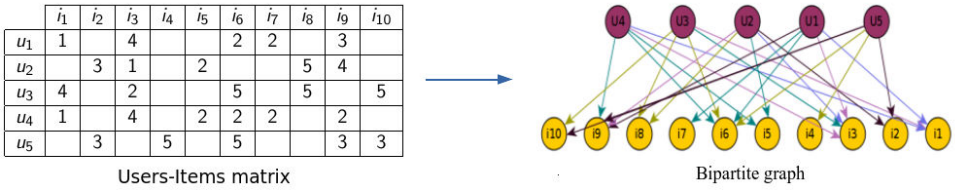


FIGURE 4.2: Learning system : Users-Items matrix can be represented as a valued bipartite graph

We separate each rate value of the bipartite graph. We obtain different bipartite graph. Edges of each bipartite graph represent one rate score. We project each bipartite graph on users and on items separately. Two user's will be connected if they have give the same rate score to the same item and two items will be linked if they were rated with the same rate value by the same user. Figure 4.3 presents the steps to create the multiplex network corresponding to users and the one corresponding to items.

Community detection in multiplex network as *muxLicod* can now be applied on these created multiplex networks.

Creating the node-attributed network

Besides considering the rate value given by the users to the items, Users-Items matrix can also be represented as adjacency matrix of a simple graph after projection on users and on items, in which we add additional information related to users (gender, age, profession, zip code, etc.) and items (price, type, etc.). This additional information will be added to the graph as nodes attribute. Each node will be associated with a number of attribute describing the node (user or item).

4.3.2 Predicting system

After obtaining the clusters of users and the items clusters, the predicting system consist to identify, for a user u and for a item i, the predicted rate's value $\widehat{R}(u,i)$ given by the user u for the item i.

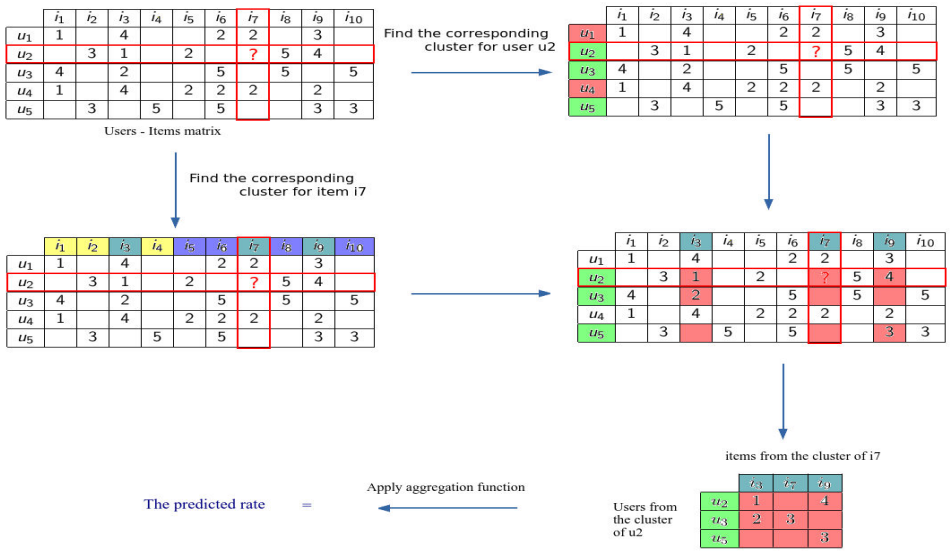


FIGURE 4.4: Predicting system

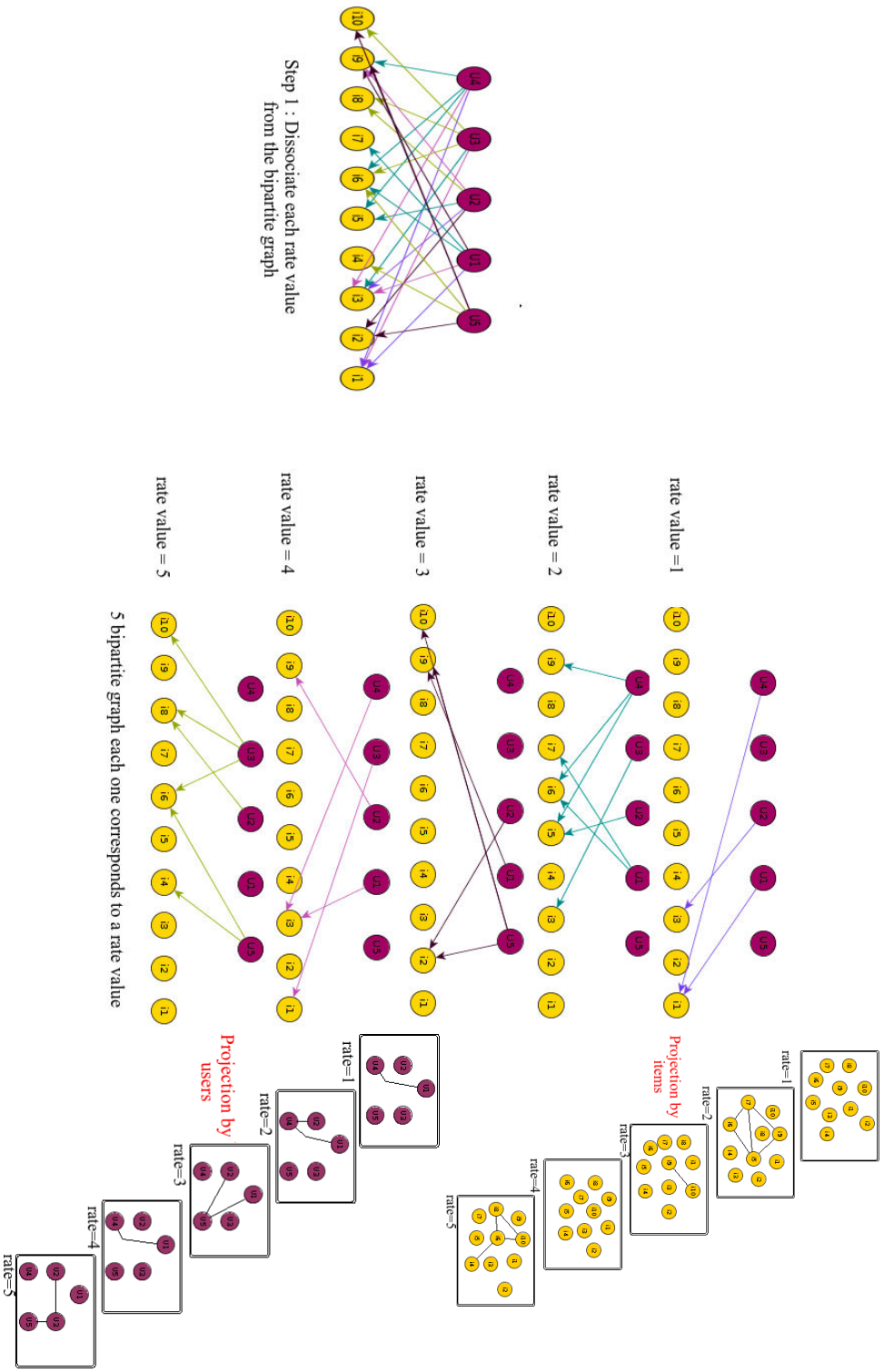


FIGURE 4.3: Creating the multiplex network

Firstly, we identify the cluster in which the user u belongs to, and also the cluster of the item i. Thereafter, the predicted rate will be computed by the aggregation between all rates found by intersection between the clusters to which the user belongs and the cluster to which the item belongs. Figure 4.4 gives an example of predicting the rate value given by the user u2 for the item i7.

4.4 Experiments

4.4.1 Datasets

We used the *MovieLens* dataset of the *GroupLens Research Center* [127] to test the performance of the proposed Recommender System using the clustering algorithms proposed in this thesis. This dataset was collected through the MovieLens web site [1], a website for movies recommendations.

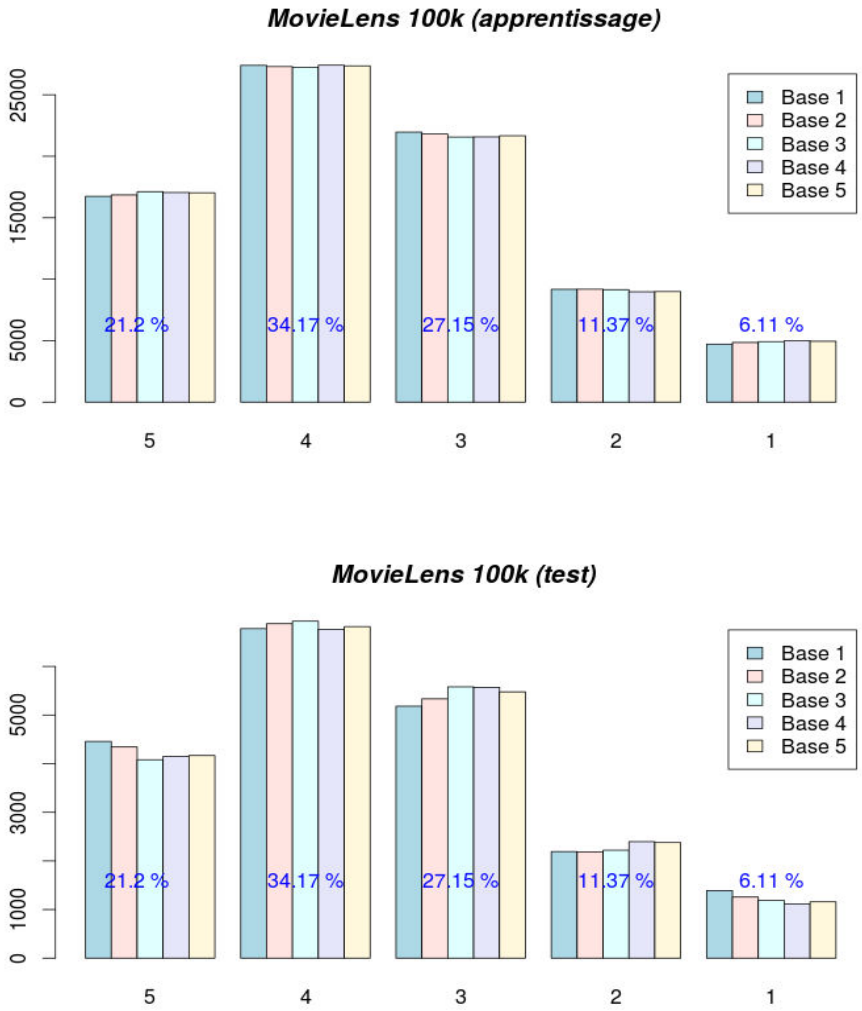


FIGURE 4.5: Rates value distribution of *MovieLens* dataset.

[1] movielens.umn.edu

The MovieLens dataset has been widely used by the scientific community to evaluate and compare recommendation algorithms. It has the advantage of being based on real votes and thus provides a good validation support.

This dataset consists of 100,000 ratings (1-5) from 943 users on 1682 movies. Each user has rated at least 20 movies. The dataset is 80%/20% splited into training and test data.
Users have the ability to share their preferences by explicitly voting for movies on a scale of integer values between 1 and 5. The matrix contains 943 users (U), 1682 movies or items (M) and 100,000 votes (Q). Thus, the vote matrix shows $93,70\%$ of missing data, considered as non-votes.

The distribution of votes is shown in Figure 4.5. Note that a large proportion of votes are 3 and 4 votes (median or satisfactory votes). There are generally few votes of unsatisfied users (1 or 2). The dataset was divided into five learning sets and five test sets called "base" and "test" respectively.

4.4.2 Evaluation criteria

There are two types of metrics to evaluate a recommendation system:

- Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate the prediction accuracy.
- Precision, Recall and F-measure to measure the recommendation quality.

Mean Absolute Error: (MAE) is defined as the average of the absolute error which is the difference between the predicted rating $\widehat{\mathcal{R}_{ij}}$ and actual rating $\mathcal{R}_{ij}$ [127]. A prediction algorithm will try to minimize the MAE, i.e. to increase the number of well predicted ratings.

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|\mathcal{R}_{ij} - \widehat{\mathcal{R}_{ij}}| \tag{4.1}$$

Root Mean Square Error: (RMSE) is similar to MAE and is biased to provide more weights to larger errors [64]. The RMSE is computed as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(\mathcal{R}_{ij} - \widehat{\mathcal{R}_{ij}})^2} \tag{4.2}$$

where N is the number of ratings.

Precision: is defined as the number of true positives (T_p) over the number of true positives plus the number of false positives (F_p). In the case of multi-class, the precision score is the mean of the precision of each subclass. Formally, if we have n class, the precision score is defined as:

$$Precision = \frac{\sum_{i=1}^{n} Precision_i}{n} \tag{4.3}$$

$$Precision_i = \frac{T_p}{T_p + F_p}$$

Recall: is defined as the number of true positives (T_p) over the number of true positives plus the number of false negatives (F_n). In the case of multi-class, the recall score is the mean of the recall of each subclass. Formally, if we have n classes, the recall score is computed by:

$$Recall = \frac{\sum_{i=1}^{n} recall_i}{n} * \tag{4.4}$$

$$*recall_i = \frac{T_p}{T_p + F_n}$$

In the following, all these measures will be used to validate the proposed Recommender System and to compare with other state-o-art models. It should be noted, that usually, it is very challenging to apply a method which will outperforms the other models on all these validation measures.

4.4.3 Results

The rating dataset is split into training set and test set, where the training set is used for model fitting and parameter tuning, and the test set serves for evaluating the recommendation system. We compare the clustering quality results obtained using the following clustering approaches:

- Community detection in multiplex network :
  - muxLicod algorithm.
  - Layer aggregation using the Louvain algorihtm denoted (LA-Louvain).
  - Layer aggregation using the Walktrap algorihtm denoted (LA-Walktrap).
  - Partition aggregation using the Louvain algorihtm denoted (PA-Louvain).
  - Partition aggregation using the Walktrap algorihtm denoted (LA-Walktrap).
- Node-attributed network : n-ANCA algorithm.
- Walktap algorithm.
- Louvain algorithm.
- $k-$means algorithm applied on users attributes information and on movies attributes information.
- GTM algorithm.
- Co-clustering algorithm.

For the GTM and topological co-clustering, we fixed a 12×12 map size. Table 4.2 summarize the clustering results using the *mode function* as an aggregation function.

The obtained results show that the recommendation system using layer aggregation approach algorithm outperforms other based on multiplex network recommendation systems in terms of prediction accuracy and in terms of recommendation quality. However, it can be noted, that the proposed n-ANCA approach outperforms others methods in terms of Precision and Recall. We observe also that the Walktrap method performs better results in terms of the MAE and RMSE, however our n-ANCA approach remains very close to these indexes (in 2nd position): only 0.03 of difference for MAE and 0.05 of difference for the RMSE index.

	MAE	RMSE	Precision	Recall
GTM	0.944	1.255	0.218	0.220
Topological co-clustering	0.929	1.256	0.255	0.209
muxlicod	0.963	1.277	0.227	0.213
LA-Louvain	0.835	1.151	0.311	0.252
LA-Walktrap	0.821	1.115	0.264	0.223
PA-Louvain	0.871	1.192	0.253	0.203
PA-Walktrap	0.880	1.202	0.270	0.201
Louvain	0.825	1.119	0.359	0.223
Walktrap	**0.789**	**1.079**	0.364	0.243
k-means	0.881	1.183	0.129	0.08
n-ANCA	0.819	1.131	**0.442**	**0.252**
n-ANCANo	0.847	1.160	0.311	0.237

TABLE 4.2: Clustering quality comparison on MovieLens dataset.

However, the goal here is not to improve all these measures as the results are similar, but to add a new evaluation method of the proposed clustering algorithm. The *muxLicod* algorithm gives worst results and this is due to the fact that *muxLicod* penalize the edges obtained only in one layer that is the case of rating value equal to 1 and 2. However, n-ANCA algorithm uses the attributes information about users and movies in the clustering process that helps to improve the precision quality as presented in Table 4.2.

4.5 Conclusions

Collaborative filtering systems recommend items based on similarity measures between users and/or items. Items recommended to a user are those preferred by its community. In this chapter, we focus on predicting the rate value for a item given by user and we proposed a new collaborative filtering system based on proposed clustering methods. The proposed Recommender System outperforms the sate-of-art methods and allows us to add a new comparison measure between proposed graph-based clustering approaches in this thesis and the state-of-art community detection methods.

The results show that the use of node information improve the recommendation quality. An extension to integrate both additional information namely nodes attribute information and edge attribute information is in perspective.

Conclusion and outlook

Contents

In this thesis, models and algorithms for combined clustering of graph and attribute data were proposed. In this chapter, we summarize the results of the thesis and provide an outlook on possible directions for future work.

4.6 Summary

This thesis is an initial attempt for constructing an integrated vision of the clustering in the attributed networks, i.e. graphs containing additional information on edges and on nodes. A graph can contains different types of edges, which represent different types of relations between vertices and also a list of attribute describing the nodes. This thesis was constructed over three main lines.
In the first part, we started by recalling additional information related to the edges. We introduced and formalized the concept of multiplex network and we defined new metrics that deals with this type of complex graphs. We proposed the *muxLicod* algorithm that compute the community detection in the multiplex graphs.

The second part of the thesis considers graphs with multi-dimensional vertex attributes. It is composed by the topological structure of the graph and a list of attributes describing each node individually, for example a node attribute describe the personal information, preferences, competences and in general, any information that can be used to describe an object.

The third part of this thesis tackle the problem of evaluating the clustering results by using a Recommender System. We introduced a new original collaborative filleting system based on clustering in attributed networks. The obtained results (internal and external criteria) shows that the proposed approaches outperforms the state-of-art methods.

4.7 Future work

Based on the results presented in this thesis, further interesting and challenging research questions arise. While in this work several different combinations of graph and attribute data were considered, other combinations of these data types are possible. For example, we could consider graphs were multidimensional vertex attributes and multi-dimensional edge attributes are given at the same time. The collaborative unsupervised learning can also be used in order to use different views of the data simultaneously.

Another challenging research direction is the analysis of attributed graphs that evolve over time. While in this thesis, we only briefly showed how the combined clustering model can be used for tracing clusters over different time steps, this only provides the first step for a more detailed analysis of the nature of the changes between the clusters of different time steps and thereby of graph evolution. This analysis, as well as extending the cluster tracing to handle clusters in subspaces of the attribute space will be an interesting challenge for future work. Furthermore, also the analysis of evolving networks with edge attributes or evolving heterogeneous networks is an interesting research direction.

Appendix A

Analysis of a world-wide board-network

A.1 Introduction

The study of interlocking directorates[1] in social sciences relies on two broad kinds of motivations: explaining the determinants of these organizational ties or/and explaining to what extent these links influence the behaviour of organizations. The structure of the corporate elite network raises also an interest *per se* when it is assumed that the circulation of elites among corporations reflects the cohesion of a social class nationally or internationally [98]. In this article we show how old and recent tools of network analysis are useful to understand the determinants and the structure of interlocking directorates. We focus on four countries: the US and the three countries with the largest GDP in Europe (France, Germany and UK). As we explain below, we also focus on the sectoral attribute of corporations as a determinant of interlocking directorates.

Our main contribution is to build an "indicator of influence" of transnational network on national network on the one hand, and between organizational networks on the other hand. The aim is to propose an indicator of the disturbance of the structure of one network when considering largest networks or other layers of ties.

To do so we first compute traditional measures of centrality to obtain a list of the top-k corporations according to centrality scores in each country. Second, we merge networks for each couple of countries and compute scores of centrality on these transnational networks. Third, we compute a distance measure between the top-k lists of national and transnational networks to assess the influence of the network of one country on the network of the other country.

This " influence indicator" is also useful to understand the disturbance of various organizational networks which are components of a multiplex network. We first compute centrality scores on either the board or the financial network of corporations and we also compute centrality scores on the multiplex network. Again, we can compute a distance between the top-k lists of simple and multiplex networks to assess the disturbance of one of them on the other.

Another contribution of this paper is to examine the distribution of corporations according to their sectoral attributes and to network measures in order to describe the diversity of national patterns of interlocking directorates. We will show that considering the structure of the corporate elites may contribute to the field of comparative political economy[2]. We will rely both on classical node-centered analysis and on the recent developments in the study of network communities.

[1] A directorate interlock is created between two firms when a director of one of them is also on the board of the other firm.

[2] See e.g. [153] or [60] for recent developments on comparative capitalism.

The article is organized as follows. The second section gives some elements of the literature on network analysis and interlocking directorates. Section three presents the sample construction, data, descriptive statistics and topological features of networks. Section four describes results on the node-oriented analysis. The fifth section deals with the community analysis. Section six focuses on the "influence indicator" between national and transnational networks or among layers of a multiplex network. Section seven concludes.

A.2 Related literature

A.2.1 Determinants of interlocking directorates

The determinants of interlocking directorates are usually understood thanks to the resource dependence theory [115]. Organizations want to secure the access to resources on which they are dependent. This may be resources from a financial or a non-financial supplier, like loans or raw materials. Resources may be also a market, for instance when only few clients exist for the activity of the corporation, or when the firm wants to limit competition in order to stabilize prices. Interlocking directorates can be seen as a collective structure of inter-organizational action. Sharing information or social relationships in board of corporations may indeed be useful to secure resources among competitors or between clients and suppliers. Sectoral similarity or dissimilarity may be consequently important to explain interlocking directorates.

Another important organizational determinant of the formation of interlocks is the ownership structure of corporations [5, 15]. Shareholders appoint and dismiss board members in general meetings. When two corporations have the same shareholders the odds that they share common board members increase since the same representative of a shareholder may seat in the board of the two companies. The financial and the human networks among the same set of corporations create a multiplex network and can be understood thanks to the recent developments in multiplex analysis.

A.2.2 The structure of national and transnational network of board of directors

Studying the structure of national and transnational network of directors aims to assess to what extent national corporate elites are able to build the basis of transnational interest groups. The first question is to know to what extent national networks are disturbed by transnational network. The "indicator of influence" we build will help us to answer this question. The main issue is to know whether these transnational networks are mainly European or transatlantic since the former indicate a geographical and political proximity (European Union) while the later indicate the possibility to collude in order to build transnational corporate rules (e.g. ability for US – and now UK – investors to invest in the European market thanks to the European passport). Of course, the cohesion of a director network is neither a necessary[3] nor a sufficient[4] condition to produce unity in the economic or political behaviour of companies. Interlocking directorates may be the result of the job market of directors

[3]Lobbying may occur thanks to business associations without interlocking directorates among corporations.

[4]Interlocking directorates will not produce unity if they do not generate similarity in the behaviour of corporations.

or the sole consequence of the resource dependence among corporations. Nevertheless, it has been proved that these social relationships are strong predictors of the uniformity of behaviour among companies, either in the political field [97] or corporate practices [16]. In this article we focus on the structure of national and transnational networks only in order to propose an indicator of influence between these networks. Previous literature has shown that transatlantic relationships may be important [23, 19, 147]. We propose here an indicator of the influence of countries in the transnational network of directors.

A.2.3 Network analysis: centrality, community, multiplex, and the influence indicator

A.3 Data, descriptive statistics and topological features of networks

A.4 Egocentric or vertices-oriented analysis

We will first analyse the distribution of corporations according to centrality measures and sectoral attributes of corporations. Secondly, we will study the ranking of corporations according to these centrality measures by focusing on the top-k companies. Focusing on the sector is relevant for the reasons presented above and also because a usual distinction in the literature is the one between Liberal Market Economies (LMEs) and Coordinated Market Economies (CMEs)[5], or between countries relying on a market-based finance and countries with a bank-based finance and stronger long term relationships between firms and financial intermediaries. France and Germany belong to the later (CMEs and bank-based finance) while UK and US belong to the former (LMEs and market-based finance). We find indeed some different patterns in the financial sector or in sectors which are more or less coordinated or liberalized depending on the country (like healthcare, utilities, energy...).

A.4.1 Distribution of corporations according to centrality measures

We explore the characteristics of the distribution of corporations according to various centrality measures computed on a dichotomized/binary network and we assess the correlation among these centrality scores thanks to a principal component analysis (PCA). We first study all corporations in the largest connected component. We then focus on corporations with minimum 2-degrees for which we can also compute a clustering coefficient. Studying companies connected with either one or multiple companies is similar to focus on the network of the largest connected component respectively with or without peripheral nodes. A change in the correlation of centrality measures between these two sub-networks would mean that the distribution of centrality scores is affected by peripheral nodes.

Variables factor maps in the first column below show that degree and closeness centralities have a large common variance in the four countries (first factor). In France and Germany, part of the variance of the eigenvector centrality is also positively correlated to these two measures, while in UK and US the variance of the

[5]This field of research studies the ways in which firms coordinate their endeavours according to the diversity of institutional complementarities among various dimensions of national economies (e.g. job market and financial market). See [52] and [153] for a presentation of research on varieties of capitalism.

eigenvector centrality adds supplementary information on the distribution of links among corporations (second factor). The second factor also shows that betweenness centrality gives supplementary information in the four countries. Finally, while eigenvector and betweenness centralities share common variance in the UK, they are partly negatively correlated in the US.

In France and Germany corporations in the core of the network (high degree and closeness centralities) are linked together frequently (high eigenvector centrality) while in UK many corporations which are not in the core of the network (low degree and closeness scores) may be notwithstanding linked to this core (high eigenvector centrality). Moreover, this connection is partly due to intermediary nodes. In the US, corporations linked to the core (high eigenvector centrality) are non-intermediary – probably peripheral – nodes. To know whether this statement is true we now do the factor analysis on the minimum 2-degrees network, i.e. excluding peripheral nodes, in order to assess to what extent it changes the distribution of centralities among corporations.

Variables factor maps in the second column below are computed on the minimum 2-degrees network, and we can therefore add the local clustering coefficient in the PCA. Correlation of degree, closeness, and betweenness centralities is similar to the one in the entire largest connected component. Clustering coefficients are unsurprisingly negatively correlated to betweenness centrality scores since the later give weight to unclosed triangle while the former weight triangle closure. The distribution of clustering coefficients among corporations appears to be partly independent from the distribution of degree and closeness centrality measures. It is worth noting that most of the variance of the eigenvector centrality in the US is now positively correlated to degree and closeness centralities, as in France and Germany. In UK the opposition between degree and closeness centrality on the one hand and eigenvector centrality on the other hand remains.

When excluding peripheral nodes, the corporate network is similar in France, Germany and US, with a strongly connected core. The distribution of degrees among corporations (figures below) confirms this difference between these three countries and the UK. In the UK, very few corporations (4%) have more than 10 connections, while 10%, 14% and 22% of corporations have 10 or more connections in Germany, France and US respectively. Closure and betweenness are homogeneously spread among high and low central firms in all countries.

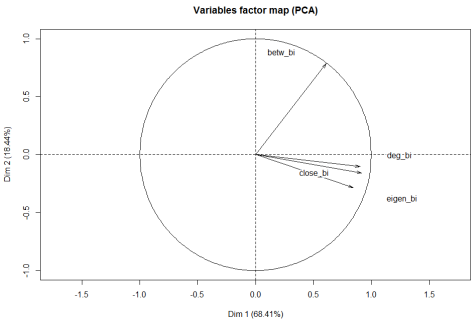


FIGURE A.1: *France minimum 1-degree*

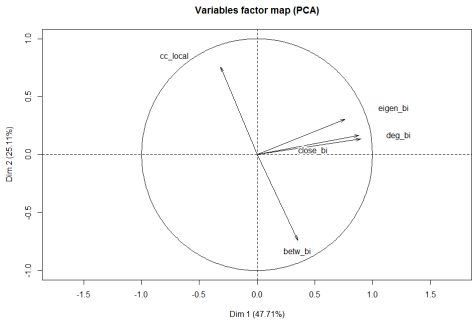


FIGURE A.2: *France minimum 2-degrees*

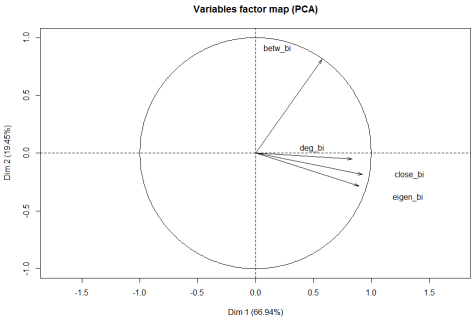


FIGURE A.3: *Germany minimum 1-degree*

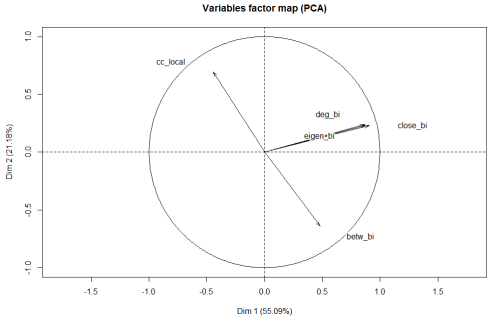


FIGURE A.4: *Germany minimum 2-degrees*

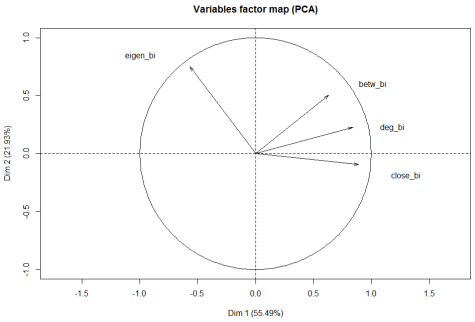


FIGURE A.5: *UK minimum 1-degree*

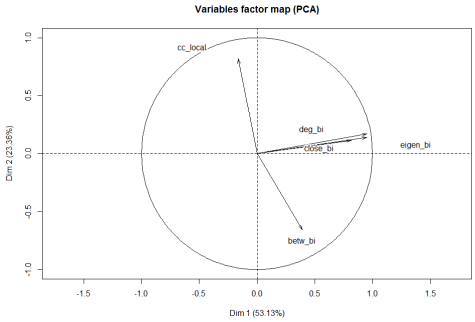


FIGURE A.6: *UK minimum 2-degrees*

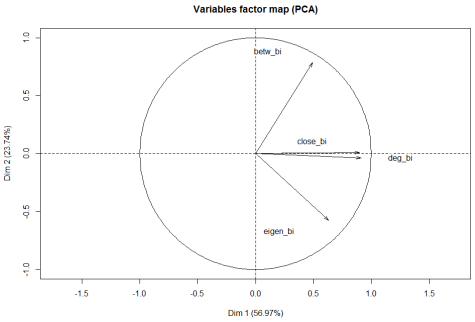


FIGURE A.7: *US minimum 1-degree*

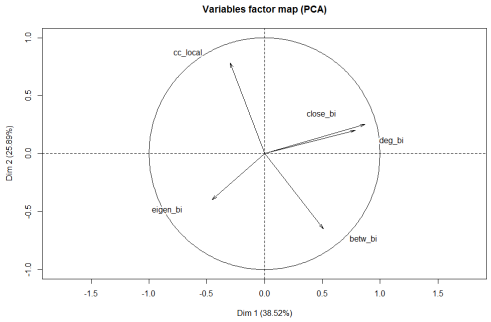


FIGURE A.8: *US minimum 2-degrees*

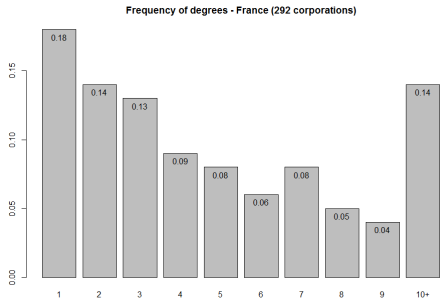


FIGURE A.9: *Frequency of degrees - France (292 corporations)*

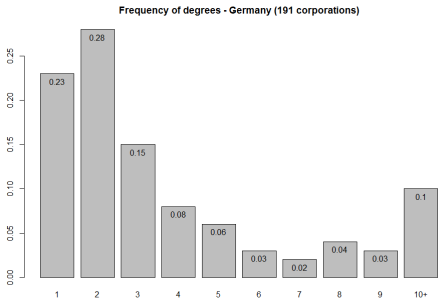


FIGURE A.10: Frequency of degrees - Germany (191 corporations)

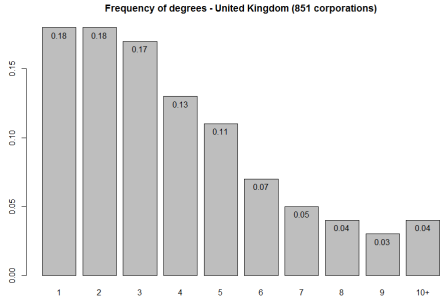


FIGURE A.11: Frequency of degrees - United kingdom (851 corporations)

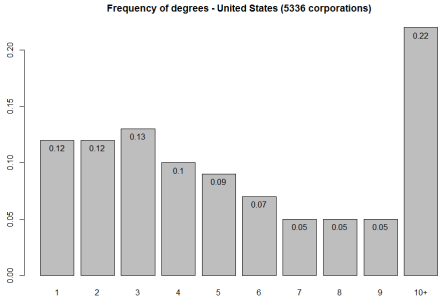


FIGURE A.12: Frequency of degrees - United States (5336 corporations)

We now focus on the four traditional measure of centrality (degree, closeness, betweenness, and eignevector) in order to assess to what extent sectoral attributes of corporations are correlated to these network measure. A common feature in all countries is that the sector Information Technology (which is about 15% of the largest connected component in the four countries) is negatively correlated to degree, closeness and betweenness. Otherwise, different patterns exist: other sectors are not correlated to centralities in Germany, except the materials one which is typical in the competitiveness of German exportations (the correlation is mainly due to the eigenvector centrality). In France and US, Utilities are positively and highly correlated to degree, closeness, and eigenvector (degree and closeness in the US, and eigenvector in France). This highly regulated industry, and the relationships it has with cities and/or State(s), may be an explanation of interlocking directorates. Financial sector is positively correlated to degree and closeness in France and negatively correlated in the US. One of the main differences between the two countries is the concentration of the financial sector (higher in France and more competitive in the US). The Healthcare industry presents an opposite pattern: negatively (positively) correlated to degree and closeness (betweenness) centralities in France, and positively (negatively) correlated to degree and closeness (betweenness) centralities in the US. We will explore deeper these differences by focusing on the top-k central firms.

	France			Germany			UK			US		
	Dim 1	*Dim 2*		*Dim 1*	*Dim 2*		*Dim 1*	*Dim 2*		*Dim 1*	*Dim 2*	
Inertia	71,05%	22,47%		65,84%	23,02%		67,15%	23,33%		67,16%	26,20%	
Degree	0,91	-0,26		0,86	-0,29		0,89	-0,24		0,91	-0,27	
Closeness	0,91	-0,29		0,87	-0,28		0,87	-0,33		0,91	-0,25	
Betweenness	0,69	0,72		0,69	0,72		0,68	0,73		0,59	0,81	
Sector												
Consumer Discretionary	19%			16%			20%			13%		
Consumer Staples	7%			4%			5%	0,48		4%		
Energy	3%			2%			7%	-0,62		8%		
Financials	11%	0,39	-0,18	21%			16%			20%	-0,23	
Healthcare	18%	-0,55		10%			9%			18%	0,46	
Industrials	18%	0,24	0,32	20%			19%	0,33		12%		
Information Technology	15%	-0,95		14%	-0,56		13%	-0,29		16%	-0,12	-0,08
Materials	5%			9%			8%			5%		
Telecommunication Services	1%			2%			2%		-0,52	1%		
Utilities	3%		-0,57	4%			2%			2%	0,54	

FIGURE A.13: Significant coefficients of correlation of variables and sectors with the two main dimensions of PCAs on degree, closeness and betweenness (significance 5% level)

	France			Germany			UK			US		
	Dim 1	*Dim 2*		*Dim 1*	*Dim 2*		*Dim 1*	*Dim 2*		*Dim 1*	*Dim 2*	
Inertia (%)	68,41	18,44		66,94	19,45		55,49	21,93		56,97	23,74	
Degree	0,90			0,83			0,84	0,23		0,91	-0,04	
Closenness	0,92	-0,16		0,92	-0,19		0,89	-0,09		0,90		
Betweenness	0,61	0,79		0,57	0,81		0,63	0,50		0,49	0,79	
Eigenvector	0,84	-0,28		0,89	-0,28		-0,57	0,75		0,63	-0,58	
Sector												
Consumer Discretionary	19%			16%			20%			13%		
Consumer Staples	7%			4%			5%	0,62	-0,25	4%		
Energy	3%			2%			7%	-0,65		8%		
Financials	11%	0,49		21%			16%			20%	-0,31	
Healthcare	18%	-0,83	0,35	10%			9%			18%	0,57	-0,17
Industrials	18%		0,33	20%			19%	0,32		12%		
Information Technology	15%	-1,01		14%	-0,67		13%	-0,31		16%	-0,23	0,08
Materials	5%			9%	0,79		8%			5%		
Telecom. Services	1%			2%			2%			1%		
Utilities	3%	0,99	-0,53	4%			2%			2%	0,45	

FIGURE A.14: Significant coefficients of correlation of variables and sectors with the two main dimensions of PCAs on degree, closeness, betweenness, and eigenvector (significance 5% level)

A.4.2 Top k firms according to centrality measures

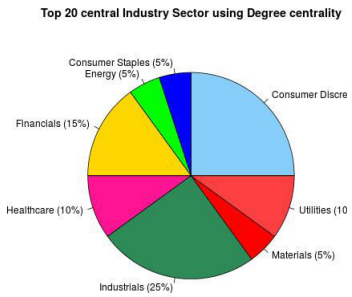


FIGURE A.15: *FRA Top 20 DEG*

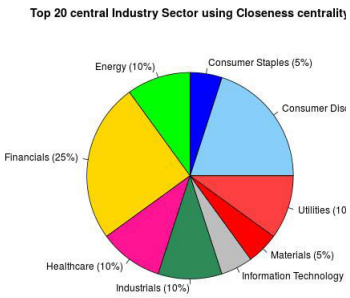


FIGURE A.16: *FRA Top 20 CLOS*

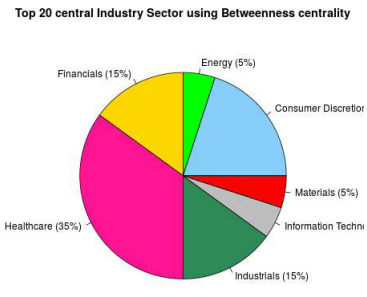


FIGURE A.17: *FRA Top 20 BETW*

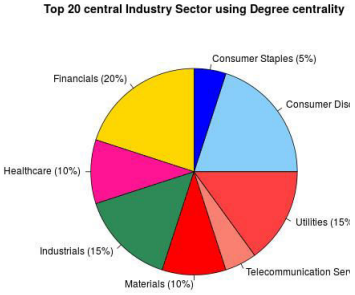


FIGURE A.18: *GER Top 20 DEG*

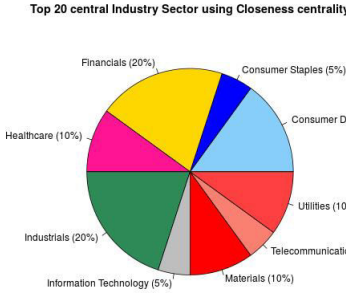


FIGURE A.19: *GER Top 20 CLOS*

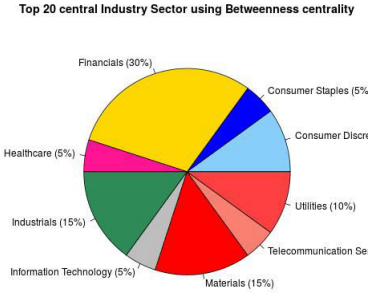


FIGURE A.20: *GER Top 20 BETW*

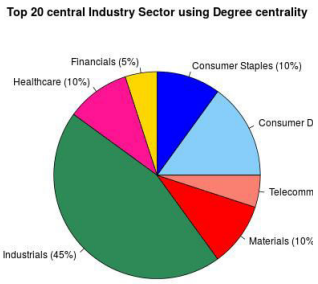


FIGURE A.21: *UK Top 20 DEG*

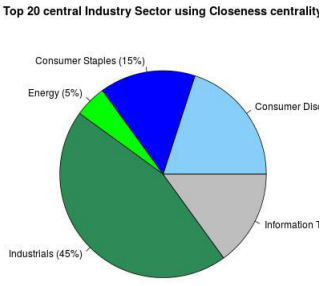


FIGURE A.22: *UK Top 20 CLOS*

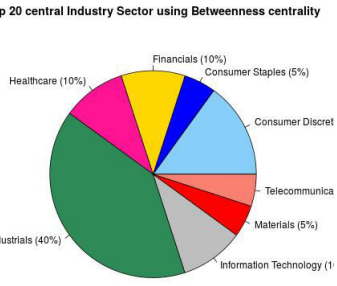
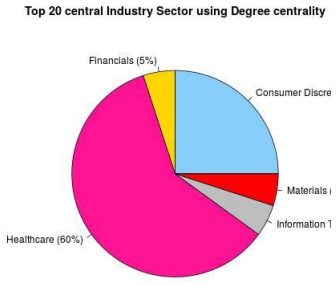
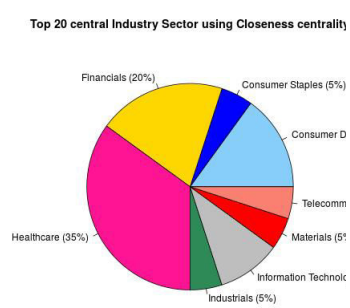
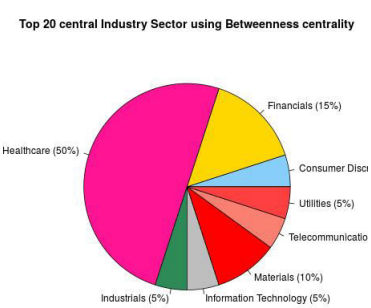


FIGURE A.23: *UK Top 20 BETW*







Top 20 firms according to degree centrality give another picture of sectoral distribution. Financial companies are important in France, but it is also the case for the sectors Industrials and Consumer discretionary (more than 65% of the top 20 while they are 48% in the largest connected components). In accordance with PCA's results, the weight of utilities is also important (10% relative to 3% in the largest connected component) and healthcare corporations represent 35% of the top 20 according to betweenness scores while they are 18% in the largest connected component.

Financial companies are important in Germany (20% for top 20 according to closeness and degree) but this is similar to the weight of financial firms in the German largest connected component (21%). Financial companies have nevertheless an important weight in top 20 betweenness scores (30%) which is congruent with their specific role in financial and interlock networks already underlined by the literature. Utilities play also an important role (15% for degrees and 10% for closeness and betweenness while they represent 4% of the largest connected component).

The specificity of UK is clearly the importance of the sector Industrials[6] (more than 40% for top 20 according to the three measures of centrality while industrial corporations are 19% of the largest connected component). A more cohesive structure is probably necessary for industrial corporations embedded in a country in which the weight of the financial sector is important (e.g. the market capitalization to GDP ratio). High centrality scores are less frequents in UK than in other countries but they are highly concentrated on a specific sector.

In the US, the Healthcare[7] sector represents more than 50% of degree and betweenness centralities (and 35% of closeness centrality) in the top-k firms, while 18% of corporations in the largest connected components belong to this sector. The financial sector is also important in the top-k firms using Betweenness and Closeness centrality, but these proportions are always inferior or equal to the share of financial companies in the largest US connected components (20%).

Finally, the relative weight of the financial sector according to degree and closeness or betweenness centrality is higher in bank-based systems and is lower in a LME like the US. Moreover, in the UK the Industrial sector is an important one according to the top-k scores of centrality. Interlocking directorates are frequent between non-financial companies and the (few) financial intermediaries in a bank-based system, while they are scarce in market-based system in which financial relationships are not mediated by specific financial intermediaries.

A.5 Communities with Licod and Louvain

To detect communities we use two different algorithms, Licod and Louvain, on both the networks of interlocking directorates and the network of common shareholders. We focus on France only to show how a community analysis and these algorithms are complementary to the previous node-centered analysis.

[6]The sector Industrials includes various sub-sectors: Capital Goods (in aerospace, electrical equipments...), Commercial and Professional Services, Transportation.

[7]Healthcare includes various sub-sectors: Healthcare Equipment and Services, and Pharmaceuticals, Biotechnology and Life Sciences.

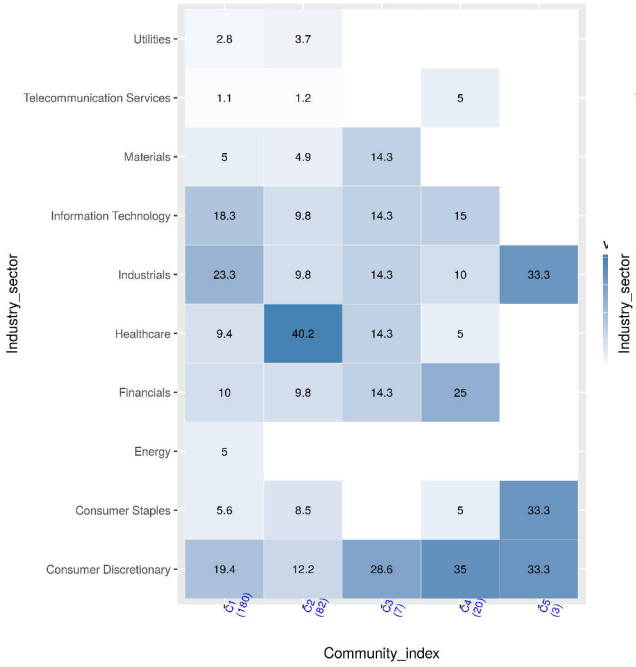


FIGURE A.27: *French Board network – LICOD*

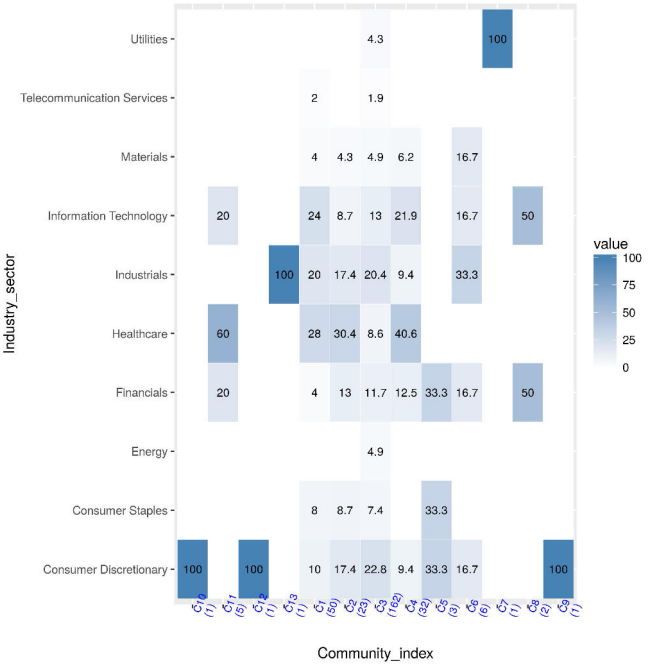


FIGURE A.28: *French Owner network – LICOD*

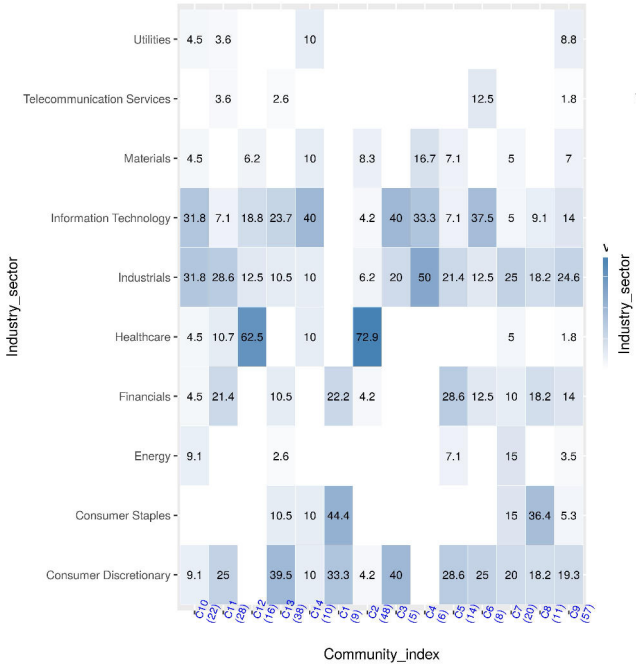


FIGURE A.29: *French Board network – LOUVAIN*

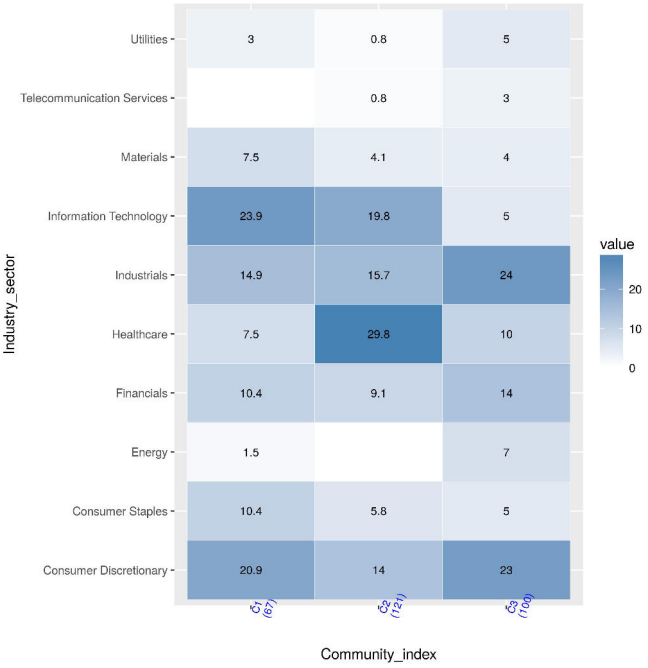


FIGURE A.30: *French Owner networks – LOUVAIN*

We first apply the Licod algorithm to the network of interlocked corporations. The first community (C1) includes the main companies of the French stock market and other mid-large and small companies. Giving the size of this community (180 constituents) it is not surprising to find the same sectoral distribution than in the largest connected component. The healthcare sector clearly dominates the other

main community (C2, 82 constituents). When we apply the Licod algorithm to the network of shareholders we also find specificity for the healthcare sector. It is about 30% in C1 and C2, and 40% in C4, with respectively 50, 23, and 32 constituents[8]. Information Technology is the other main component of C1 and C4 (more than 20%).

We now use the Louvain algorithm to identify communities. This algorithm produces an alternative representation, with more communities for network of interlocked corporations and less communities for the network of common shareholders. Let's have a look to communities in which the healthcare sector is well represented (C2 and C12, with respectively 48 and 16 constituents). While with Licod healthcare is in a community with the Consumer Discretionary, Consumer Staples, and Financial sectors, these three sectors disappear or are marginal with the Louvain algorithm. Information technology and Industrial sectors are included in the Healthcare communities with both Louvain and Licod. Finally, the Louvain algorithm on the common shareholder network also identifies a community mainly including Healthcare (32%) and Information technology (19%).

In conclusion, it means that in France the Healthcare sector, with low degree and closeness and high betweenness (see section A.4), is well connected by few common board members, and that these corporations are included in the portfolio of same sets of shareholders which basically invest in the two sectors relying heavily on intellectual property rights (healthcare and Information technology).

Depending on the purpose of the analysis, social scientists should be aware that when one wants to identify communities in social networks, the classification will depend on the density of the network. The number of community in Louvain is higher than in Licod for low density network (density=0.017 for interlocked corporations) but it is lower in high density network (density=0.437 for network of shareholders)[9]. For who wants few categories it is relevant to use the Licod algorithm for low density network and the Louvain algorithm in high density network. For who wants numerous categories it is relevant to use the Louvain algorithm for low density network and the Licod algorithm in high density network. Both are complementary in the analysis.

A.6 Influence indicator

A.6.1 The influence indicator for national and transnational networks

Our influence indicator aims to assess the influence of a domestic network on another domestic network when these two networks are merged. The indicator is a distance between the ranking of the top-k firms in the national network and the ranking of the top-k firms in the transnational network. The indicator ranges from 0 to 0.5 or to 1 depending on the probability we assign to the change of the non-observable ranking of an individual getting out the top-k list. A small value means that the national network is not influenced by the transnational network.

Comparing top-degrees-firms between two lists is not relevant to assess the proximity of two networks since we will find high ranked firms in each country while they are not necessarily connected. We think that a best measure of disruption between two networks is the influence indicator when it is computed on closeness. Among the centrality measures, closeness is the best one to understand how two networks come closer when merging two domestic networks in one transnational

[8] Three companies of the Healthcare sector are also in a small community (C11) with two other financial companies.

[9] This is true also for Germany, UK, and US.

TABLE A.1: Influence indicator

		Influences					
		FRA	GER	UK	US	FRA-GER-UK	FRA-GER-UK-US
Is influenced	FRA		0.365	0.428	0.500		
	GER	0.477		0.500	0.500		
	UK	0.487	0.398		0.483		
	US	0.104	0.136	0.178			
	FRA-GER-UK						
	FRA-GER-UK-US						

The influence indicator is a distance between the top-k firms ranked by their closeness scores in the national network and the ranking of the top-k firms in the transnational network (i.e. a couple of country). In this table, it is assumed that corporations removed from the top-k list change their rank outside this list with a probability $p = 0$. The higher the score, the higher the influence. The influence of the French-German network on France is 0.365 while it is 0.477 on Germany.

network. If we assume that corporations removed from the top-k list don't keep their rank outside this list with a probability $p = 0$, then an indicator equals to 0.5 indicates that all the top-k firms of a country are no longer in the list (or are not ranked in the same order) because they have been replaced by other corporations in the same country or by other corporations in the other country. An indicator close to zero means that top nodes in a country are also top nodes in the transnational network. In the table A.1 we can see that the indicator of influence is 0.5 for France and Germany when their networks are merged with the US. Indeed, French corporations are no longer in the top 20 of closeness scores in the French-US transnational network, while one German corporation (Deutsch Bank) which was not on the national top 20 belongs now to the top 20 of closeness scores in the German-US transnational network. Reciprocally, the US network is not really disrupted by the French network or by the German network. Nevertheless, the value of the indicator is slightly higher for Germany than for France. We can simply compare these numbers thanks to the ratio of the "relative influence" among two countries (table A.2). For example, the French-US transnational network disrupts the French network 4.8 times more than it disturbs the US network. The factor is 3.7 for Germany and 2.7 for UK. The influence of the US is the strongest one, but the relative influence shows that the closeness of Germany with the US network is higher than for France. The UK is closer to the US relative to the other main European countries.

TABLE A.2: Relative influence indicator (ratio "is influenced"/"influences")

		Influences					
		FRA	GER	UK	US	FRA-GER-UK	FRA-GER-UK-US
Is influenced	FRA		0.765	0.879	4.808		
	GER	1.307		1.256	3.676		
	UK	1.138	0.796		2.713		
	US	0.208	0.272	0.369			
	FRA-GER-UK						
	FRA-GER-UK-US						

The relative influence indicator is a ratio of the influence indicator computed in table A.1. In this table, the upper triangular matrix is the inverse of the lower triangular matrix. The French-German transnational network disrupts the German network 1.307 times more than it disturbs the French network.

The influence indicator we build can be apply not only on a top-k list but also on an entire list in order to have a measure of the disturbance of the entire network. It is also possible to compute the influence indicator to compare not only networks among countries but also various layers of a multiplex network. Such networks are frequents in social and organizational sciences.

A.6.2 The influence indicator for simple and multiplex networks

The multiplex network is made of an affiliation network of board members (directors) and an affiliation network of shareholders. We first compute closeness scores for the common director network and the common shareholder network separately. The average of the director and shareholder closeness scores for each firm gives the closeness score of each firm in the multiplex network. We then apply the influence indicator to compare the distribution of closeness scores between the layers of the multiplex network, and between each layer and the multiplex network.

Closeness scores produce a complete different ranking for each layer of the multiplex network (the influence indicator is 0.5 in the four countries, i.e. the maximum value when the probability to change the ranking outside the top-k list is $p = 0$).

The influence indicator is also 0.5 when we compare the top-k companies according to the shareholder network and the top-k firms according to the closeness scores of the average-multiplex. The multiplex network always completely disturb the shareholder network.

When comparing the top-k firms according to the board network and the top-k firm according to the average-multiplex network, it appears that the board network is influenced by the multiplex network in the following order: US (0.478), France (0.347), Germany (0.174), UK (0.054).

These results mean that the distribution of closeness scores in the UK shareholder network is dominated by the distribution of the closeness scores in the UK board network. Basically, UK corporations are widely held by a set of institutional investors which are widespread among these corporations [81, 40]. Consequently, the distribution of the closeness scores of the board network is the only relevant one to discriminate the corporations.

The US are also usually seen as a country with many widely-held firms [81]. Nevertheless, [57] has shown that listed US firms having important block holders are also frequents. Moreover, when these block holders are banks they may obtain a seat on the board of non financial companies due to their fiduciaries activities [130]. Interlocking directorates in the US are nonetheless explained by many other determinants [97, 98, 99]. Thanks to our influence indicator we can show that, contrary to the UK, the shareholder network is an important component of the inter-firm US network since it highly disturbs the US board network. Main investments of shareholders in US companies are probably not as well diversified than in UK corporations.

The distribution of the closeness scores in the French and the German board networks is partly disturbed by the distribution of the scores in the ownership network. An important feature of the ownership network in these countries is the untie of ownership links among corporations and the subsequent rise of new foreign shareholders in their ownership structure since the late nineties [5, 151, 52, 100]. The influence indicator shows that in Germany these new shareholders are more homogeneously widespread among companies than in France. Indeed, the closeness scores of the average-multiplex network –and the underlying board and shareholder networks– don't disturb so much the hierarchy of closeness scores computed on the

board network while they completely disturb the ranking of closeness scores of the shareholder network.

Appendix B

Multiplex network analysis tools : a comparative study

B.1 Introduction

In the last few years, networks have proved to be a useful tool to model structural complexity of a variety of complex systems in different domains including sociology, biology, ethology and computer science. A variety of applications and libraries have been recently proposed to ease the analysis and exploration of complex networks. Most of existing tools are designed for static simple networks, where all edges are of the same type. However, real networks are often heterogeneous and dynamic. The concept of multiplex networks has been introduced in order to ease modeling such networks. A multiplex network is defined as a multilayer interconnected graph. Each layer contains the same set of nodes but interconnected by different types of links. This simple, but expressive model, can be used to model multi-relational networks where each layer corresponds to a given type of links. It can also be readily used to model dynamic network where a layer corresponds to the network state at a given time stamp. In a formal way, a multiplex network is defined as a graph:

$$G =< V, E_1, \ldots, E_\alpha : E_k \subseteq V \times V \; \forall k \in \{1, \ldots, \alpha\} >$$

Where V is a set of nodes and E_k is a set of edges of type k. α denotes the number of layers in the multiplex.

Analysis of multiplex networks requires redefining almost all of basic complex network metrics ?. Few network analysis toolkits provide suitable primitives for multiplex network handling. In this work, we intend to fill this gap by providing *MUNA* a toolkit for multiplex network analysis building on igraph ?. *MUNA* is provided in two versions R and python that makes it readily usable for developers costumed to the use of igraph. It is available under GPL licence and can be downloaded from http://lipn.fr/ kanawati/software. A special attention in *MUNA* is made to the problem of community detection and evaluation in multiplex networks. In this paper provide a brief description of main features of *MUNA*. Related work is briefly summarized in next section. Main functions of *MUNA* are detailed in section 3. Finally we conclude in section 4.

B.2 Overview of the existing multiplex analysis library

B.2.1 Pymnet

The Multilayer Networks Library for Python called Pymnet supports general types of networks with multiple aspects such that the networks can for example have both

temporal and multiplex aspect at the same time. It supports also coupling between layers. The library is based on the general definition of multilayer networks. The main data structure underlying this definition is a global graph G_M implemented as dictionary of dictionaries. Therefore, the graph is a dictionary where for each node, e.g. (u, v, α, β), is a key and values is another dictionary containing information about the neighbours of each node. Thus, the inner dictionaries have the neighbouring nodes as keys and weights of the edges connecting them as values. This type of structure ensures that the average time complexity of adding new nodes, removing nodes, querying for existence of edges, or their weights, are all on average constant, and iterating over the neighbours of a node is linear. Further, the memory requirements in scale as $\mathcal{O}(n + l + e)$, and are typically dominated by the number of edges in the network. The Pymnet library allow to :

- Construct multilayer network *pymnet.MultilayerNetwork(aspects=0, noEdge=0, directed=False, fullyInterconnected=True)*
- Add empty layer or node in the network.
- Get layer or supra adjacency matrix.

There is no palette to analyse multilayer or multiplex network in this library and it based on the NetworkX library.

B.2.2 Gephi

Gephi is a tool that have been developed in Java, basically to explore and understand monoplex graphs. He provides an new module to facilitate exploration of multilevel graphs. Aggregate networks using data attributes or use built-in clustering algorithm.

- Expand and contract subgraphs in Meta node.
- Link and attribute clustering

B.2.3 Muxviz

Multilayer Analysis and Visualization of Networks called MuxViz provides three main methods. This library is developed with R and use igraph function. These are the principal contribution of MuxViz :

- Visualisation
- Compression of layers

B.3 *MUNA : main Features*

Functions provided by *MUNA* can be grouped into four main groups that we detail hereafter. We limit the description to functions provided in the python flavour or *MUNA* . Equivalent function are provided in the corresponding R package.

B.3.1 Multiples network generation & editing

The main class provided in *MUNA* is the class Multiplex. This class offers a simple constructor method allowing to create an empty multiplex network (with no nodes, edges or layers). Main attributes of this class are :

- `nodes` : A list of names of nodes belonging to the multiplex.
- `layers` : a list of igraph graph objects. This represents the layers of the multiplex. The node's names allow to map nodes form one layer to another (since the identifier of nodes can vary for the same node in different igraph objects representing layers.
- `name` : a string object that corresponds to the name of the multiplex network.

Different methods are provided in order to edit multiplex network object. Main editing methods are :

- `add_vertex(v_name)` : to add a vertex to all layers at once.
- `add_edge(x,y,layer,w)` : to add an edge between nodes x, and y in layer layer. w is the weight of the added edge. x and y are nodes names.
- `add_layer(g)` : to add the g igraph object as a layer. Nodes in graph g should have an attribute name used as node identifier. This method adds automatically, to all existing layers, nodes in g that do not exist in the existing layers.
- `ego_graph(node)` : returns a multiplex network centred on the given node.
- `subgraph(node_set)` : returns a multiplex network defined over the subset of nodes included in the set `node_set`.
- `summary()` : returns a string describing the main topological features of the multiplex (similarly to the summary method provided by igraph graph objects. Methods to delete vertices, edges and layers are also provided.

B.3.2 Centralities & dyadic metrics

Computing basic centralities (degree, proximity, betweenness, etc.) requires first defining basic concepts such as node's degree, node's neighbourhood and shortest paths in multiplex networks ?. We discuss these basic issues in next paragraphs.

Neighbourhood Different options can be considered to define the neighbourhood of a node in a multiplex. One simple approach is to make the union of all neighbours across all layers. Another more restrictive definition is to compute the intersection of node's neighbours sets across all layers. In [17, 77], authors define a multiplex neighbourhood of a node by introducing a threshold on the number of layers in which two nodes are linked. Formally we have:

$$\Gamma_m(v) = \{u \in V \text{ such that } count(i) > m : A_{vy}^{[i]} > 0\}$$

We extend further this definition by proposing a similarity-guided neighbourhood: Neighbours of a node v are computed as a subset of $\Gamma(v)^{tot}$ composed of

nodes having a similarity with v exceeding a given threshold δ. Using the classical Jaccard similarity function this can be formally written as follows:

$$\Gamma^{mux}(v) = \{x \in \Gamma(v)^{tot} : \frac{\Gamma(v)^{tot} \cap \Gamma(x)^{tot}}{\Gamma(v)^{tot} \cup \Gamma(x)^{tot}} \geq \delta\} \tag{B.1}$$

$\delta \in [0, 1]$ is the applied threshold.

The threshold δ allows to fine-tune the neighbourhood size ranging from the most restrictive definition (interaction of neighbourhood sets across all layers) to the most loose definition (the union of all neighbours across all layers).

Node degree: The degree of a node is defined as the cardinality of the set of direct neighbours. By defining the multiplex neighbourhood function we can define directly a multiplex node degree function. In [7] the multiplex degree of a node is defined as the entropy of node's degrees in each layer. In a formal way we can write:

$$d_i^{multiplex} = -\sum_{k=1}^{\alpha} \frac{d_i^{[k]}}{d_i^{[tot]}} log\left(\frac{d_i^{[k]}}{d_i^{[tot]}}\right) \tag{B.2}$$

The basic idea underlying this proposition, is that a node should be involved in more than one layer in order to qualify; otherwise its value is zero. The degree of a node i is null if all its neighbours are in a single layer. However, it reaches its maximum value if the number of neighbours is the same in all layers. We use this definition as kind of node's centrality computation.

Shortest path : We simply define the shortest-path between two nodes in a multiplex as the aggregation of shortest paths in the different layers:

$$SP^{multiplex}(u, v) = \mathcal{F}(SP^{[1]}(u, v), \ldots, SP^{[\alpha]}(u, v)) \tag{B.3}$$

Where $\mathcal{F}$ is an aggregation function. An example is the average function.

Shortest path: Two approaches can be applied to compute the length of the shortest-path between two nodes in a multiplex network:

- The first approach consist in computing the shortest-path in an aggregated network. The following function applies this approach : `shortest_path(,x,y,flatten_fct)` where x; y are two nodes and flatten_fct is a layer aggregation function (see hereafter).
- The second approach consists in computing an aggregation of the shortest path lengths across all layers. In the current version we simply compute the average length of shortest paths using the following function : `shortest_path_mean(x,y)`:
-

B.3.3 Layer aggregation

Flatten a multiplex network into a simple monoplex network is a frequent operation that allows to reuse some of classical network analysis tools. Different aggregation schemes can be applied. In general, the layer aggregation approach consists in transforming a multiplex network into a weighted monoplex graph G =< V;E;W > where W is a weight matrix. Different weights computation approaches can be applied. *MUNA* provides the following layer aggregation methods :

- `flatten_binary()` : implements a binary layer aggregation. Two nodes are linked in the aggregated graph if they are linked in one layer at least. Formally we have:

- `flatten_redundency()` : Returns a weighted graph: two nodes are linked in the result graph if they are linked in one layer. The edge is weighted by the number of layers in which nodes are connected. Formally we have:

- `flatten_linear_combinaison(weight_vector)`: It implements the layer aggregation approach proposed in ?: in this work authors propose to consider differently the different layers of a multiplex. The weight of a link is the resulting aggregated graph should then take into account the difference of layers contributions. A linear combination schemes can then be applied :

  the weights wk can also be learned based on user defined constraints on the clustering of some nodes into communities.

B.3.4 Community detection & evaluation

MUNA offer three different approaches for community detection in multiplex networks : `community_partitionAggregation(community_algo)` : this method applies a partition aggregation based community detection algorithm. The idea is to apply a standard community detection algorithm designed for monoplex network `(community_algo)` to each layer of the multiples. Then an ensemble clustering approach is applied on the obtained clusterings in order to compute the final community structure. The CSPA ensemble clustering approach is used for that purpose. All basic community detection approaches provided in igraph can be used here.

`community_layerAggregation(community_algo,la)`: this method applies first one of the three layer aggregation approaches discussed earlier (the selection is made via the parameter la). Then a classical community detection algorithm (one of igraph provided algorithms) can be applied to the resulting monoplex network.

`community_genLouvain()` : This applies the generalized Louvain approach using the multiplex modularity as defined in ?.

`community_muxLicod()` : This implements muxLicod, a seed-centric approach for community detection in complex networks ?. A seed-centric approach is generally structured into three main steps ?: 1) Seed computation. 2) Seed local community computation and 3) Community computation out from the set of local communities computed in step 2. Each of the above mentioned steps can be implemented applying different techniques. In the current version seeds are nodes that have a higher degree centrality than most of their direct neighbours. Notice that here we apply both multiplex neighbourhood definition and multiplex degree computation. Each node in the network ranks identified seeds in function of the length of the shortest-path linking it to each seed. Again, here the multiplex shortest path is considered. Neighbouring nodes exchange their seed ranks and merge their rankings and each node will be assigned to the community of the seed ranked at the top.

B.3.5 Community evaluation

Different community evaluation indexes are proposed in *MUNA* for assessing the quality of communities computed in multiplex networks. These are the following :

- `modularity(partition)` : returns a list of the (Newman) modularity of provided partition relative to each layer in the network.

- `modularity_multiplex(partition)` : returns the multiplex modularity of a partition as defined in ?.
- `partition_quality(partition,q_function)` : returns a list giving for each community in partition the value of the selected quality function q function. Two community quality functions are provided: the redundancy, and the complementarity as defined in ?

Redundancy criteria (ρ) **[8] :** The redundancy ρ computes the average of the redundant link of each intra-community in all multiplex layers. The intuition is that the link intra-community should be recurring in different layers. The computing of this indicator is as follows: We denote by:

- P the set of couple (u, v) which are directly connected to at least one layer.
- $\bar{\bar{P}}$ the set of couple (u, v) which are directly connected in at least two layers.
- $P_c \subset P$ represents all links in the community c
- $\bar{\bar{P}}_c \subset \bar{\bar{P}}$ the subset of $\bar{\bar{P}}$ and which are also in c.

The redundancy of the community c is given by:

$$\rho(c) = \sum_{(u,v)\in\bar{\bar{P}}_c} \frac{\parallel \{k : \exists A_{uv}^{[k]} \neq 0\} \parallel}{\alpha \times \parallel P_c \parallel} \tag{B.4}$$

The quality of a given multiplex partition is defined as follow:

$$\rho(\mathcal{P}) = \frac{1}{\parallel \mathcal{P} \parallel} \sum_{c\in\mathcal{P}} \rho(c) \tag{B.5}$$

$$\gamma(P) = \frac{1}{\|P\|} \sum_{c\in P} \gamma(c) \tag{B.6}$$

Complementarity criteria (γ) **[8] :** The complementarity γ is the conjunction of three measures :

- Variety $\mathcal{V}_c$: this is the proportion of occurrence of the community c across layers of the multiplex.

$$\mathcal{V}_c = \sum_{s=1}^{\alpha} \frac{\|\exists (i,j) \in c / A_{ij}^{[s]} \neq 0\|}{\alpha - 1} \tag{B.7}$$

- Exclusivity ε_c : this is the number of pairs of nodes, in community c, that are connected exclusively in one layer.

$$\varepsilon_c = \sum_{s=1}^{\alpha} \frac{\|\overline{P_{c,s}}\|}{\|P_c\|} \tag{B.8}$$

  with P_c : is the set of pairs (i, j) in community c that are connected at least in one layer. $\overline{P_{c,s}}$: is the set of pairs (i, j) in community c that are connected exclusively in layer s.

- Homogeneity $\mathcal{H}_c$: how uniform is the distribution of the number of edges, in the community c, per layer. The idea that the link intra-community must have a uniform distribution among all layers.

$$\mathcal{H}_c = \begin{cases} 1 & if \quad \sigma_c = 0 \\ 1 - \frac{\sigma_c}{\sigma_c^{max}} & otherwise \end{cases} \tag{B.9}$$

with

$$avg_c = \sum_{s=1}^{\alpha} \frac{\|P_{c,s}\|}{\alpha}$$

$$\sigma_c = \sqrt{\sum_{s=1}^{\alpha} \frac{(\|P_{c,s}\| - avg_c)^2}{\alpha}}$$

$$\sigma_c^{max} = \sqrt{\frac{(max(\| P_{c,d} \|) - min(\| P_{c,d} \|))^2}{2}}$$

The higher the complementarity the better is the partition. The complementarity is given by the below equation :

$$\gamma(c) = \mathcal{V}_c \times \varepsilon_c \times \mathcal{H}_c$$

B.3.6 Datasets

MUNA provide some famous multiplex networks. These datasets are the following:

- CKM Physicians Innovation Network This dataset, introduced in [28], describes relationships among physicians in the context of new drug adoption. Observed relationships include: advice, discussion and friendship.
- Lazega Law Firm Network This dataset comes from a network study of corporate law partnership reported in [88]. The network describes relationships between employees in gems of three different relationships: co-working, advice and friendship.
- Vickers Chan 7th Graders The dataset is collected by Vickers from 29 seventh grade students in a school in Victoria, Australia [150]. Students were asked to nominate their classmates on a number of tree relationships: Whom do you get on with in class? Who are your best friends class? Who would you prefer to work with?

Network	#N	#L	#E	D	CC	#clust
CKM Physicians Innovation	246	3	1552	0.015	0.184	4
Lazega Law Firm Network	71	3	2224	0.223	0.376	1
Vickers Chan 7th Graders	29	3	741	0.425	0.612	1

TABLE B.1: Main characteristics of famous multiplex networks

B.4 Conclusion

We briefly described *MUNA* a library for multiplex network analysis. The library is developed in both R and python and built on top of the igraph library. A main focus in the current version is put on community detection and evaluation in multiplex networks. The development of *MUNA* is continued in the following directions: enriching the community detection approaches by integrating new algorithms

(mainly multi-objective community detection approaches and label propagation approaches) as well as providing local community detection primitives. Another important development direction is about providing support for importing/exporting graphs from/to standard data exchange format.

Bibliography

[1] *International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011, Kaohsiung, Taiwan, 25-27 July 2011*. IEEE Computer Society, 2011.

[2] C. C. Aggarwal and C. K. Reddy, editors. *Data Clustering: Algorithms and Applications*. CRC Press, 2014.

[3] A. Amelio and C. Pizzuti. A cooperative evolutionary approach to learn communities in multilayer networks. In *Parallel Problem Solving from Nature–PPSN XIII*, pages 222–232. Springer, 2014.

[4] M. R. Anderberg. *Cluster Analysis for Applications*. New York: Academic Press, 1983.

[5] T. Auvray and O. Brossard. French connection: interlocking directorates and ownership network in an insider governance system. *Revue d'Économie Industrielle*, (154):177–206, 2016.

[6] A.-L. Barabasi, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaboration. *Physica A*, 311(3-4):590–614, 2002.

[7] F. Battiston, V. Nicosia, and V. Latora. Metrics for the analysis of multiplex networks. *CoRR*, abs/1308.3182, 2013.

[8] M. Berlingerio, M. Coscia, and F. Giannotti. Finding and characterizing communities in multidimensional networks. In *ASONAM* [1], pages 490–494.

[9] M. Berlingerio, F. Pinelli, and F. Calabrese. Abacus: frequent pattern mining-based community discovery in multidimensional networks. *Data Min. Knowl. Discov.*, 27(3):294–320, 2013.

[10] J. C. Bezdek and N. R. Pal. Some new indexes of cluster validity. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(3):301–315, 1998.

[11] C. M. Bishop, M. Svensén, and C. K. I.Williams. Gtm: The generative topographic mapping. *Neural Comput*, 10(1):215–234, 1998.

[12] V. Blondel, J. Guillaume, R. Lambiotte, and E. Mech. Fast unfolding of communities in large networks. *J. Stat. Mech*, page P10008, 2008.

[13] V. D. Blondel, J.-l. Guillaume, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:P10008, 2008.

[14] J. Bobadilla, F. Ortega, A. Hernando, and A. GutiéRrez. Recommender systems survey. *Know.-Based Syst.*, 46:109–132, July 2013.

[15] L. Bohman. Bringing the owners back in: An analysis of a 3-mode interlock network. *Social Networks*, 34(2):275 – 287, 2012.

[16] C. H. S. Bouwman. Corporate Governance Propagation through Overlapping Directors. *Review of Financial Studies*, 24(7):2358–2394, 2011.

[17] P. Brodka and P. Kazienko. *Encyclopedia of Social Network Analysis and Mining*, chapter Multi-layered social networks. Springer, 2014.

[18] G. Brown. Ensemble learning. In C. Sammut and G. Webb, editors, *Encyclopedia of Machine Learning*, pages 312–320. Springer US, 2010.

[19] V. Burris and C. L. Staples. In search of a transnational capitalist class: Alternative methods for comparing director interlocks within and between nations and regions. *International Journal of Comparative Sociology*, 53(4):323–342, 2012.

[20] D. Cai, Z. Shao, X. He, X. Yan, and J. Han. Mining hidden community in heterogeneous social networks. In *ACM-SIGKDD Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD'05)*, Chicago, IL, Aug 2005.

[21] T. Caliński and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[22] R. J. G. B. Campello and E. R. Hruschka. On comparing two sequences of numbers and its applications to clustering analysis. *Information Sciences*, 179(8):1025–1039, 2009.

[23] W. Carroll and J. P. Sapinski. The global corporate elite and the transnational policy-planning network, 1996-2006: A structural analysis. *International Sociology*, 25:501–538, 2010.

[24] C.-H. Cheng, A. W. Fu, and Y. Zhang. Entropy-based subspace clustering for mining numerical data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 84–93, New York, NY, USA, 1999. ACM.

[25] H. Cheng, Y. Zhou, and J. X. Yu. Clustering Large Attributed Graphs: A Balance between Structural and Attribute Similarities. *ACM Trans. Knowl. Discov. Data*, 5(2):12:1—-12:33, 2011.

[26] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. *A short introduction to computational social choice*. Springer, 2007.

[27] F. R. Chung. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[28] J. Coleman, E. Katz, and H. Menzel. The diffusion of an innovation among physicians. *Sociometry*, 20(4):253–270, 1957.

[29] P. M. Comar, P.-N. Tan, and A. K. Jain. Simultaneous classification and community detection on heterogeneous network data. *Data Min. Knowl. Discov.*, 25(3):420–449, 2012.

[30] D. Combe, C. Largeron, E. O. Egyed-Zsigmond, and M. Géry. Combining relations and text in scientific network clustering. *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1280–1285, 2012.

[31] D. Combe, C. Largeron, M. Géry, and E. Egyed-Zsigmond. I-louvain: An attributed graph clustering method. In *Advances in Intelligent Data Analysis XIV*, pages 181–192. Springer, 2015.

[32] G. Cordasco and L. Gargano. Label propagation algorithm: a semi-synchronous approach. *IJSNM*, 1(1):3–26, 2012.

[33] A. D Gordon. *Classification, Second Edition (Monographs on Statistics and Applied Probability, 82)*. 06 1999.

[34] T. Dang and E. Viennet. Community detection based on structural and attribute similarities. In *International Conference on Digital Society (ICDS)*, pages 7–12, 2012.

[35] D. B. D.L. Davies. A cluster separation measure. *IEEE Trans. Pattern Anal. Machine Intell.*, 1 (4):224–227, 1974.

[36] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, 2001.

[37] J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104, 1974.

[38] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622. ACM, 2001.

[39] B. Everitt, S. Landau, and M. Leese. *Cluster analysis.* Arnold ; Oxford University Press, May 2001.

[40] M. Faccio and L. H. P. Lang. The ultimate ownership of Western European corporations. *Journal of Financial Economics*, 65(3):365–395, 2002.

[41] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[42] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. 2000.

[43] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[44] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[45] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences,*, 104(1):36, 2007.

[46] S. Fortunato and C. Castellano. Community structure in graphs. pages 1141–1163, 2009.

[47] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[48] D. F. Gleich and C. Seshadhri. Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In Q. Yang, D. Agarwal, and J. Pei, editors, *KDD*, pages 597–605. ACM, 2012.

[49] A. Goder and V. Filkov. Consensus clustering algorithms: Comparison and refinement. In J. I. Munro and D. Wagner, editors, *ALENEX*, pages 109–117. SIAM, 2008.

[50] B. H. Good, Y.-A. de Montjoye, and A. Clauset. Performance of modularity maximization in practical contexts. *Physical Review E*, 81(4):046106, 2010.

[51] B. H. Good, Y.-A. de Montjoye, and A. Clauset. The performance of modularity maximization in practical contexts. *Physical Review*, E(81):046106, 2010.

[52] P. A. Hall and D. Soskice, editors. *Varieties of capitalism: The institutional foundations of comparative advantage*. Oxford University Press, New-York, 2001.

[53] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[54] P. Hansen and B. Jaumard. Cluster analysis and mathematical programming. *Math. Program.*, 79:191–215, 1997.

[55] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[56] T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[57] C. G. Holderness. The Myth of Diffuse Ownership in the United States. *Review of Financial Studies*, 22(4):1377–1408, 2009.

[58] L.-C. Huang, T.-J. Yen, and S. cho Timothy Chou. Community detection in dynamic social networks: A random walk approach. In *ASONAM* [1], pages 110–117.

[59] P. Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.

[60] G. Jackson and R. Deeg. The long-term trajectories of institutional change in european capitalism. *Journal of European Public Policy*, 19(8):1109–1125, 2012.

[61] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[62] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[63] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[64] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender Systems - An Introduction*. Cambridge University Press, 2010.

[65] D. Jin, D. He, D. Liu, and C. Baquero. Genetic algorithm with local search for community mining in complex networks. In *ICTAI (1)*, pages 105–112. IEEE Computer Society, 2010.

[66] P. E. Jupp and K. V. Mardia. Maximum likelihood estimators for the matrix von mises-fisher and bingham distributions. *The Annals of Statistics*, pages 599–606, 1979.

[67] R. Kanawati. Licod: Leaders identification for community detection in complex networks. In *SocialCom/PASSAT*, pages 577–582, 2011.

[68] R. Kanawati. Licod : Leaders identification for community detection in complex networks. *IEEE.*, page 577–582, 2013.

[69] R. Kanawati. Détection de communautés dans les réseaux multiplexes : état de l'art. *RNTI*, RNTI-A-7:20, Avril 2014. to appear.

[70] R. Kanawati. Seed-centric approaches for community detection in complex networks. In G. Meiselwitz, editor, *6th international conference on Social Computing and Social Media*, volume LNCS 8531, pages 197–208, Crete, Greece, June 2014. Springer.

[71] R. Kanawati. Seed-centric approaches for community detection in complex networks. In G. Meiselwitz, editor, *6th international conference on Social Computing and Social Media*, volume LNCS 8531, pages 197–208, Crete, Greece, June 2014. Springer.

[72] R. Kanawati. Yasca: An ensemble-based approach for community detection in complex networks. In Z. Cai, A. Zelikovsky, and A. G. Bourgeois, editors, *COCOON*, volume 8591 of *Lecture Notes in Computer Science*, pages 657–666. Springer, 2014.

[73] R. Kanawati. Empirical evaluation of applying ensemble methods to ego-centered community identification in complex networks. *Neurocomputing*, 150, B:417–427, February 2015.

[74] R. Kanawati. Empirical evaluation of applying ensemble methods to ego-centred community identification in complex networks. *Neurocomputing*, 150:417–427, 2015.

[75] D. R. Karger. Global min-cuts in rnc, and other ramifications of a simple min-out algorithm. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, pages 21–30, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.

[76] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series in Probability and Statistics, 1990.

[77] P. Kazienko, P. Brodkadka, and K. Musial. Individual neighbourhood exploration in complex multi-layered social network. In *Web Intelligence/IAT Workshops*, pages 5–8, 2010.

[78] R. R. Khorasgani, J. Chen, and O. R. Zaiane. Top leaders community detection approach in information networks. In *4th SNA-KDD Workshop on Social Network Mining and Analysis*, Washington D.C., 2010.

[79] T. Kohonen. *Self-organizing Maps*. Springer-Verlag Berlin, Berlin, 1995.

[80] T. Kohonen. *Self-organizing Maps*. Springer Berlin, 2001.

[81] R. La Porta, F. Lopez-de-Silanes, and A. Shleifer. Corporate Ownership Around the World. *Journal of Finance*, 54(2):471–517, 1999.

[82] R. Lambiotte. Multi-scale modularity in complex networks. In *WiOpt*, pages 546–553. IEEE, 2010.

[83] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84(6):066122, 2011.

[84] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *CoRR*, abs/1107.1, 2011.

[85] A. Lancichinetti and S. Fortunato. Consensus clustering in complex networks. *CoRR*, abs/1203.6093, 2012.

[86] A. Lancichinetti and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, Physical Review E(4):046110, 2008.

[87] C. Largeron, P.-N. Mougel, R. Rabbany, and O. R. Zaïane. Generating Attributed Networks with Communities. *Plos One*, 10(4):e0122777, 2015.

[88] E. Lazega. *The collegial phenomenon : the social mechanisms of cooperation among peers in a corporate law partnership*. Oxford university press, Oxford, 2001.

[89] I. X. Leung, P. Hui, P. Lio, and J. Crowcroft. Towards real-time community detection in large networks. *Phys. Phy. E.*, 79(6):066107, 2009.

[90] S. Li, Y. Chen, H. Du, and M. W. Feldman. A genetic algorithm with local search strategy for improved detection of community structure. *Complexity*, 15(4):53–60, 2010.

[91] T. Li, S. Ma, and M. Ogihara. Entropy-based criterion in categorical clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 68. ACM, 2004.

[92] X. Liu and T. Murata. Community Detection in Large-Scale Bipartite Networks. In *Web Intelligence*, pages 50–57. IEEE, 2009.

[93] H. Lou, S. Li, and Y. Zhao. Detecting community structure using label propagation with weighted coherent neighborhood propinquity. *Physica A: Statistical Mechanics and its Applications*, 392(14):3095 – 3105, 2013.

[94] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[95] A. Mahajan and D. Teneketzis. Optimal performance of networked control systems with nonclassical information structures. *SIAM Journal on Control and Optimization*, 48(3):1377–1404, 2009.

[96] M. Meila. Comparing clusterings by the variation of information. In B. Schölkopf and M. K. Warmuth, editors, *COLT*, volume 2777 of *Lecture Notes in Computer Science*, pages 173–187. Springer, 2003.

[97] M. S. Mizruchi. *The structure of corporate political action: Interfirm relations and their consequences*. Harvard University Press, Cambridge, United States, 1992.

[98] M. S. Mizruchi. What Do Interlocks Do? An Analysis, Critique, and Assessment of Research on Interlocking Directorates. *Annual Review of Sociology*, 22(1):271–298, 1996.

[99] M. S. Mizruchi. *The fracturing of the American corporate elite*. Harvard University Press, 2013.

[100] F. Morin. A Transformation in the French Model of Shareholding and Management. *Economy and Society*, 29(1):36–53, 2000.

[101] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010.

[102] T. Murata. Detecting communities from tripartite networks. In M. Rappa, P. Jones, J. Freire, and S. Chakrabarti, editors, *WWW*, pages 1159–1160. ACM, 2010.

[103] T. Murata. Modularity for heterogeneous networks. In M. H. Chignell and E. Toms, editors, *HT*, pages 129–134. ACM, 2010.

[104] T. Murata. Comparison of inter-layer couplings of multilayer networks. In *Proceedings of the 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 448–452. IEEE Computer Society, 2015.

[105] W. Nawaz, K.-U. Khan, Y.-K. Lee, and S. Lee. Intra graph clustering using collaborative similarity measure. *Distributed and Parallel Databases*, pages 583–603, 2015.

[106] J. Neville, M. Adler, and D. Jensen. Clustering relational data using attribute and link information. In *Proceedings of the text mining and link analysis workshop, 18th international joint conference on artificial intelligence*, pages 9–15, 2003.

[107] M. E. J. Newman. Mixing patterns in networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 67(2 Pt 2):026126, 2003.

[108] M. Ovelgönne and A. Geyer-Schulz. Cluster cores and modularity maximization. In G. I. Webb, B. Liu, C. Zhang, D. Gunopulos, and X. Wu, editors, *ICDM Workshops*, pages 1204–1213. IEEE Computer Society, 2010.

[109] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik. Validity index for crisp and fuzzy clusters. *Pattern recognition*, 37(3):487–501, 2004.

[110] L. Pan, C. Dai, W. Chongjun, X. Junyuan, and M. Liu. Overlapping community detection via leaders based local expnsion in social networks. In *Proceddings of the 24th IEEE Conferenceon Tools with Artficial Intelligence (ICTA'12)*, 2012.

[111] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos. Community detection in social media - performance and application considerations. *Data Min. Knowl. Discov.*, 24(3):515–554, 2012.

[112] C. Peng, T. G. Kolda, and A. Pinar. Accelerating community detection by using k-core subgraphs. *CoRR*, abs/1403.2226, 2014.

[113] J. Pfeffer and G. Salancik. *The external control of organizations: A resource dependence perspective*. Stanford Business Books, 2003. 1978 for the first edition.

[114] M. C. Pham, Y. Cao, R. Klamma, and M. Jarke. A clustering approach for collaborative filtering recommendation using social network analysis. 2011.

[115] C. Pizzuti. Boosting the detection of modular community structure with genetic algorithms and local search. In S. Ossowski and P. Lecca, editors, *SAC*, pages 226–231. ACM, 2012.

[116] C. Pizzuti. A multiobjective genetic algorithm to find communities in complex networks. *IEEE Trans. Evolutionary Computation*, 16(3):418–430, 2012.

[117] P. Pons and M. Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10(2):191–218, 2006.

[118] P. Pons and M. Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, pages 191–218, 2006.

[119] M. Pujari. *Link Prediction in Large-scale Complex Networks (Application to bibliographical Networks). (Prévision de liens dans des grands graphes de terrain (application aux réseaux bibliographiques))*. PhD thesis, Paris 13 University, France, 2015.

[120] M. Pujari and R. Kanawati. Link prediction in multiplex bibliographical networks. *International Journal of Complex Systems in Science*, 2, 2013.

[121] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. In *Proc. Natl. Acad. Sci. USA*, pages 2658–2663, 2004.

[122] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76:1–12, September 2007.

[123] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.

[124] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1), 2006.

[125] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186. ACM, 1994.

[126] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *Eur. Phys. J. Special Topics*, 13:178, 2009.

[127] R. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics.*, 20:53–65, 1987.

[128] J. A. C. Santos and A. S. Rumble. The American keiretsu and universal banks: Investing, voting and sitting on nonfinancials' corporate boards. *Journal of Financial Economics*, 80(2):419–454, 2006.

[129] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[130] M. Seifi and J.-L. Guillaume. Community cores in evolving networks. In A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, and S. Staab, editors, *WWW (Companion Volume)*, pages 1173–1180. ACM, 2012.

[131] M. Seshadri, S. Machiraju, A. Sridharan, J. Bolot, C. Faloutsos, and J. Leskovec. Mobile call graphs: beyond power-law and lognormal distributions. In Y. Li, B. Liu, and S. Sarawagi, editors, *KDD*, pages 596–604. ACM, 2008.

[132] D. Shah and T. Zaman. Community detection in networks: The leader-follower algorithm. In *Workshop on Networks Across Disciplines in Theory and Applications, NIPS*, 2010.

[133] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[134] K. Steinhaeuser and N. V. Chawla. Community detection in a large real-world social network. In *Social computing, behavioral modeling, and prediction*, pages 168–175. Springer, 2008.

[135] A. Strehl and J. Ghosh. Cluster ensembles: a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[136] L. Subelj and M. Bajec. Robust network community detection using balanced propagation. *European Physics Journal B*, 81(3):353–362, 2011.

[137] S. Sumathi and S. Sivanandam. Data mining tasks, techniques, and applications. *Introduction to Data Mining and its Applications*, pages 195–216, 2006.

[138] D. D. Suthers, J. Fusco, P. K. Schank, K.-H. Chu, and M. S. Schlager. Discovery of community structures in a heterogeneous professional online network. In *HICSS*, pages 3262–3271. IEEE, 2013.

[139] K. T. *Self-organizing Maps*. Springer Berlin, 2001.

[140] L. Tang and H. Liu. *Community Detection and Mining in Social Media*. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers, 2010.

[141] L. Tang and H. Liu. *Community Detection and Mining in Social Media*. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers, 2010.

[142] S. Thrun, L. K. Saul, and B. Schölkopf, editors. *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*. MIT Press, 2004.

[143] A. P. Topchy, A. K. Jain, and W. F. Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(12):1866–1881, 2005.

[144] S. Tsironis, M. Sozio, and M. Vazirgiannis. Accurate spectral clustering for community detection in mapreduce. In E. Airoldi, D. Choi, A. Clauset, K. El-Arini, and J. Leskovec, editors, *Frontiers of Network Analysis: Methods, Models, and Applications*, Lake Tahoe, Nevada, December 2013. NIPS workshop.

[145] K. Van der Pijl. *The making of an Atlantic ruling class*. Verso Books, 2014.

[146] L. Vendramin, R. J. Campello, and E. R. Hruschka. On the comparison of relative clustering validity criteria. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 733–744. SIAM, 2009.

[147] A. Verma and S. Butenko. Network clustering via clique relaxations: A community based approach. In D. A. Bader, H. Meyerhenke, P. Sanders, and D. Wagner, editors, *Graph Partitioning and Graph Clustering*, volume 588 of *Contemporary Mathematics*, pages 129–140. American Mathematical Society, 2012.

[148] M. Vickers and S. Chan. Representing classroom social structure. Victoria Institute of Secondary Education, 1981.

[149] A. Weber. An empirical analysis of the 2000 corporate tax reform in Germany: Effects on ownership and control in listed companies. *International Review of Law and Economics*, 29(1):57–66, 2009.

[150] J. J. Whang, D. F. Gleich, and I. S. Dhillon. Overlapping community detection using seed set expansion. In Q. He, A. Iyengar, W. Nejdl, J. Pei, and R. Rastogi, editors, *CIKM*, pages 2099–2108. ACM, 2013.

[151] M. A. Witt and G. Jackson. Varieties of capitalism and institutional comparative advantage: A test and reinterpretation. *Journal of International Business Studies*, 47(7):778–806, 2016.

[152] Z. Yakoubi and R. Kanawati. Applying leaders driven community detection algorithms to data clustering. In *The 36th Annual Conference of the German Classification Society on Data Analysis, Machine Learning and Knowledge Discovery (GfKI'12)*, Hildesheim, Germany, August 2012.

[153] Z. Yakoubi and R. Kanawati. Licod: Leader-driven approaches for community detection. *Vietnam Journal of Computer Science*, 1(4):241–256, 2014.

[154] J. Yang, J. McAuley, and J. Leskovec. Community detection in networks with node attributes. *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 1151–1156, 2013.

[155] X. Yang, Y. Guo, Y. Liu, and H. Steck. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1–10, 2014.

[156] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.

[157] Y. Zhang, J. Wang, Y. Wang, and L. Zhou. Parallel community detection on large networks with propinquity dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 997–1006. ACM, 2009.

[158] Y. Zhao and G. Karypis. Comparison of agglomerative and partitional document clustering algorithms. Technical report, MINNESOTA UNIV MINNEAPOLIS DEPT OF COMPUTER SCIENCE, 2002.

[159] Y. Zhou, H. Cheng, and J. X. Yu. Clustering large attributed graphs: An efficient incremental approach. *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 689–698, 2010.