Algorithmique des réseaux socio-sémantiques pour la visualisation par points de vue des communautés en ligne

Juan David Cruz Gomez

N° d'ordre : 2012telb0269

Sous le sceau de l'Université européenne de Bretagne

Télécom Bretagne

En habilitation conjointe avec l'Université de Rennes I

École Doctorale – MATISSE

Socio-semantic Networks Algorithm for a Point of View Based Visualization of On-line Communities

---

Thèse de Doctorat

Mention : « Informatique »

Présentée par **Juan David Cruz Gómez**

Département : LUSSI

Laboratoire : Lab – STICC Pôle : CID

Directeur de thèse : François Poulet
Encadrant de thèse : Cécile Bothorel

Soutenue le 10 décembre 2012

Jury :

M. Guy Melançon, Professeur, Université de Bordeaux I, France (Rapporteur)
Mme. Clémence Magnien, Chercheur, CNRS, LIP6 - UPMC, France (Rapporteur)
M. Bernie Hogan, Chercheur, Internet Institute Oxford University, Royame-Uni (Examinateur)
Mme. Marie-Aude Aufaure, Professeur, École Centrale de Paris, France (Examinateur)
M. Ralf Klamma, Professeur, RWTH Aachen University, Allemagne (Examinateur)
M. François Poulet, Maître de conférences, Université de Rennes I - IRISA (Directeur de thèse)
Mme. Cécile Bothorel, Maître de conférences, Télécom - Bretagne (Encadrant)

A Amparo, Humberto y Magda, mi familia,
las personas que más quiero en el mundo.

Acknowledgements

I would like to thank first all the members of the LUSSI Department at Telecom - Bretagne for these three years of help, discussion and ideas you gave me. Moreover, thanks for being patient with my French speaking skills that during these years you, kindly, helped to improve. So I offer my sincere thanks to Ghislaine Le Gall who helped me even before I arrived in France, Yvon Kermarrec the head of the department for being always available for answering any question, Gilles Coppin, Patrick Meyer and Nicolas Jullien for the superb and clever questions that helped me refine my work, Sébastien Bigaret and Philipe Tanguy for helping me setting my server up. I also want to thank Karine Roudaut for the nice chats and advises. I would also like to thank Vivianne Guillerm, Martine Besnard and Fabienne Guyader from the Scientific Direction at Telecom - Bretagne who helped me throughout all the PhD solving all the questions and problems that I came up to with. I do not like to do lists because I always forget someone, and if this is the case, please forgive me.

Next I would like to thank all the friends I met during these years, Alexander, Claire, Delphine, Elisa, Fahad, Hélène, Ibrahim, Isabel, Javier, Jonathan, Jorge, Juan Pablo, July, Justine, Kedar, Liliana, Luiz de Patricia, Marine, Marius, Oscar, Patricia, Sandra, Santiago, Soraya, Suellen, Vinicius, Yenny and a lot more who made this time a great learning journey and helped me to have some relax in hard times. Of course, to my parents and my sister who have always supported and encouraged me to go after my dreams.

This research work was financed by a Future & Ruptures scholarship from the Insititut Mines-Telecom that made this work financially feasible.

I would also thank the jury committee who took the time to read and understand this work and then to come to the "end of the world" to presence my defense and ask nice questions which have enriched my work and helped me to improve.

And last but definitely not least thank to Cécile Bothorel and François Poulet, my advisors. Cécile and François 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, specially those in French. I bet it was a bit tough.

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

Abstract

Socio-semantic Networks Algorithm for a Point of View Based Visualization of On-line Communities

Juan David Cruz Gómez
Department LUSSI, Telecom - Bretagne
10 décembre 2012

With the explosion of social networking sites like Facebook, LinkedIn and Pinterest, people face a new way to communicate and share large amounts of information transmitted in, near, real time. To manipulate this information efficiently, researchers face important theoretical and technical challenges, for example, storing and loading data into memory, processing capacity and latency. In social networks such as Facebook, it is possible to identify two types of information: first, a set of actors described individually by their profiles which contains information like age, sex, interests and other personal information; the second type of information is a set of ties, which in the Facebook case are friendship ties, describing the connections between actors in the network, or which is called the structure of the network. These informations can be seen as two dimensions of the same social network: a structural dimension that represents the connections between actors and a compositional dimension that describes each actor individually.

Classic approaches deal with each type of data separately, on the one hand the structural information is represented as a graph in which the actors represent the nodes and the relationships represent the edges. All the analyses are reduced to graph mining problems. On the other hand, the profile information is typically represented as vectors of features that are used by algorithms to summarize, cluster, classify or extract patterns reducing the problem to classic data mining problems. In the case of community detection problem, using either the structural dimension or the compositional dimension will reduce the scope of the information that can be extracted: for the first case the communities will have well connected, but not necessarily similar, nodes and for the second case, the communities will be composed of similar but loosely connected nodes.

We want to use both dimensions to find communities of well-connected and similar nodes. This approach requires a new definition of community that includes the two dimensions presented before and a new community detection model based on that definition. This PhD Thesis introduces a new definition of community on social networks and presents a new model for detecting and visualizing communities of well-connected and similar nodes. This model starts by defining the notion of *point of view* that divides the compositional dimension in such a way the network can be analyzed from different perspectives. Then the model uses the compositional dimension, defined by the point of view, to influence the community detection process integrating both dimensions. The last step of the model is the visualization process that allows us to place the nodes according to their structural and compositional similarities and then identify important nodes regarding the interactions between communities. This model is based on the division of the nodes into two categories according to their connectivity characteristics: nodes connecting different communities and nodes connecting nodes only within their own communities.

Keywords: community detection, social network analysis, visualization of clustered graphs, community interaction.

RÉSUMÉ

Socio-semantic Networks Algorithm for a Point of View Based Visualization of On-line Communities

Juan David Cruz Gómez
Département LUSSI, Telecom - Bretagne
10 décembre 2012

L'explosion récente des sites de réseaux sociaux en ligne comme Facebook, LinkedIn et Pinterest, les gens sont de plus en plus confrontées à toute une nouvelle façon pour se communiquer et pour partager une quantité importante d'information qui est transmis presque en temps réel. Pour manipuler cette information de façon efficace, des chercheurs font face aux défis théoriques et ainsi techniques comme le stockage et charge des donnés dans la mémoire par exemple. Dans des réseaux sociaux tels que Facebook, il est possible d'identifier deux types d'information : d'abord un ensemble d'acteurs décrits par leur information du profil laquelle contient des intérêts, le sexe, l'âge et des autres informations de type personnelle ; puis, le deuxième type d'information fait référence aux connections entre des acteurs, que pour le cas de Facebook sont la représentation de liens d'amitié entre ces acteurs. Celles-ci sont les deux dimensions dans lesquelles un réseau social peut être décomposé : une dimension structurelle qui représente les connexions entre des acteurs et une dimension compositionelle qui décrit l'information chaque acteur de façon individuelle.

Les approches typiques utilisent chacune de ces dimensions séparément, d'un côté la structure est représentée par un graphe dont les nœuds représentent des acteurs et les arêtes représentent les liens entre ces acteurs. Toutes les analyses sont donc réduites à des problèmes de la fouille de graphes. D'un autre côté l'information du profil, représenté par des vecteurs, est utilisée pour résumer, classifier ou trouver des motifs fréquents, en réduisant le problème à un problème de fouille de donnés. Dans le problème de détection de communautés il est possible d'utiliser soit la dimension structurelle ou bien la dimension compositionelle : dans le premier cas les communautés seraient composées par des groupes de nœuds fortement connectés mais peu similaires, et pour le deuxième cas, les groupes auraient des nœuds similaires mais faiblement connectés. Donc en ne choisissant que une des dimensions la quantité possible d'information à extraire est réduite.

Cette Thèse a pour objective une nouvelle approche pour utiliser au même temps les dimensions structurelle et compositionelle lors de la détection de communautés de façon telle que les groupes aient des nœuds similaires et bien connectés. Pour la mise en œuvre de cette approche il faut d'abord une nouvelle définition de communauté que prend en compte les deux dimensions présentées auparavant et ensuite un modèle nouveau de détection qui utilise cette définition, en trouvant des groupes de nœuds similaires et bien connectés. Le modèle commence par l'introduction de la notion de *point de vue* qui permet de diviser la dimension compositionelle pour analyser le réseau depuis perspectives différentes. Ensuite le modèle, en utilisant l'information compositionelle, influence le processus de détection de communautés qui intègre les deux dimensions du réseau. La dernière étape est la visualisation du graphe de communautés qui positionne les nœuds selon leur similarité structurelle et compositionelle, ce qui permet d'identifier de nœuds importants pour les interactions entre communautés ; le modèle est basé dans la division de l'ensemble des nœuds en deux sous-ensembles selon leur connectivité : (1) des nœuds qui sont connectés avec des nœuds dans autres communautés, (2) des nœuds

avec des voisins que dans sa propre communauté.

Mots-clés : détection de communautés, analyse des réseaux sociaux, visualisation des graphes de communautés, interactions entre communautés.

Resumen

Socio-semantic Networks Algorithm for a Point of View Based Visualization of On-line Communities

Juan David Cruz Gómez
Departamento LUSSI, Telecom - Bretagne
10 décembre 2012

Con el creciente número de sitios de redes sociales como Facebook, LinkedIn y Pinterest, las personas enfrentan una nueva forma para comunicarse y para compartir una gran cantidad de información que fluye a una gran velocidad. Para manipular eficientemente esta información, los investigadores se enfrentan a diferentes desafíos tanto teóricos como técnicos como son el almacenamiento y carga de los datos en la memoria,la capacidad de procesamiento y la latencia. En redes sociales como las de Facebook es posible identificar dos tipos de información: primero, un conjunto de actores descritos por su información de perfil, la cual puede contener datos como la edad, el género e intereses personales; el segundo tipo de información se refiere a las conexiones entre los actores, que en el caso de Facebook describe los vínculos de amistad entre ellos. Estas son dos dimensiones de la misma red social: una dimensión estructural que representa las conexiones entre los actores y una dimensión composicional que describe cada actor de forma individual.

Las estrategias de análisis clásicas usan cada una de las dimensiones de información por separado, por un lado la estructura es representada por un grafo en el que los nodos representan actores y los arcos representan los vínculos entre ellos. Los análisis son reducidos entonces a problemas de minería de grafos. Por otro lado, la información de perfil es representada en forma vectorial y es utilizada para resumir, agrupar, clasificar o encontrar patrones, reduciendo de nuevo el problema, esta vez, a un problema clásico de minería de datos. En el problema de detección de comunidades se puede utilizar ya sea la dimensión estructural o ya sea la dimensión composicional, reduciendo así el espectro de la información que puede ser extraída: en el primer caso las comunidades estarán compuestas por nodos fuertemente conectados, pero no necesariamente similares, mientras que en el segundo caso, las comunidades estarán compuestas por nodos similares pero débilmente conectados.

Este trabajo busca una nueva estrategia que permita utilizar al mismo tiempo la dimensión estructural y la dimensión composicional de tal forma que se puedan encontrar comunidades con nodos similares y bien conectados entre ellos. Esta estrategia requiere una nueva definición de comunidad que incluya las dos dimensiones presentadas antes así como un nuevo modelo de detección de comunidades que pueda utilizar la nueva definición. Esta Tesis doctoral introduce una nueva definición de comunidad y presenta un nuevo modelo de detección y visualización de comunidades de nodos similares y bien conectados. Dicho modelo comienza definiendo la noción de *punto de vista* el cual permite dividir la dimensión composicional de tal forma que la red pueda ser analizada desde perspectivas diferentes. El modelo toma información composicional, definida por el punto de vista, para influenciar el proceso de detección de comunidades integrando ambas dimensiones. Finalmente, con el modelo de visualización de las comunidades, los nodos son ubicados de acuerdo con su similitud tanto estructural como composicional, haciendo posible la identificación de nodos que puedan ser importantes para la interacción entre comunidades. Este modelo está basado en la división del conjunto de nodos en dos categorías según sus características de conectividad: nodos que conectan diferentes

comunidades y nodos cuyos vecinos están dentro de su propia comunidad.

Palabras clave: detección de comunidades, análisis de redes sociales, visualización de comunidades, interacción de comunidades.

Contents

List of Figures

List of Tables

Résumé en Français

1 Introduction

À l'heure actuelle les sites de réseaux sociaux en ligne sont devenus un composant intégral des usages professionnels et personnels. Les entreprises intègrent dorénavant leurs clients dans leur chaîne de valeur, depuis les travaux R&D jusqu'au service après vente. Dans leur sphère personnelle, les internautes interagissent et partagent différents types de medias à travers le monde entier.

C'est important de différencier ici les réseaux sociaux des sites de réseautage social. Un réseau social est l'ensemble des relations sociales entre les membres d'un groupe. Il est communément défini par des connexions entre acteurs ; les acteurs peuvent être des personnes ou des organisations, les liens peuvent être de plusieurs types : amitié, affiliation, dépendance ou communication. Les réseaux sociaux existent, qu'ils soient médiatés par internet ou non, qu'ils soient explicites ou non.

Les sites de réseautage social sont, quant à eux, des sites Web conçus pour pratiquer le réseautage social sur Internet. Ces services permettent à un individu ou une organisation de constituer leur réseau social en ligne. Selon le site, les internautes déclarent des liens d'amitié, des relations professionnelles ; ils sont amenés à développer leur réseau et interagissent, souvent autour de contenus.

L'analyse de réseaux sociaux considère les réseaux sociaux comme des graphes, où les acteurs sont représentés par des nœuds et les relations par des arêtes. Le médecin psychiatre Jacob L. Moreno a introduit le sociogramme en 1933. Représenter un réseau social par un graphe objective les relations interpersonnelles d'un groupement social, ce qui permet d'en étudier la structure des relations : les liens d'influence, les individus en position de pouvoir, la dynamique du groupe, etc.

Toutefois, un réseau social contient plus d'informations que seulement le graphe de relations : il contient également des informations à propos de chaque acteur, par exemple ses préférences de lecture, son âge, son emplacement géographique, et en gé-

néral, d'autres informations relatives au contexte du réseau (en ligne ou hors ligne).

Ainsi, un réseau social selon [Wasserman and Faust, 1994] est composé de deux variables : une variable structurelle, qui décrit les connexions entre les acteurs, une variable de composition, qui décrit chaque acteur de façon individuelle selon son information propre. Une troisième variable peut-être définie décrit des groupes d'acteurs définis soit par des critères externes soit par l'une ou les deux autres variables ; cette variable est dite d'affiliation

Ces informations additionnelles non structurelles sont désormais accessibles dans des données réelles, issues des sites de réseautage, par exemple, et sont à l'heure actuelle sous-exploitées. Les réseaux sociaux enrichis de la sorte seront appelés dans la suite de ce document *réseaux sociaux augmentés*, pour les différentier des réseaux sociaux purement structurels modélisés par des sociogrammes.

Cette thèse présente un cadre général pour faciliter l'intégration des trois variables présentes dans un réseau social. Notre proposition est de combiner les variables structurelle et de composition pour générer une variable d'affiliation, et à partir de cette combinaison, exécuter différentes analyses de réseaux sociaux augmentés. Au-delà de la formalisation d'un modèle de réseau social augmenté, et de la méthode d'intégration de variables, cette proposition inclut un modèle visuel qui explicite l'influence des variables structurelle et de composition dans la définition de la variable d'affiliation.

2 Motivation et contexte de la thèse

2.1 Contexte de la thèse

Les réseaux sociaux du monde réel contiennent différents types d'information qui permettent de décrire des composantes du réseau. Une topologie de ces éléments a été proposée par [Wasserman and Faust, 1994]. Cette topologie sur les réseaux sociaux structure notre réflexion concernant l'analyse de réseaux sociaux augmentés. Ceux-ci peuvent être décrits à travers trois variables : (1) la variable structurelle qui contient l'information des liens entre les acteurs pris dans leur ensemble, (2) la variable de composition, qui décrit chaque acteur, par exemple avec un profil, et (3) la variable d'affiliation, utilisée pour décrire les nœuds selon leur affectation à différents groupes, événements ou division d'une entreprise.

Dans la Figure 1 est présenté un exemple de réseau social d'entreprise avec des employés travaillant à deux endroits : les nœuds carrés représentent des personnes de New York tandis que les nœuds ronds représentent des personnes du bureau de Boston.

Cet exemple corporatif est simple, et surtout de petite taille, mais il aide à présenter les variables introduites : la variable structurelle décrit les collaborations entre les em-

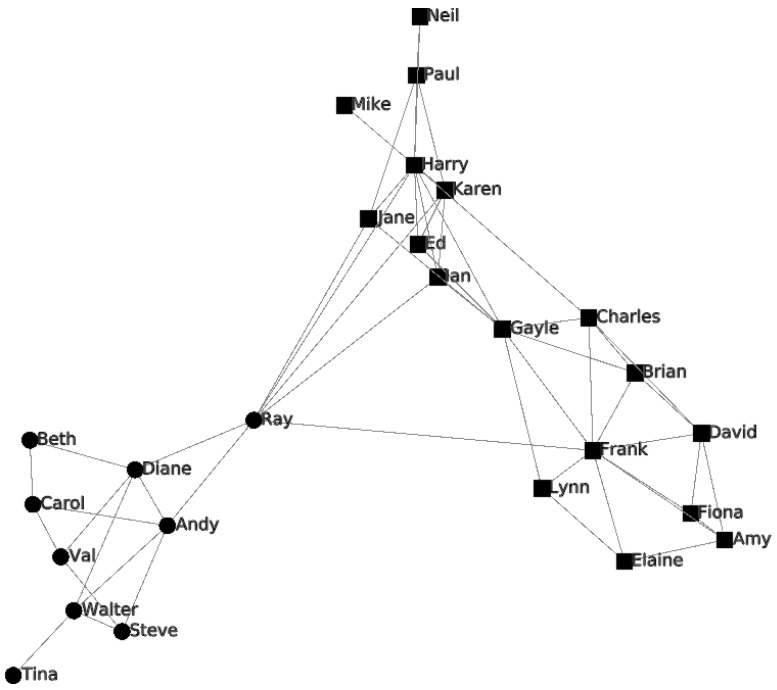


FIGURE 1 – Exemple d'un réseau social corporatif basique

ployés ; la variable de composition contient le nom, l'affectation à l'un des deux bureaux, les compétences et les activités de chaque acteur ; la variable d'affiliation est ici dérivée de l'affectation de chaque acteur à l'une des deux agences de l'entreprise.

L'analyse des réseaux sociaux cherche à extraire de nouvelles connaissances de chacune des variables présentées ci-dessus. Toutefois ce traitement en général n'utilise qu'un seul type de variable en même temps. Par exemple, en n'utilisant que la variable structurelle il est possible d'identifier des nœuds centraux ou effectuer d'autres mesures sur la structure du graphe. En revanche, avec la variable de composition, les chercheurs peuvent appliquer des techniques de fouille de données sur les données décrivant les individus, et analyser une partition, des motifs et des règles d'association par exemple.

La variable d'affiliation peut nourrir l'étude de l'appartenance (ou la co-appartenance) des acteurs à certains groupes ou associations. Cette variable a cette particularité qu'elle peut être aussi générée à partir de l'une ou l'autre des deux autres variables. L'affectation à un groupement social peut résulter d'un processus de détection de communautés qui produit une partition du graphe et classe les nœuds dans des groupes découverts par un algorithme. La variable d'affiliation n'est autre qu'une représentation d'un graphe biparti [Wasserman and Faust, 1994]. Un cas d'étude complet sur les problématiques de ce type de graphes est abordé par [Latapy et al., 2008].

Dans ces exemples d'analyse abordés, une portion de l'information est rejetée, au détriment d'analyses plus complètes, tenant compte de la richesse des données disponibles. Comme présenté dans l'état de l'art de ce travail ci-après, la littérature existante est encore limitée concernant des approches de détection de communautés dans les réseaux sociaux qui exploitent toute l'information disponible. Dans ce travail nous voulons générer la variable d'affiliation à partir de l'intégration des variables structurelle et de composition.

Une des questions à prendre en compte avant d'analyser la variable de composition, est la diversité des valeurs qu'elle peut contenir. Par exemple cette variable contient l'information du genre, de l'âge, de la taille et d'autres caractéristiques de l'acteur, comme les préférences personnelles en termes de sports, livres, art, voire des mots clés relatifs à des publications. Malgré le fait que cette variété permet de bien décrire chaque acteur, elle engendre des difficultés pour analyser les acteurs comme un ensemble. Ce phénomène est bien connu sous le nom de Malédiction de la dimension [Bellman, 1961] et sera abordé dans ce document.

Une approche pour éviter ce phénomène est de réduire la dimension. Plutôt que de la réduire de manière automatique (on sort du cadre de ce travail), nous proposons de confier cette tâche à l'analyste du réseau social. En fonction du but recherché, il sera amené à diviser le profil des acteurs en des sous-ensembles qui ont un sens pour lui : l'idée est d'imaginer ici une méthode générique d'analyse de réseau exploitant la sélection, opérée par un décideur, d'un sous-profil pertinent dans le contexte d'une analyse. On pourra ainsi par exemple analyser un réseau social en ne considérant que les loisirs ou bien en ne considérant que les compétences académiques des acteurs.

Le cadre d'analyse proposé dans cette thèse permet d'analyser un réseau social en tenant compte du profil des acteurs. L'analyse porte principalement sur la détection de communautés pour laquelle on souhaite classer, au sein d'un même groupe, les acteurs à la fois proches dans la structure du graphe et proches du point de vue de leur profil. Cette analyse automatique s'accompagne d'une analyse visuelle avec la proposition d'une visualisation adaptée.

L'idée est à terme de pouvoir approfondir l'étude d'un réseau social en montrant l'impact des profils, mais aussi des sujets discutés, sur la structuration du réseau. L'un des premiers cas d'usage proposés est de comparer les différents regroupements obtenus par la sélection de différents sous-profils. Nous parlerons ici de "point de vue" pour désigner la sélection d'un sous-profil. Ce sous-profil conditionnera l'ensemble du processus d'analyse, et peut être vu alors comme un paramètre de l'analyse. A noter que le "point de vue" prend tout son sens dans un outil interactif d'analyse visuelle où le décideur est amené à dérouler plusieurs scénarios. Ce "point de vue" peut également être "nul" si l'on ne le considère pas du tout, et dans ce cas on se ramène à l'analyse de la variable structurelle

seule. Enfin, à terme, le point de vue pourrait être généré de manière automatique, de façon à trouver ceux maximisant des critères de qualité qu'il faudrait alors définir. Cet aspect du travail est au-delà du cadre de la thèse et figure dans les perspectives identifiées.

Nous définissons ci-après la notion de *point de vue* et la manière dont nous modélisons les réseaux sociaux réels, en augmentant le graphe social par la variable de composition. Ensuite nous décrivons plus précisément les questions abordées dans ce travail de thèse.

2.2 DÉFINITION DU PROBLÈME

L'analyse d'un réseau social intégrant les variables qui le composent n'est pas un processus simple. Cette intégration de variables exige d'établir des relations entre elles, mais aussi de créer un mécanisme exploitant leur unification afin d'extraire de nouvelles connaissances, en complément des mécanismes d'analyse basés sur chacune d'elles prise séparément.

Le problème peut être divisé en deux sous problèmes ; d'abord, comment représenter les variables de telle façon qu'elles puissent être utilisées et analysées de façon holistique, et par quelle méthode ?

Ensuite, comment utiliser un modèle visuel pour représenter l'intégration des trois variables concernées ?

Pour faire cela nous introduisons ici les définitions des différentes *variables* et de *graphe augmenté* , ainsi que la notion de *point de vue*. Soit $G\left(V,E\right)$ un graphe non orienté représentant la structure du réseau social, où V est l'ensemble des nœuds et E est l'ensemble des arêtes ; ce graphe contient uniquement l'information concernant les connections des nœuds, c'est-à-dire la variable structurelle. La variable de composition de ce réseau peut être définie comme un vecteur F^{*} modélisant les caractéristiques de chaque acteur : ce vecteur peut être défini dans un espace $p-$dimensionnel où p peut varier selon la nature du profil décrivant les acteurs.

Cette représentation devient difficile à gérer quand la valeur de p croît : un nombre élevé de dimensions produit l'effet connu comme la *malédiction de la dimension* qui, à mesure que les dimensions augmentent, diminue la pertinence des mesures mathématiques employées, de plus, utiliser toutes les caractéristiques qui décrivent un acteur peut introduire du bruit pour certaines applications, et tout simplement ne pas être pertinent pour l'analyste qui cherche à étudier l'impact de certaines caractéristiques bien choisies sur la structuration du réseau social.

Definition 2.1 (Variable structurelle G)**.** *La variable structurelle représente l'ensemble des relations entre les acteurs : cette variable est le graphe social.*

Definition 2.2 (Variable compositionnelle F^*)**.** *La variable compositionnelle représente l'information F^* qui décrit chaque acteur de façon individuelle : par exemple les profils des acteurs.*

Definition 2.3 (Variable d'affiliation $\mathcal{A}$)**.** *La variable d'affiliation représente l'appartenance des acteurs à des sous-ensembles d'acteurs du réseau : par exemple des clubs ou des communautés.*

La variable d'affiliation peut être dérivée à partir soit de la variable structurelle, soit de la variable compositionnelle, soit d'une combinaison des deux autres telle que proposée par exemple dans ce travail.

Definition 2.4 (Point de vue PoV_{F^*})**.** *Étant donné un ensemble de caractéristiques de la variable compositionnelle F^*, un point de vue PoV_F est l'une des 2^f combinaisons des f éléments de F^*.*

Le point de vue est un sous-ensemble de F^*. Grâce au point de vue, le réseau social peut être décrit selon différentes perspectives, chacune d'elle étant définie par un point de vue.

Definition 2.5 (Réseau social augmenté $\mathcal{S}^+$)**.** *Étant donné un réseau social $\mathcal{S}\left(G, PoV_{F^*}\right)$ où G est un graphe représentant la variable structurelle et PoV_{F^*} représentant la variable de composition dans la forme d'un point de vue, le réseau social augmenté est défini par $\mathcal{S}^+\left(G, PoV_{F^*}, \mathcal{A}\right)$, où $\mathcal{A}$ la variable d'affiliation dérivée des autres deux variables.*

En conséquence, le réseau social augmenté peut être décrit par des perspectives différentes qui permettent de réaliser une analyse pour chaque perspective. Par exemple, les nœuds peuvent être groupés de plusieurs manières : un par point de vue, en rendant possible les contrastes selon les caractéristiques sélectionnées.

3 ÉTAT DE L'ART

L'une des hypothèses de ce travail est de considérer que la détection de communautés constitue un type d'analyse permettant de mettre en œuvre les trois variables décrites précédemment. Nous ambitionnons de classer, au sein d'un même groupe, les acteurs à la fois proches dans la structure du graphe et proches du point de vue de leur profil.

3.1 DÉTECTION DE COMMUNAUTÉS

En général, l'analyse des réseaux sociaux est dédiée à la variable structurelle. On trouve par exemple l'identification des nœuds importants ou des relations non triviales. Les

différentes métriques d'analyse sont en général calculées sur le graphe pris comme un tout, avec la notion structurante de chemin. La détection de communauté est l'une de ces analyses désormais clé lorsque l'on traite de grands (voire très grands) graphes réels. Elle permet de trouver de manière automatique des zones denses, topologiquement parlant, dans le graphe social.

Il s'agit d'un processus de clustering basé sur une distance entre les noeuds définie par les chemins du graphe. Il tient compte du nombre important d'arêtes tissant le réseau d'un sous-ensemble de noeuds en comparaison avec le réseau global. L'idée de base est de trouver des partitions dont le nombre d'arêtes à l'intérieur des groupes est supérieur au nombre d'arêtes en dehors des groupes. Par conséquent, chaque groupe ainsi identifié possède une forte connectivité à l'intérieur et une faible connectivité avec les autres groupes.

Cette idée a été résumée par [Gaetler, 2005] qui a défini trois indices pour mesurer la qualité des partitions trouvées : la couverture, qui mesure le poids de toutes les arêtes dans les groupes versus le poids des arêtes connectant les groupes ; la conductance, fondée sur l'observation que si un groupe est bien connecté, alors un nombre important d'arêtes doivent être enlevées pour le diviser en deux parties égales ; enfin, la performance indique si les deux extrémités d'une arête appartiennent au même groupe ou si deux nœuds ne formant pas une arête appartiennent à des groupes différents (le groupement est alors dit correct). Une autre mesure, utilisée de plus en plus souvent [Fortunato, 2010] est la modularité. Cette mesure, proposée par [Newman and Girvan, 2004], mesure la proportion des arêtes dans les groupes versus le nombre des arêtes en dehors des groupes, elle compare aussi cette proportion avec celle d'une partition aléatoire du même graphe.

La méthode de détection de communautés présentée par [Newman and Girvan, 2004] trouve et enlève, de façon itérative, l'arête avec le degré d'intermédiation le plus élevé. Ce processus permet de trouver des groupes connectés de façon souple avec d'autres groupes mais fortement connectés à l'intérieur de chacun d'eux. La partition sélectionnée est celle avec la modularité la plus haute ; toutefois le calcul direct de la modularité a une complexité en $O(n^2)$, où n est le nombre de nœuds du graphe, ce qui fait que le temps de calcul pour l'algorithme est important.

[Blondel et al., 2008] proposent une approche pour la détection de communautés définie comme un problème d'optimisation dont la fonction objectif est la maximisation de la modularité. La méthode commence par affecter chaque nœud à une communauté "singleton" en générant une première partition et en calculant la valeur initiale de modularité. Puis, chaque nœud est changé de sa communauté vers une autre communauté voisine, et pour tout changement la différence de modularité est calculée : si la nouvelle affectation augmente la modularité, le changement est actée. Si aucun changement n'améliore la modularité, le nœud reste dans sa communauté d'origine. Ce processus est

effectué jusqu'à ce qu'aucun changement ne soit plus possible. Le coût de cet algorithme est linéaire par rapport au nombre d'arêtes dans le cas de graphes non denses, et est particulièrement intéressante pour les grands graphes réels.

Le travail proposé par [Du et al., 2007] s'intéresse lui-aussi à la détection de communautés dans les réseaux sociaux à grande échelle. La méthode commence par énumérer toutes les cliques maximales (des graphes complets qui ne sont contenus dans aucun autre sous-graphe complet) ; après l'énumération des cliques, elle génère des noyaux qui font office de centre de gravité où les nœuds sont affectés selon leur proximité avec chacun de ces noyaux. La complexité est en $O(n^2)$, où n est le nombre de nœuds du graphe.

La plupart des méthodes trouvent des partitions disjointes, toutefois dans les applications des réseaux sociaux il est tout à fait possible de trouver des nœuds qui appartiennent simultanément à plusieurs communautés. [Pizzuti, 2009] présente un algorithme génétique dont le but est de maximiser le nombre d'arêtes dans chaque groupe. [Lipczak and Milios, 2009] présentent aussi un algorithme génétique pour identifier des communautés floues. La différence fondamentale est liée à la définition de la fonction d'adaptation, qui est composée par une combinaison de trois mesures : le normalized cut par [Shi and Malik, 2000], la largeur de silhouette [Rousseeuw, 1987] et la modularité. La complexité de ces méthodes est en $O(n^2 + \lambda)$, où λ est le nombre d'itérations des simulations.

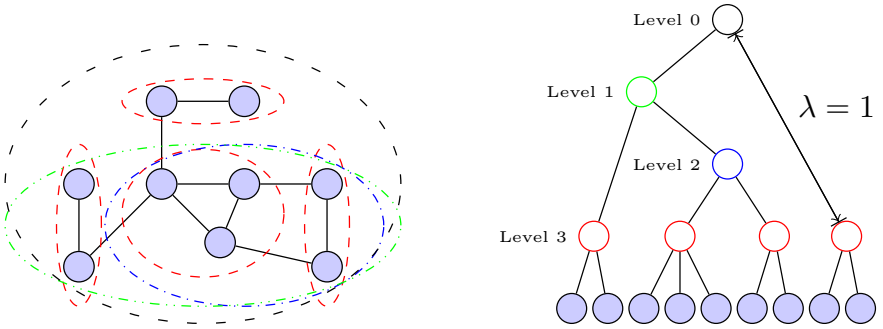
Les méthodes précédentes n'utilisent que la variable structurelle du réseau social. A notre connaissance, seuls [Zhou et al., 2009], explorent le fait que les communautés puissent être aussi identifiées à partir des variables de composition du réseau. Ils présentent en effet une méthode proche de la nôtre pour utiliser simultanément les variables structurelles et de composition pour identifier les communautés. Leur méthode utilise un surfeur aléatoire qui traverse k groupes prédéfinis et qui essaye de maximiser la distance entre les groupes en affectant les nœuds à chaque groupe selon leur similarité. Pour ce faire un graphe augmenté est créé, qui représente les variables de composition du graphe. Avec la matrice de probabilité de transition du graphe augmenté, le surfeur aléatoire trouve les k groupes sémantiquement proches. La qualité structurelle est mesurée par la densité d'arêtes dans chaque groupe.

3.2 Visualisation de graphes de communautés

Un graphe de communautés est un type de graphe hiérarchique dont la distance entre deux nœuds de l'arbre d'inclusion est au maximum égale à un.

La Figure 2(a) présente un exemple de graphe de communautés et la Figure 2(b) présente l'arbre d'inclusion de la partition. Cet arbre est utilisé pour visualiser les différents

niveaux qui représentent les groupes.



(a) Exemple d'un graphe de communautés $C(G,T)$

(b) Arbre d'inclusion T issue d'un graphe de groupes

FIGURE 2 – Exemple d'un graphe de communautés et son arbre d'inclusion associé

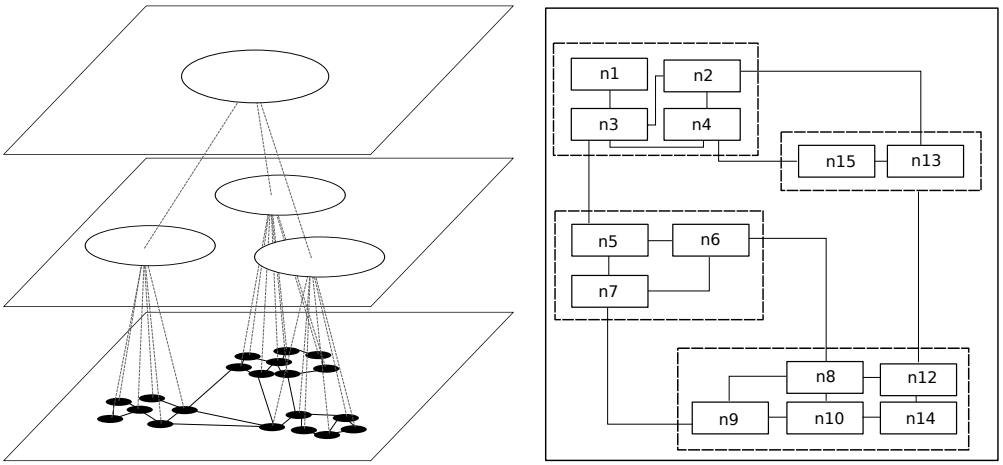
Cette définition est proposée et utilisée par [Eades and Feng, 1997] lors de la présentation de leur méthode de visualisation. Cette méthode représente chaque niveau de l'arbre d'inclusion comme une couche tridimensionelle. D'abord, pour la première couche, les nœuds feuilles de chaque communauté sont placés ; dans la deuxième couche, un cercle est ensuite dessiné au-dessus de chaque groupe. Ce processus est répété jusqu'à la racine de l'arbre d'inclusion. La figure 3(a) présente un exemple de la visualisation avec la méthode de Eades et Feng.

[Tamassia, 1987] présente un algorithme qui utilise des rectangles pour représenter les nœuds et des lignes droites avec des angles droits pour représenter les arêtes ; l'auteur utilise cette approche pour dessiner des circuits VLSI. Les groupes sont représentés par des rectangles autour des nœuds. Un exemple de cette représentation est montré dans la Figure 3(b). L'algorithme présenté par [di Giacomo et al., 2007] utilise aussi une approche orthogonale, mais appliquée à la représentation de sites web.

[Noack, 2003] présente un algorithme dirigé par des forces qui utilise d'une part un modèle linéaire pour représenter l'attraction entre deux nœuds voisins et, d'autre part, un modèle logarithmique pour modéliser la répulsion entre deux nœuds non connectés. Ce modèle n'utilise pas un graphe déjà groupé : il constitue aussi une méthode pour identifier graphiquement des communautés dans un réseau social.

[Bourqui et al., 2007] présentent une méthode de visualisation qui prend en compte les poids des arêtes. Cette méthode impose quatre contraintes :

- interdire le chevauchement des groupes pour faciliter l'interprétation du dessin,
- garder l'arbre d'inclusion pour visualiser la hiérarchie produite par l'algorithme de détection de communautés,



(a) Exemple de graphe de communautés dessiné avec l'algorithme multi-niveaux

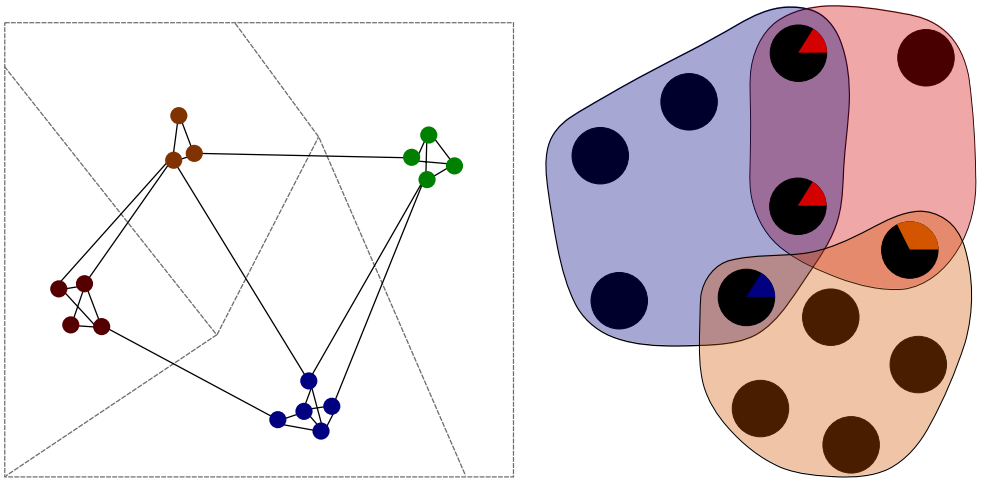
(b) Exemple d'un graphe de communautés dessiné avec la représentation orthogonale

FIGURE 3 – Exemple de résultats des algorithmes de layout – I

- utiliser un polygone convexe pour définir chaque groupe,
- respecter les poids du graphe en utilisant des fonctions de minimisation de l'énergie.

Avec la définition de graphe de communautés de [Eades and Feng, 1997] et avec la définition de graphe quotient de [Brockenauer and Cornelsen, 2001], les auteurs commencent par placer les nœuds individuels, puis les nœuds des niveaux suivants. À l'inclusion de chaque niveau, l'espace de visualisation est divisé en utilisant des diagrammes de Voronoï. La Figure 4(a) présente un exemple de ce modèle de visualisation.

En général les méthodes de visualisation de graphes de communautés ont été conçues pour mettre en évidence les différences entre chaque groupe, ce qui implique de présenter chaque groupe éloigné des autres. [Santamaría and Therón, 2008] proposent une méthode pour dessiner des communautés non disjointes. Ils utilisent une version modifiée de l'algorithme dirigé par des forces dont la force d'attraction est modélisée par un ressort et la répulsion par un modèle de gravitation. Pour éviter l'encombrement des éléments visuels, les arêtes ne sont pas dessinées, par contre les groupes sont modélisés par des polygones convexes. Ainsi, les nœuds appartenant à différents groupes sont placés à la frontière de ces polygones de telle façon que le partage soit évident. La Figure 4(b) présente un exemple de cette visualisation ; les nœuds partagés sont dessinés comme un camembert indiquant le degré d'appartenance à chaque groupe.



(a) Exemple de graphe de communautés dessiné avec l'algorithme pour des graphes pondérés

(b) Exemple d'un graphe de communautés dessiné avec la représentation des communautés chevauchantes

FIGURE 4 – Exemple de résultats des algorithmes de layout – II

Les algorithmes classiques n'utilisent que la structure du graphe pour placer les nœuds et en général il n'est pas facile de leur ajouter d'autres critères, comme la similarité entre nœuds, pour trouver la position de chacun des nœuds.

4 INTÉGRATION DES VARIABLES DANS UN RÉSEAU SOCIAL

4.1 INTRODUCTION

Nous avons indiqué précédemment que les composants d'un réseau social sont divisés en trois variables : la variable structurelle, qui représente les connexions entre les acteurs, la variable de composition qui décrit chaque acteur de façon individuelle et la variable d'affiliation qui décrit l'appartenance de chaque acteur à un ou plusieurs groupes.

La variable structurelle peut être utilisée pour décrire le réseau social globalement, en analysant seulement les connexions entre les acteurs. En revanche, la variable de composition permet d'analyser chaque acteur de façon individuelle, par exemple à partir de ses préférences, compétences professionnelles, langues et livres favoris. Cette information peut être utilisée pour créer des catégories d'acteurs, qui, en général, n'ont pas de rapport avec la structure du réseau. La troisième variable est spéciale dans le sens qu'elle peut être générée à partir de l'une des deux autres variables : si elle est générée à partir de la première variable, l'affectation reflète l'appartenance aux communautés de nœuds

fortement connectés ; si elle est générée à partir de la deuxième variable, l'affectation représente l'appartenance à des clusters de nœuds similaires.

Cette section présente la partie du modèle dédiée à l'intégration des variables composant un réseau social augmenté. L'intégration proposée est réalisée à travers la création de variables d'affiliation issues de la combinaison de la variable structurelle et des différents points de vue dérivés de la variable de composition.

Avant de continuer, nous introduisons quelques notations que seront utilisées ultérieurement. Soit $\mathcal{S}^{+}\left(G, PoV_{F^*}, \mathcal{A}\right)$ un réseau social augmenté, où $\mathcal{A}$ représente la variable d'affiliation. $\mathcal{A}$ résulte d'un processus de partitionement dont le résultat es une partition $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$ de l'ensemble V des nœuds en k sous-ensembles disjoints ; $e\left(u, v\right) \in E$ est l'arête entre les nœuds u et v, $e\left(u\right)$ est l'ensemble de toutes les arêtes depuis et vers le nœuds u. Soit $E\left(C_i, C_j\right), 1 \leq i, j \leq k$ l'ensemble des arêtes connectant les groupes i et j. $E\left(C_i, C_i\right) = E\left(C_i\right), 1 \leq i \leq k$ est l'ensemble des arêtes connectant les nœuds présents seulement dans le groupe i.

$\mathcal{A}_G$ est une variable d'affiliation dérivée de la variable structurelle. La partition décrite par cette variable est définie par des groupes de nœuds fortement connectés. D'un autre côté $\mathcal{A}_{PoV_{F^*}}$ est une variable dérivée de la variable de composition conditionnée par un point de vue ; les groupes dans cette partition sont proches en termes de similarité de profil dans PoV_{F^*}.

Notons que les partitions générées à partir des différentes variables ont été créées avec des critères différents, donc les mesures de qualité utilisées pour chacune sont différentes. Ce fait sera abordé ultérieurement lors de l'intégration des deux variables.

4.2 Intégration des variables dans des réseaux sociaux augmentés

L'intégration des variables structurelles et de composition permet de guider le processus d'identification des communautés en ajoutant la similarité de chaque nœud aux critères de la structure utilisés pendant l'identification des communautés. Pour ce faire, nous divisons le processus de détection de communautés en deux phases : d'abord, un clustering de nœuds selon leur similarité de profil, puis, un algorithme de clustering structurel exploitant la partition de composition, c'est-à-dire influencé d'une façon telle que les groupes contiennent les nœuds similaires et connectés.

4.2.1 Première phase : clustering de composition

Étant donné un point de vue dérivé d'un ensemble PoV_{F^*}, chaque nœud peut être caractérisé par son vecteur d'attributs ou une instance u du point de vue. Il est possible

d'utiliser ces vecteurs en entrée d'un algorithme de classification non supervisée comme les cartes auto-organisatrices ([Kohonen, 1997]). Cela permet de créer des groupes de nœuds suivant la similarité de leurs attributs, c'est-à-dire les instances de u sont les données en entrée de l'algorithme. L'avantage de cet algorithme est que, à la différence des approches comme les $k-$means, l'utilisateur n'a pas besoin de fixer a priori le nombre final de groupes.

L'algorithme de cartes de Kohonen utilisé a un réseau $\mathcal{N}$ basé sur une grille rectangulaire de taille de $f \times f$ neurones, avec $f = |PoV_{F^*}|$, le nombre d'attributs utilisés dans le point de vue. Les valeurs initiales des poids sont tirées aléatoirement. Les poids des neurones sont ajustés selon leur proximité au neurone gagnant. Un taux d'apprentissage η est utilisé pour éviter les maxima locaux et des convergences prématurées. Après chaque itération, le taux d'apprentissage est réduit par un facteur $\varepsilon, 0 < \varepsilon < 1$. Le voisinage est calculé avec une taille t et le neurone gagnant est de centre c.

La sortie est alors une partition $\mathbf{C}_{SOM}$ formée par des groupes de nœuds similaires en termes du point de vue choisi. Pour mesurer la qualité de la partition nous utilisons la distance moyenne entre les points de chaque groupe, laquelle a été mise à l'échelle pour avoir des valeurs entre 0 et 1.

4.2.2 Deuxième phase : influence de la composition sur le clustering structurel

Une fois que la partition compositionnelle $\mathbf{C}_{SOM}$ a été calculée, on peut alors entrer dans la seconde phase de la méthode. Dans cette étape, on utilise un algorithme classique de détection de communautés, le fast unfolding algorithm – FU, proposé par [Blondel et al., 2008]. Cet algorithme utilise un processus de Monte-Carlo pour optimiser la modularité Q, présentée par [Newman and Girvan, 2004].

Avant l'exécution du fast unfolding algorithm, on inclut les informations obtenues lors de la première phase. Cela est effectué par le changement des poids des arêtes en fonction de la partition obtenue $\mathbf{C}_{SOM}$. Pour chaque paire de sommets $v_i, v_j \in V$, $\forall v_i \neq v_j$, le poids de l'arête $e\,(v_i, v_j)$ est modifié par la distance euclidienne des instances du point de vue correspondant à chaque nœud :

$$w_{ij} = 1 + \alpha(1 - d(\mathcal{N}_{ij}))\delta_{ij} \tag{1}$$

avec $\alpha \geq 1$ une constante, $d\,(\mathcal{N}_{ij})$ la distance entre les neurones i et j, et $\delta_{ij} = 1$ si v_i et v_j appartiennent au même cluster dans $\mathbf{C}_{SOM}$ et 0 sinon.

Une fois que les poids sont modifiés selon l'équation 1, une partition, $\mathbf{C}_{SOM-FU}$ est calculée en utilisant le FU. Cette nouvelle partition contient l'ensemble des communautés finales et l'information structurelle. En modifiant les poids du graphe avec l'équation 1,

le graphe devient pondéré et les arêtes avec un poids plus grand ont une probabilité plus élevée d'être affectées à la même communauté.

Une nouvelle partition est trouvée en utilisant l'algorithme de Fast Unfolding. La variable d'affiliation $\mathcal{A}_{\mathcal{S}^+}$ résultante reflète l'ensemble des communautés avec des groupes de nœuds similaires et connectés, qui intègrent l'information des variables structurelle et de composition.

4.3 Expériences et résultats de la détection de communautés

Les expériences concernant la détection de communautés ont été effectuées sur deux réseaux sociaux et avec trois points de vue, deux pour le premier réseau et un pour le deuxième. En plus des trois points de vue, pour chaque graphe le point de vue nul est utilisé. Rappelons que le point de vue nul est celui qui ignore la variable de composition.

Le premier réseau social est un réseau social issu de Facebook, avec 334 nœuds et 5394 arêtes. Le premier point de vue utilisé est composé par les compétences professionnelles et académiques de chaque acteur ; le deuxième point de vue contient les préférences sportives de chaque acteur.

Les résultats des expériences avec cette configuration montrent qu'après l'intégration de la variable de composition, nous obtenons bien des groupes plus homogènes selon la variable de composition utilisée mesurée avec l'entropie. Par contre, la connectivité, qui était forte dans la partition générée uniquement par la variable structurelle, baisse, en faisant diminuer la modularité. Cela est un effet attendu en raison de la nature des mesures de qualité utilisées pendant la création de chaque partition. Malgré la réduction de la modularité, celle-ci reste satisfaisante, les groupes restent avec des nœuds suffisamment connectés, montrant un compromis entre les deux mesures de qualité.

Le deuxième réseau est un graphe de co-citations issu de DBLP et aimablement fourni par [Zhou et al., 2009]. Ce jeu de données est particulièrement intéressant pour nous, car les travaux de [Zhou et al., 2009], proches des nôtres, ont produit des résultats auxquels nous pouvons nous comparer. Ce jeu de données représente les connections entre différents auteurs de quatre thématiques de l'informatique. Ce réseau est composé de 10000 nœuds et 65734 arêtes, et les profils des auteurs sont décrits par ces quatre thématiques. Nous utilisons le point de vue qui permet de décrire les auteurs par ces mêmes quatre thématiques.

[Zhou et al., 2009] mesurent les communautés trouvées par rapport à l'entropie et à la densité des arêtes dans chaque groupe. Notre partition, générée avec notre méthode, montre de meilleures valeurs d'entropie et de densité. Le tableau 1 résume les résultats obtenus.

PoV	Groups	Density ρ	Entropy $\mathcal{H}$
NULL	843	$0,8324$	Respect to PoV_{SCCO} : $0,2744$
SCCO	862	$0,8245$	$0,1461$
Zhou et al.,	800	$\approx 0,51$	$\approx 0,36$

TABLE 1 – Résumé des résultats de notre processus de détection de communautés pour le graphe DBLP

4.4 CONCLUSION

L'algorithme présenté dans ce chapitre permet d'intégrer la variable structurelle et la variable de composition pour générer une variable d'affiliation, qui représente une partition avec des groupes de nœuds connectés et similaires en termes de l'information du point de vue.

Le clustering est fait en deux étapes : dans la première un point de vue est choisi, et avec ce point de vue une variable d'affiliation préliminaire est créée. Cette variable est utilisée dans la deuxième étape pour changer la structure du graphe et ainsi influencer l'algorithme de détection de communautés.

Les résultats montrent d'abord que les partitions changent en fonction du point de vue choisi et ensuite que les groupes contiennent des nœuds similaires et fortement connectés. Toutefois, il existe un compromis entre les deux mesures de qualité utilisées : lorsque la modularité est incrémentée, l'entropie est automatiquement décrémentée, l'effet contraire est aussi vrai.

5 VISUALISATION DE COMMUNAUTÉS

5.1 INTRODUCTION

Étant donnée une variable d'affiliation $\mathcal{A}$ dérivée, soit de la variable structurelle, soit de la variable de composition, soit de notre processus d'intégration de ces deux variables, l'étape suivante est la création d'un modèle pour visualiser les communautés de la partition.

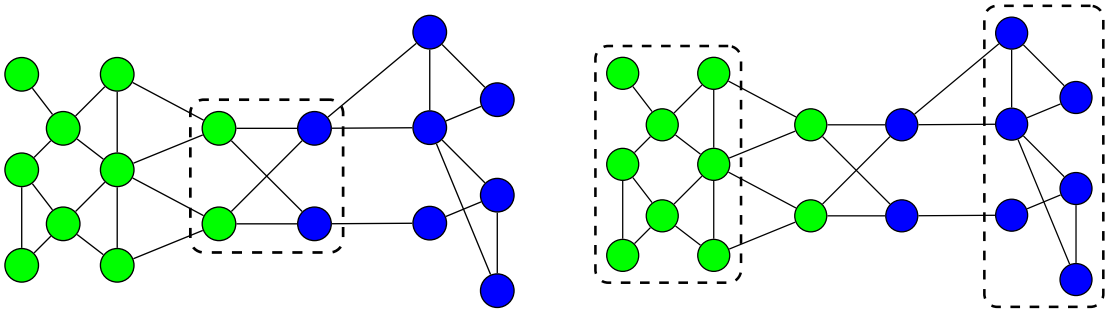
Notre modèle de visualisation doit donc être capable, en plus d'exhiber classiquement les groupes, de montrer l'existence des autres variables et leur relation avec les communautés. Notre modèle visuel utilise ainsi la variable d'affiliation pour dessiner la partition, mais se limite pour l'instant à mettre en évidence la variable structurelle du réseau social ; la variable compositionnelle n'est pas encore exhibée dans la version actuelle du modèle. L'innovation réside dans la visualisation des interactions entre communautés.

À l'heure actuelle la plupart des algorithmes de dessin de graphes de communautés sont orientés de manière à montrer la séparation de chaque groupe ; le modèle présenté dans ce chapitre est conçu pour mettre en évidence les interactions entre les communautés et pour révéler des rôles importants de nœuds impliqués dans ces interactions. Ces rôles sont définis selon la partition et la structure locale de chaque nœud et le modèle visuel est utilisé pour aider à leur identification.

La première étape consiste à trouver les nœuds en charge de la connexion des groupes, c'est-à-dire, les nœuds qui sont connectés avec des nœuds d'autres groupes ; ces nœuds peuvent être vus comme des ponts entre les communautés. Pour ce faire l'ensemble des nœuds est divisé en deux catégories : les nœuds dont les liens commencent et finissent dans leur propre communauté, dits nœuds intérieurs, et les nœuds avec au moins un lien qui commence ou finit dans une autre communauté, appelés nœuds de frontière.

Les nœuds seront placés selon une mesure de similarité avec laquelle deux nœuds avec des voisinages similaires seront proches l'un de l'autre. Cette mesure de similarité utilise la proportion de voisins en commun des deux nœuds en disant que ces nœuds sont similaires si ses voisins sont similaires.

En utilisant une partition de k communautés, la division des nœuds produit $k+1$ sous-ensembles : un avec les nœuds de frontière provenant de toutes les communautés et k avec les nœuds intérieurs provenant de chaque communauté. L'algorithme calcule une matrice de similarité pour chacun des $k+1$ sous-ensembles et à partir de chacune de ces matrices la position de chaque nœud est établie. Les Figure 5(a) et 5(b) présentent des exemples de chaque type de nœud dans un réseau avec deux groupes.



(a) Exemple de nœuds de frontière (b) Exemple de nœuds intérieurs

FIGURE 5 – Exemple de la localisation de chacun des types des nœuds dans un réseau social. Chacune des couleurs représente un groupe de la partition

Nous rappelons ici les notations utilisées pendant la description de l'algorithme. Étant donné un graphe $G(V, E)$ où V est l'ensemble des nœuds et E est l'ensemble des arêtes, et une partition $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$, soit $e(u, v) \in E$ l'arête entre les nœuds u et v,

$e(u)$ est l'ensemble de toutes les arêtes liées au nœud u. Soit $E(C_i, C_j), 1 \leq i, j \leq k$ l'ensemble d'arêtes connectant les groupes i et j et soit $E(C_i, C_i) = E(C_i), 1 \leq i \leq k$ l'ensemble d'arêtes à l'intérieur du groupe i.

Definition 5.1 (Nœud de frontière v^+)**.** *Étant donné un groupe $C_i \in \mathbf{C}$, un nœud v est dit de frontière de C_i si et seulement si $\exists \varepsilon \in e(v) : \varepsilon \notin E(C_i)$.*

Ainsi les nœuds de frontière ont des liens vers/depuis d'autres communautés. La Figure 5(a) présente un exemple de nœuds de frontière.

Definition 5.2 (Nœud intérieur v^-)**.** *Étant donné un groupe $C_i \in \mathbf{C}$, un nœud v est dit intérieur à C_i si et seulement si $\forall \varepsilon \in e(v),\ \varepsilon \in E(C_i)$.*

Cela signifie que le nœud intérieur a uniquement des liens avec d'autres nœuds de sa propre communauté. La Figure 5(b) présente un exemple de nœuds intérieurs.

5.2 PRÉSENTATION ET DESCRIPTION DE L'ALGORITHME

L'objectif de l'algorithme est de placer les nœuds de façon telle que leur proximité indique la similarité existant entre eux. L'algorithme de tracé proposé est divisé en deux étapes : d'abord, placer les nœuds de frontière, puis, dans un deuxième temps, placer les nœuds intérieurs. Ainsi l'analyste pourra axer son exploration du graphe sur les nœuds intervenant dans les relations inter-communautés.

5.2.1 MULTI-DIMENSIONAL SCALING

Le multi-dimensional scaling (MDS), est une technique pour représenter visuellement des objets en fonction de leur similarité ou de leur dissemblance. La distance dans un sous espace $\rho-$dimensionnel, généralement euclidien, respecte la similarité de l'espace d'origine. Cet algorithme permet alors de placer les nœuds selon leur similarité.

On souhaite représenter le graphe de clusters dans un espace bidimensionnel de sorte à voir les interactions entre les différents groupes. Il existe plusieurs implémentations de l'algorithme MDS qui peuvent être classifiées en trois types (Ingram et al., 2009) : des méthodes classiques cherchant une solution analytique en minimisant une fonction d'effort. Ces méthodes ont une complexité en $O(n^3)$; des méthodes basées sur la distance et l'optimisation non linéaire comme par exemple la méthode de descente de gradient, avec une complexité en $O(L \cdot n^2)$ où L est la dimension cible, et enfin, des méthodes basées sur la simulation de systèmes masses-ressorts qui ont une complexité en $O(n^3)$: un processus de complexité $O(n^2)$ exécuté n fois.

Dans ce travail, nous utilisons l'algorithme SMACOF (Scaling by MAjorizing a COmplicated Function), qui est du deuxième type, et qui converge de façon monotone en un point stable par réduction d'une fonction de stress [Ingram et al., 2009]. La complexité de l'algorithme est en $O\left(L \cdot n^2\right)$, où n est le nombre d'éléments de l'ensemble.

Pour mesurer l'approximation des distances par rapport aux dissemblances, nous utilisons une fonction de stress. Ainsi, étant donnée une matrice $\mathbf{X}$ de points dans un espace $\rho-$dimensionnel, la fonction de stress $\sigma\left(X\right)$ est définie comme suit :

$$\sigma\left(\mathbf{X}\right) = \sum_{i<j} \left(d_{ij} - \delta_{ij}\right)^2 \tag{2}$$

où d_{ij} est la distance et δ_{ij} est la dissemblance entre les objets i et j.

- **Mesures de dissemblance** Dans le cas étudié ici, nous cherchons à représenter les nœuds d'un graphe. Pour utiliser le MDS, nous devons définir la dissemblance δ_{ij} entre les nœuds. Une mesure très utilisée [Wasserman and Faust, 1994] est la distance géodésique entre les nœuds : $d\left(u, v\right) = \min_p \mathbf{A}^{[p]}_{uv}$, où $\mathbf{A}$ est la matrice d'adjacence et $\mathbf{A}^{[p]}$ est la matrice puissance p. Cela représente le chemin le plus court de longitude p entre les nœuds u et v. Un autre type de mesure englobe les métriques basées sur la disparité des voisinages de deux nœuds donnés. Deux mesures de ce type sont la distance de Jaccard et la distance cosinus [Fortunato, 2010]. Soit $N\left(u\right)$ l'ensemble des voisins du nœud u. Alors, étant donnés deux nœuds $u, v \in V$, la distance de Jaccard d_J est donnée par :

  $$\delta_J\left(u, v\right) = 1 - \frac{\left|N\left(u\right) \cap N\left(v\right)\right|}{\left|N\left(u\right) \cup N\left(v\right)\right|} \tag{3}$$

  Des voisinages similaires indiquent que les nœuds sont aussi similaires.

- **Initialisation des points** Puisque le MDS utilise un algorithme de descente de gradient, les valeurs initiales des points peuvent changer le résultat final. Cela veut dire que le résultat est un minimum local dépendant du point de départ. Nous définissons une matrice $\mathbf{X}^{[0]}$ avec les coordonnées initiales des nœuds. Ces coordonnées peuvent être définies de façon aléatoire, mais le dessin obtenu sera alors différent d'une exécution à l'autre. Or nous voulons un dessin identique, pour le même ensemble de données, à chaque exécution, de manière à préserver une stabilité visuelle pour l'utilisateur. Nous utilisons pour ce faire une procédure qui place chaque point sur une circonférence de rayon $r = 1$ et de centre $c = (0, 0)$. Ensuite, la position du nœud i est :

  $$\begin{aligned} \mathbf{X}^{[0]}_x\left(i\right) &= \cos\theta_i \\ \mathbf{X}^{[0]}_y\left(i\right) &= \sin\theta_i \end{aligned} \tag{4}$$

où $\theta_i = (i-1)\frac{2\pi}{k}$, $i = 1, 2, \ldots, k$. Cette initialisation suit la recommandation de [Borg and Groenen, 1997] et permet d'avoir toujours la même position initiale pour les nœuds.

5.2.2 Dessin des nœuds de frontière V^+

Les nœuds de frontière sont placés dans un cercle de rayon $0,5$. Ce cercle est centré sur l'origine absolue $c = (0, 0)$, qui est utilisée par l'algorithme SMACOF comme centre de référence. L'ensemble V^+ est pris en entier et contient les nœuds de tous les groupes. L'idée est de montrer les relations entre les communautés grâce à la position de chaque nœud frontière. Comme la dissemblance entre nœuds est calculée selon leur voisinage, les positions proches permettent de voir leur proximité structurelle dans le graphe G. Ainsi des nœuds qui sont à l'interface entre les mêmes nœuds auront des rôles similaires de médiation par exemple.

L'algorithme retourne $\mathbf{X}_{V^+}$, l'ensemble des coordonnées pour chaque nœud de frontière. Une fois que les positions sont déterminées, elles sont modifiées de façon à ce que la distance de chaque point à l'origine soit inférieure à $0,5$. Cette normalisation permet de bien séparer les zones destinées à chaque type de nœud.

5.2.3 Dessin des nœuds intérieurs V_i^-

V_i^- est le sous-ensemble des nœuds intérieurs à la communauté i. Chacun de ces sous-ensembles doit être placé près des nœuds de frontière déjà placés, qui appartiennent à la même communauté. Pour ce faire, nous définissons le centre $\mathcal{P}_i$ de la communauté comme :

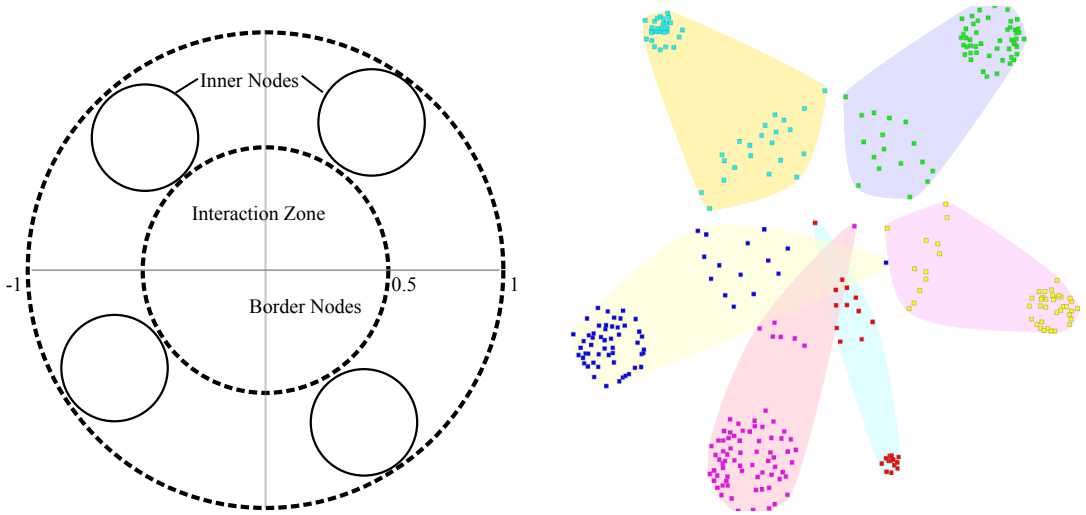
$$\begin{aligned} \mathcal{P}_i^x &= \pm\sqrt{\frac{1}{m_i^2+1}} \times r \\ \mathcal{P}_i^y &= \mathcal{P}_i^x \times m_i \end{aligned} \tag{5}$$

où $r = 0,75$ est le rayon du cercle au milieu de l'anneau défini par l'espace des nœuds de frontière et la limite de la surface de dessin, soit le cercle $x^2 + y^2 = 1$, et m_i est la pente du vecteur formé par le centre de masse des nœuds de frontière de la communauté i et l'origine. Donc, $\bar{x_i}$ et $\bar{y_i}$ sont :

$$(\bar{x_i}, \bar{y_i}) = \sum_{u \in C_i \cap V^+} \frac{u_x, u_y}{|C_i \cap V^+|} \tag{6}$$

où u_x, u_y sont les coordonnées x et y du nœud u. Ainsi, chaque point $\mathcal{P}_i = (\mathcal{P}_i^x, \mathcal{P}_i^y)$ est le centre de sa communauté et sera utilisé pour placer les nœuds de chaque communauté.

De cette manière les nœuds intérieurs de chaque communauté sont placés selon leur ressemblance de voisinage et face aux nœuds de frontière de leur communauté.



(a) Localisation des éléments du modèle de dessin

(b) Localisation des nœuds intérieurs selon la position des nœuds de frontière

FIGURE 6 – Localisation des éléments et exemple du modèle de visualisation

La Figure 6(a) présente la disposition visuelle des différents éléments du modèle et la Figure 6(b) montre un exemple de la localisation finale des nœuds avec l'algorithme décrit.

5.3 RÔLES DANS LES RÉSEAUX DE COMMUNAUTÉS

Un aspect important de l'analyse des réseaux sociaux est la mesure de l'importance de certains acteurs du réseau. Cette importance peut être la capacité de certains nœuds pour déconnecter deux ou plusieurs groupes dans le réseau, ou pour concentrer le flux de messages entre différentes équipes de travail dans une entreprise. Cet aspect a été étudié dans des espaces académiques autant qu'industriels comme présenté par [Aldrich and Herker, 1977], par [Guimera and Amaral, 2005] et par [Cross and Parker, 2004]. Le Tableau 2 présente un résumé comparatif des rôles trouvés dans les réseaux de communautés.

La dernière colonne spécifie le type de nœud qui peut être trouvé dans le rôle (I pour des nœuds intérieurs, F pour des nœuds de frontière). Dans la suite de ce travail nous utilisons les définitions des rôles de Guimerà et Amaral parce que ces définitions sont plus variées pour décrire l'ensemble des nœuds.

Aldrich and Herker	Cross and Parker	Guimerà and Amaral	Type
N/D	Personnes périphériques	Nœuds ultra périphériques	I
		Nœuds périphériques	F
	N/D	Nœuds connecteurs non centraux	F
	N/D	Nœuds sans proches non centraux	F
Nœuds d'ouverture types I et II	Nœuds centraux	Nœuds centraux provinciaux	I, F
	Nœuds d'ouverture	Nœuds connecteurs centraux	F
	Nœuds d'intermédiation	Nœuds sans proches centraux	F

TABLE 2 – Résumé comparatif des définitions des rôles dans les réseaux de communautés et leur utilisation potentielle pour nos deux types de nœuds (I pour les nœuds intérieurs, F pour les nœuds de frontière)

5.4 EXPÉRIENCES DE L'ALGORITHME DE LAYOUT

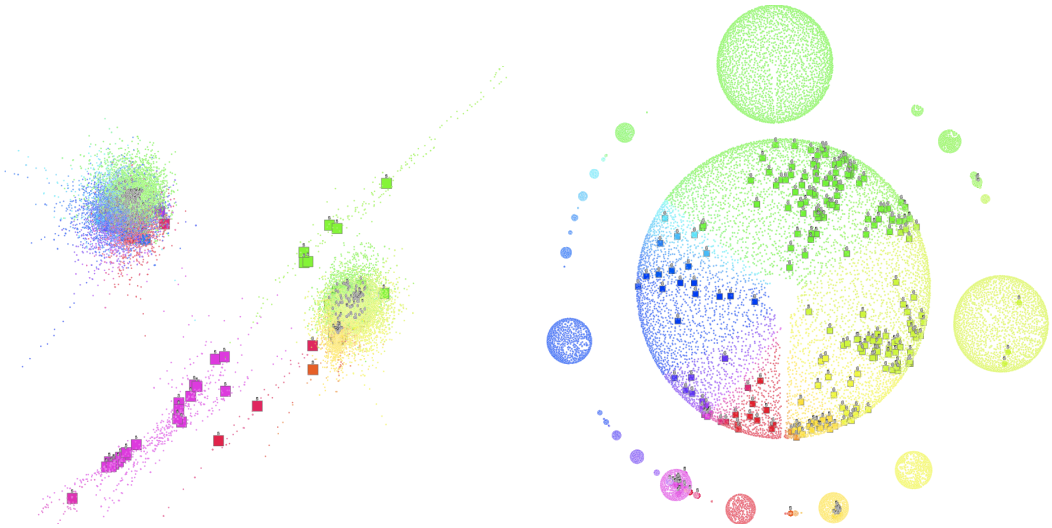
L'objectif des expériences est de tester l'algorithme en termes de passage à l'échelle et de sa capacité à séparer des nœuds impliqués dans les interactions de ceux qui n'y participent pas, et d'identifier les rôles de chacun d'eux.

Graphe	Nœuds	Arêtes	δ_G	Nb. Comm	Q	Temps(s)
Réseau d'interaction de protéines	19928	82406	$4,150 \times 10^{-4}$	56	0,6493	1021
Réseau DBLP	10000	65734	$1,315 \times 10^{-3}$	843	0,8324	346
Réseau Twitter	5389	46440	$3,199 \times 10^{-3}$	12	0,5728	89
Réseau Facebook	334	5394	$9,7 \times 10^{-2}$	6	0,7728	36

TABLE 3 – Description des jeux de donnés utilisés pour tester l'algorithme de layout

Le Tableau 3 présente les graphes utilisés pendant les expériences de l'algorithme de layout. La densité fait référence à la proportion d'arêtes par rapport au nombre de nœuds ; ces densités indiquent que les graphes utilisés peuvent avoir une structure de communauté. La dernière colonne du tableau est le temps d'exécution de l'algorithme pour placer les nœuds de chaque réseau.

La Figure 7(a) présente le résultat du layout du réseau d'interactions en utilisant



(a) Layout du réseau d'interactions avec Fruchterman & Reingold

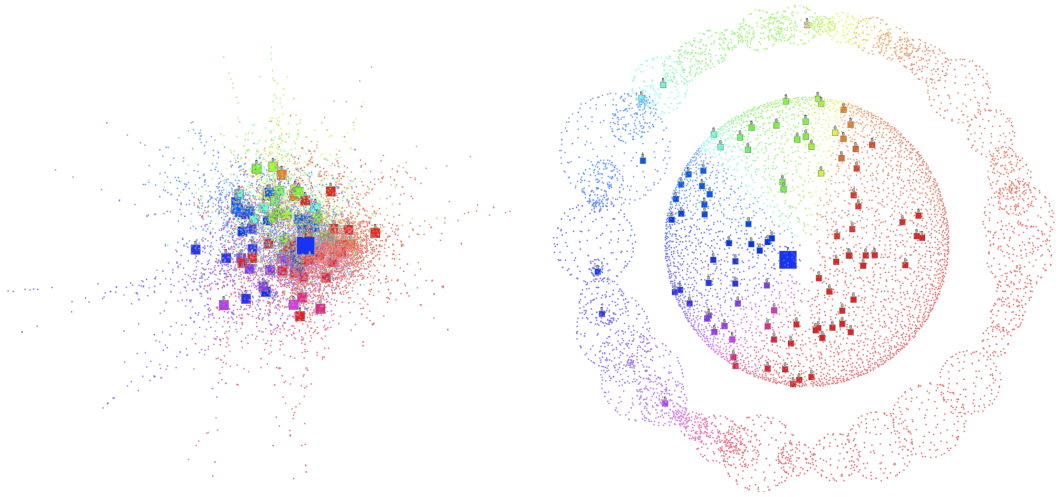
(b) Layout du réseau d'interactions avec notre algorithme

FIGURE 7 – Comparaison des résultats du layout pour le réseau d'interaction de protéines

l'algorithme Fruchterman & Reingold. Dans ce cas les nœuds sont placées avec une approche basée sur les forces. Cela fait que les communautés sont mélangées et il est difficile d'identifier tous les nœuds avec un rôle spécifique (en général la conformation des communautés). La Figure 7(b) présente le layout du même réseau avec notre algorithme. Ici nous notons les différentes communautés existantes et en plus les nœuds impliqués dans les interactions.

La Figure 8(a) montre le layout résultant de l'algorithme Fruchterman & Reingold pour le réseau DBLP. Pour ce réseau les nœuds sont placés vers le centre du dessin. En conséquence l'identification des nœuds impliqués dans les interactions est difficile. La Figure 8(b) montre le même réseau dessiné avec notre algorithme. Ce dessin présente les nœuds impliqués dans les interactions entre communautés et aussi, permet l'identification de certains rôles dans le réseau, par exemple, le grand nœud au centre : ce nœud concentre un nombre important de connections dans sa communauté et en même temps est fortement connecté avec d'autres communautés dans le réseau.

La Figure 9(a) présente le résultat du layout pour le réseau Twitter avec l'algorithme de Fruchterman & Reingold. Ce dessin montre les nœuds distribués partout dans l'espace ; le large nœud a un nombre élevé de connexions avec sa propre communauté, toutefois la localisation ne donne pas beaucoup d'information en ce qui concerne ses liens. Ce même



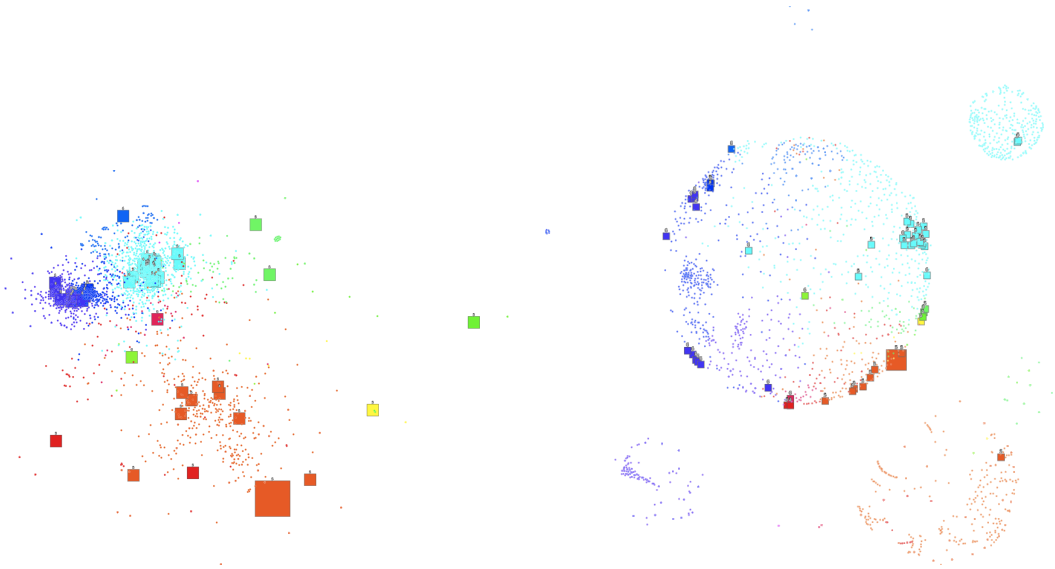
(a) Layout du réseau DBLP avec l'algorithme Fruchterman & Reingold

(b) Layout du réseau DBLP avec notre algorithme

FIGURE 8 – Comparaison des résultats du layout pour le réseau DBLP

nœud dans la Figure 9(b), est placé dans la zone d'interaction. En plus, ce nœud est loin des autres nœuds dans sa communauté, cela veut dire que ce nœud peut couper l'accès à la zone d'interaction aux nœuds qui lui sont connectés.

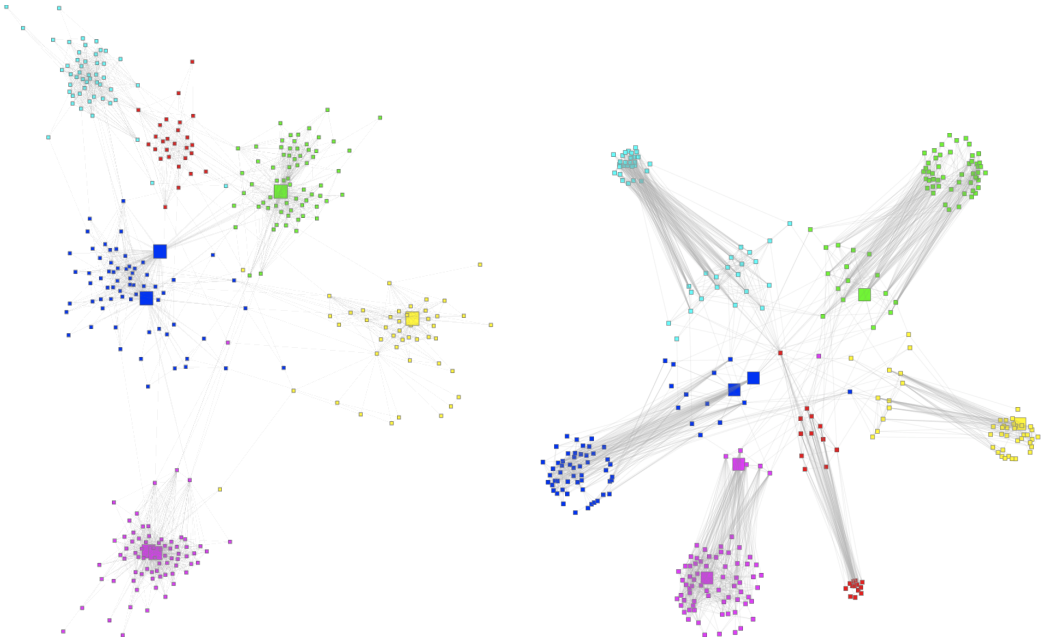
La Figure 10(a) présente le layout pour le réseau Facebook avec l'algorithme Fruchterman & Reingold. Bien que les communautés puissent être identifiées plus facilement, les nœuds impliqués dans les interactions entre communautés restent difficiles à identifier. La Figure 10(b) présente le réseau dessiné avec notre algorithme. Ici les nœuds impliqués dans les interactions sont placés dans la zone d'interaction en permettant d'identifier avec la position de chacun d'eux son importance pour la connectivité des communautés.



(a) Layout du réseau Twitter avec Fruchterman & Reingold

(b) Layout du réseau Twitter avec notre algorithme

FIGURE 9 – Comparaison des résultats du layout pour le réseau Twitter



(a) Layout du réseau Facebook avec Fruchterman & Reingold

(b) Layout du réseau Facebook avec notre algorithme

FIGURE 10 – Comparaison des résultats du layout pour le réseau Facebook

6 Conclusion et perspectives

Cette thèse est une tentative pour construire une vision intégrée des algorithmes de détection de communautés en utilisant l'information provenant de la variable structurelle et de la variable de composition présentes dans un réseau social réel.

Les contributions faites dans ce travail sont placées selon deux axes principaux : d'abord un modèle de détection de communautés qui permet d'intégrer les variables structurelle et de composition pour trouver des communautés de nœuds proches et fortement connectés. Ensuite, nous avons présenté un algorithme pour dessiner les nœuds dans un réseau de communautés. Cet algorithme est centré sur l'identification des interactions entre communautés et également l'identification des nœuds exerçant un rôle important dans ces interactions.

6.1 Algorithme de détection de communautés

L'algorithme de détection de communautés proposé dans ce travail, intègre les variables structurelle et de composition dans une variable d'affiliation qui décrit une partition des nœuds du réseau. Cette partition contient des groupes de nœuds similaires et connectés : les groupes sont formés en trouvant un compromis entre deux mesures de qualité, une pour la variable structurelle et une autre pour la variable de composition.

La variable de composition peut être composée de plusieurs caractéristiques ce qui rend parfois son analyse difficile à faire. Pour réduire cet impact nous avons introduit une notion de point de vue. Cette nouvelle notion permet de diviser la variable de composition pour créer différentes perspectives depuis lesquelles le réseau peut être analysé.

Une fois le point de vue choisi, les nœuds sont groupés selon leur similarité. Cette partition est composée par des groupes de nœuds avec une information de composition similaire. Enfin à partir de cette partition, la structure du réseau est adaptée de façon telle que les poids des arêtes reflètent la similarité des nœuds selon le point de vue sélectionné ; ce nouveau réseau est utilisé par l'algorithme de détection de communautés qui intègre alors les deux types d'information dans une variable d'affiliation adaptée aux réseaux sociaux augmentés.

6.2 Algorithme de layout de réseaux de communautés

Notre algorithme de dessin divise l'ensemble des nœuds dans un réseau de communautés en deux groupes : les nœuds qui ont des liens dans leur propre communauté et vers d'autres communautés et les nœuds avec des connexions seulement dans leur propre communauté. Les nœuds du premier type sont appellés nœuds de frontière : ils

connectent leurs communautés vers/depuis l'extérieur ; les autres nœuds sont appelés nœuds intérieurs.

Cette division permet, d'abord d'identifier les nœuds impliqués dans les interactions entre communautés et de les placer d'une façon plus représentative en facilitant l'identification des rôles et des interactions. De plus, la division permet de réduire la complexité de l'algorithme utilisé ; on ne place qu'un sous-ensembles de points au lieu de tous les nœuds de l'ensemble ce qui réduit la complexité en occupation mémoire et en charge du processeur.

Notre algorithme est générique sur le réseau de communautés à dessiner. Aussi il exploite toute variable d'affiliation et en particulière la nôtre.

6.3 PERSPECTIVES ET TRAVAUX FUTURS

Pendant la réalisation de cette thèse et après l'exécution des expériences, nous nous sommes posé quelques questions autour de nos algorithmes. Pour la partie de détection de communautés, notre méthode cherche un compromis entre deux mesures de qualité, chacune pour un type différent de partition : ce compromis peut être vu comme le rapprochement des deux partitions. L'idée plus générale que nous souhaitons explorer est de mesurer la distance entre les deux partitions et de trouver une nouvelle configuration des partitions de telle façon que la distance entre elles soit minimale. Cette approche permettra d'avoir une mesure externe de la qualité de la partition finale : nous envisageons alors de formaliser une nouvelle définition des communautés dans les réseaux sociaux réels.

D'un autre côté, le modèle visuel est utilisable avec n'importe quel réseau de communautés. Or, comme l'utilisation des différents points de vue produit des partitions différentes, nous voudrions visualiser l'impact d'un point de vue sur un layout existant du même réseau. La visualisation de cet impact permettrait d'expliciter la variable de composition incluse lors de la création de la partition avec notre algorithme de détection de communautés.

Nous visons ici un modèle visuel permettant d'explorer de façon approfondie un réseau social réel, selon différents points de vue, de manière interactive, tout en exploitant les informations riches augmentant le graphe relationnel.

Chapter 1
Introduction

Social networks have become an integral communicating and sharing environment for companies and people: in the first case, by enabling companies to integrate customers into their respective value chains using these networks from R&D to customer service; in the second case, people around the world share different types of media, known as social media, enabling them to share ideas throughout the world.

It is important here to differentiate social networks from social networking sites. Social networks represent a set of actors and the ties between them. These actors can be people or organizations or both, while the ties may represent friendship, affiliation, dependance or communication for example; these networks exist even if they are implicit, as friendship. Social networking sites are, usually, Web sites intended to make explicit the relationships between actors, e.g., the tie between two persons is stored somewhere to express their friendship.

These social networking sites usually represent the social networks as graphs where actors are the nodes and the ties between them are the edges. However, a social network contains a lot more information than just the actors and their connections: they contain information about each actor, for example reading preferences, age, geographic location, and in general, whatever information available in a given context.

According to [Wasserman and Faust, 1994] a social network contains two variables: a *structural* variable, describing the connections between the actors and a *composition* variable, describing each actor individually according to its own information. Using each one of these variables it is possible to generate a two-mode network (one per variable), or affiliation network that can be used to describe groups of nodes generated either from the structural variable (well connected nodes) or from the composition variable (nodes with similar profile). This affiliation network is described by a third variable called the *affiliation* variable. This variable indicates whether a node belongs, or not, to a specific group.

This PhD thesis presents a general framework for integrating all the variables com-

posing a social network. Our proposition is to combine the structural and composition variables to generate an affiliation variable from this combination to perform different community analyses. Additionally, this proposition includes a visual model to make explicit the influence of the structural and composition variables in the definition of the affiliation variable.

PRESENTATION AND CONTEXT OF THE THESIS

Social networks contain two variables: structural and composition. Each one allows describing and analyzing the social network from different dimensions, however, most of the usual analyses performed are made using either the structural or the composition variable. Each one of these dimensions can be used to produce an affiliation variable that describes the affiliation of each node to one of k groups or communities.

Using only the structural variable all the information contained in the composition variable is discarded, reducing the scope of the possible analyses that can be made. Including the composition variable in the analysis of a network may allow to discover unseen information and to find different affiliation configurations according to different perspectives.

This integration is not a straightforward task, instead it requires the analysis of the representation of the involved variables: the structural variable can be represented as a graph or as an adjacency matrix, the composition variable can be represented as vectors in some $\rho-$dimensional space, and the affiliation variable is usually represented as an inclusion matrix.

Due to the nature of the composition variable it is necessary to analyze how to produce and represent it. In general, this variable can be represented as vector of features describing each actor; this a $\mathbb{R}^f$ vector, where f is the number of features. Despite this representation is "natural" to describe each actor, it poses a series of problems to manipulate and to generate an affiliation network from this variable: first, when f is large, the clustering is subject of the *curse of the dimensionality* [Hughes, 1968], making that each vector has a reduced statistical significance. Second, as a direct implication of the dimensionality issue, when each actor is described by a large set of features the clusters found by a clustering algorithm will not be representative[1], because it is expected that each individual has unique features.

In this work we introduce the concept of *point of view* which allows dividing the set of features into different subsets according to different perspectives from which the affiliation variable can be analyzed. This division can reduce the impact of the issues presented above, but which is more important, allows us to have different perspectives

[1] Or even difficult to interpret

from which the social network can be analyzed; in the case of community detection using different points of view will produce different partitions. These partitions can be seen as multiple affiliation variables (derived from the composition variable) that can influence the structural variable to produce partitions of well connected and similar nodes, which is still an open research question.

SUMMARY OF OBJECTIVES AND GOALS

Given the two variables composing a social network, we want to integrate they in such a way that the social network can be analyzed from a holistic perspective. However, most of the works in social network analysis are based only in the structural variable and in the affiliation variable derived from it.

Thus, the general objective of this work is to create an integration framework that helps generating an affiliation variable, or a partition, aware of both dimensions, the individual information of each actor and of their structure.

To attain this objective the work is divided into three main axes:

1. Design a community detection algorithm that integrates the composition and structural variables.

2. Design a layout algorithm to make explicit the relation between the structural, the composition and the affiliation variables.

3. Given the layout algorithm, design a model to explore the graph according to different points of view and how each of those impact the partition.

The purpose of the first axis is to design an approach to obtain partitions of well connected (structural variable) and similar nodes (composition variable.) One important aspect of this community detection algorithm is the inclusion of the notion of point of view to divide the set of features and produce fine grained communities.

The affiliation variable includes information from the other two variables of the social network. In the case of the structural variable, it is included into the affiliation variable by constructing groups of well connected nodes, i.e., the groups are created in such a way the edges between groups are less that the edges within the groups. Thus the second axis is intended to make explicit the final configuration of the structural variable under the light of the affiliation variable.

The composition variable is included via points of view that will influence the partition configuration in different ways. The purpose of the third axis is to make explicit that influence and observe how, given two partitions of the same social network, they have

been impacted by each point of view. This axis allows making further analyses of the network and the communities configuration, however, in this work we will leave it as a part of the perspectives and future work.

ORGANIZATION OF THE DISSERTATION

The work is divided into five more chapters through which each one of the axes presented above are developed. Below is presented a brief a summary of each chapter.

MOTIVATION AND CONTEXT OF THE THESIS

This chapter presents in detail the variables included into social networks and how their integration may produce interesting results in terms of the analyses that can be performed. Additionally, in this chapter is introduced and formalized the concept of *point of view*, which is used in next chapters when discussing the community detection algorithm and the integration of variables.

Finally, the problem statement and two questions are presented. These questions are aligned to the objectives presented in the introduction and which are answered throughout the thesis.

BIBLIOGRAPHIC REVISION

This chapter presents a bibliographic revision, which is divided into four parts: quality measures, data clustering, community detection and layout of clustered graphs. The structure of this chapter is due to the different components involved with the community detection and visualization model.

The quality measures section contains some of the most used functions to measure the quality of a partition. This section is divided into two subsections, one devoted to quality measures for data partitions and another devoted to quality measures for graph partitions. The difference between these two types of measures lies on the way each one is calculated: in data clustering it is expected the distance between groups to be large while the distance of element within groups to be small, and in the case of graph, it is expected that the number of edges connecting groups be less than all the edges within each group.

The data clustering section explores some important works in *classic* data clustering and in how variables can be represented for each problem. The community detection section explores several approaches to community detection divided according how the

graph is represented. This section finishes presenting some clustering analysis and evaluation measures.

The last section of this chapter presents a bibliographic revision on layout algorithms for clustered graphs. The section explores different approaches to represent a graph partition. Most of the existing techniques search for showing the division between communities, i.e., to represent the inter cluster sparsity and the intra cluster density in a 2 or 3 dimensional space. These representations, however hides the structural variable included into the partition.

Integration of variables in a social network

In this chapter is presented the community detection algorithm which uses the structural and composition variables to produce a partition with groups of close nodes in terms of structure and of their characteristics.

This is accomplished by taking advantage of the way the modularity is calculated: the modularity use the weights of the graph to calculate the communities, the graph is changed to influence the community detection algorithm in such a way the algorithm privileges the creation of groups of similar nodes.

Visualization of communities

In this chapter is introduced a novel layout algorithm for clustered graphs, which is oriented to the connections of the graph. The ultimate goal of this layout is to highlight the interactions between communities and to make explicit the structural variable introduced into the affiliation variable. This layout algorithm, besides from the division of the communities, is capable of using the similarity between nodes to place similar nodes close and dissimilar nodes far away.

Conclusion and perspectives

This final chapter presents the conclusions and the contribution of this PhD thesis. Additionally, there are included here some questions generated during the development of this work.

Publications

- J. D. Cruz, C. Bothorel, and F. Poulet, "Community detection and visualization in social networks: integrating structural and semantic information," *Transactions on*

Intelligent Systems and Technology – TIST, Accepted - To be published in 2013.

- J. D. Cruz, C. Bothorel, and F. Poulet, "Influence de l'information sémantique sur la détection et la visualisation des communautés dans les réseaux sociaux," *Revue d'Intelligence Artificielle*, vol. 26, pp. 369 – 392, October 2012.

- J. D. Cruz, C. Bothorel, and F. Poulet, "Layout algorithm for clustered graphs to analyze community interactions in social networks," in *Proceedings of the 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 717 – 718, IEEE Computer Society / ACM SIGMOD, 2012.

- J. D. Cruz, C. Bothorel, and F. Poulet, "Semantic clustering of social networks using points of view," in *Proceedings of COnférence en Recherche d'Infomations et Applications* (G. Pasi and P. Bellot, eds.), pp. 175–182, Éditions Universitaires d'Avignon, 2011.

- J. D. Cruz, C. Bothorel, and F. Poulet, "Point of view based clustering of socio-semantic networks," tech. rep., LUSSI - Dépt. Logique des Usages, Sciences Sociales et de l'Information (Institut Mines-Télécom-Télécom Bretagne-UEB), IRISA - Institut de recherche en informatique et systèmes aléatoires (INRIA), January 2011.

- J. D. Cruz, C. Bothorel, and F. Poulet, "Entropy based community detection in augmented social networks," in *Proceedings of the 2011 International Conference on Computational Aspects of Social Networks – CASON*, (Salamanca), pp. 163 – 168, 2011

- J. D. Cruz, C. Bothorel, and F. Poulet, "Détection de communautés dans les réseaux socio-sémantiques par point de vue." Journée thématique : fouille de grands graphes, Toulouse, France, 2010.

Chapter 2
Motivation and context of the thesis

2.1 Introduction

According to [Wasserman and Faust, 1994] social networks contain, or can be described by two types of variables: (1) structural variable, which has information about the relationships between all the actors taken in general as a whole, (2) composition variable, which contains characterizing information about the actors as individuals.

Additionally, social networks can be transformed into two-mode networks: one mode for the actors and the other mode for a set of groups to which the actors belong. This kind of two-mode networks are described by a third variable called (3) affiliation variable, that describes the nodes according to their belonging to different groups or events for example.

Figure 2.1 shows an example of a corporate network depicting employees from two different locations: the square nodes represent the New York office, while the circle nodes represent the Boston office.

This example is simple but illustrates each one of the variables introduced above: the structural variable describes the connections between employees, the composition variable contains the name and, for example the duties of each actor, and the affiliation variable is derived from the assignation of each actor to one of the company branches.

Social network analysis seeks for extracting new information from the different variables listed above, however such information extraction is usually performed using one type of variable at the same time. For example, by using only the structural variable it is possible to identify central nodes in the network or take measures over the structure of the graph. On the other hand, using the composition variable, researchers can infer information, as in data mining, first by clustering the data set and then by analyzing each cluster, or by identifying patterns and association rules.

The affiliation variable can be also generated through a community detection process, for example or by studying the membership of actors to groups, associations and even to

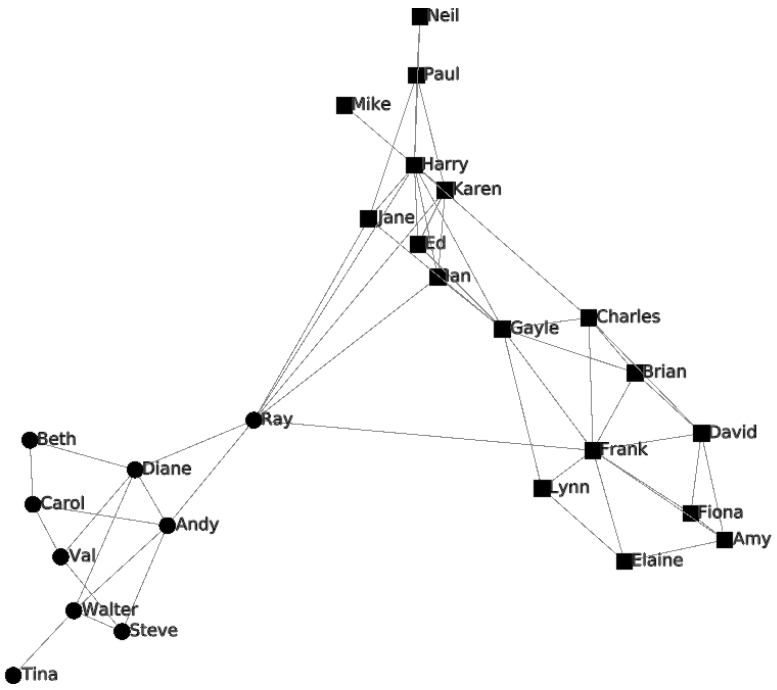


Figure 2.1: Example of a basic social network of a company with offices in two different locations. Adapted from [Cross and Parker, 2004]

study the co-membership. The affiliation variable is in fact a representation of a bipartite graph [Wasserman and Faust, 1994]; a complete study about the general properties of this type of graphs was presented by [Latapy et al., 2008].

In each case a portion of the information is discarded, restricting the number and scope of the analysis. As presented in Chapter 3, the existing literature is very limited regarding the approaches to analyze a social network as a whole: using its different dimensions in an integrated way.

In this work we want to use the affiliation variable to be generated from the integration of the structural and composition variables.

One issue to deal with when analyzing the composition variable is the diversity of values it may contain. For example, this variable contains information about gender, age, nationality, height, and other physical characteristics, but also it may contain information like personal preferences in sports, books, art; despite this variety it may describe well each individual and increase the difficulty to analyze the group of individuals.

One approach to tackle this issue is the division of the profile data into subsets of features representing different aspects of each individual, for example hobbies, preferences and academic competences. As presented in Chapter 4 this division allows to analyze

groups of individuals from different *points of view*.

This chapter hence introduces our notion of *point of view* and how to augment the social graph with this composition information. Additionally it defines the questions we want to answer and outlines our approach contrasting it with some important works in the area.

The chapter is organized as follows: in Section 2.2 the context of the thesis is presented, then in Section 2.3 the problem is defined and the research questions presented, concluding with Section 2.4.

2.2 CONTEXT OF THE THESIS

With the advent of social networking sites such as Facebook[1], LinkedIn[2] or Twitter[3], just to cite some few examples, and the large amount of information being posted online, social networks have become a great source of knowledge about people, their interest, and sometimes, their lifestyle. Although these social networking sites are very popular and people tend to think those are the only examples of social networks, social networks are present in several scenarios, for example in companies, where social structures can be extracted from email messages or from project participation relationships as shown in [Perer et al., 2011, Cross and Parker, 2004].

Social networks have been studied for a long time now: first by sociologists long before the digital era; since late 90s with the introduction of Internet and new ways of communicating, the field took another breath making biologists, sociologists, physicists and computer scientists to start exploiting the social network analysis in their respective fields. Most of the approaches represent the social structure as a graph, like the Web structure, protein interaction graphs and social graphs. This representation is useful to describe how actors are connected, making it possible to find paths between actors and to identify central actors based on their connectivity characteristics.

However, since a social network represents a set of actors and their relationships, several analyses may be performed using additional the information.

In [Wasserman and Faust, 1994] authors define three types of information, or variables within a social network:

1. **Structural:** this information is related to the connections of each individual in the network. Measures of this variable study the nodes in a pairwise way generalizing the measure for the whole network.

[1] http://www.facebook.com/
[2] http://www.linkedin.com/
[3] http://twitter.com/

2. **Composition:** this information is referred to the individual information of each node in the network. It is defined individually, i.e., composition variables describe each actor in a particular context through the definition of a set of features.

3. **Affiliation:** this kind of information represents the mode of a network indicating the attachment of each node to a group, for example a company, a department, a club, a school or a community. Therefore this variable can be defined by any extrinsic criteria related with the particular study being performed. Recall that this variable does not belong to the social network but it is generated from it and is central to our work.

In this PhD thesis we consider that the affiliation variable is the result of a clustering process: using a structural variable, a composition variable or using both. Figure 2.2 shows our general architecture for processing the information that composes the social network and integrating it to a visualization and exploration model. The affiliation variable does not exist *per se* in the social network but instead, it is generated from the other two variables either one by one (see Chapter 3) or by combining both of them.

One of the main contributions of this thesis the proposition of a model of augmented social networks that integrates into the analysis both the structural and composition variables, first by deriving an affiliation variable through a clustering process and second, creating a visualization of the clustered graph.

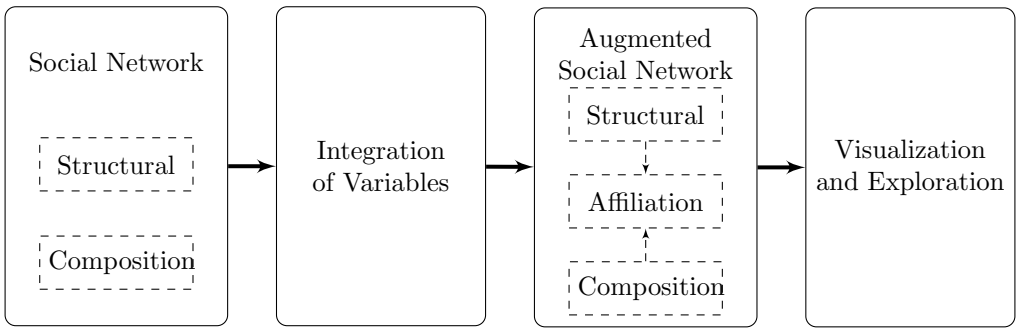


Figure 2.2: Diagram of the general architecture of the system

Once the affiliation variable has been defined by the integration process, and knowing that this affiliation variable describes a partition of the social network, a visualization and exploration scheme is proposed to analyze the structure, and the influence of the structural and composition variables while creating the partition, i.e., this last step helps to make explicit the contribution of the two other variables in the definition of the affiliation variable.

2.3 PROBLEM DEFINITION

Analyzing a social network by integrating its describing variables is not a straightforward process. This integration of variables requires to establish the relationships between them and to create a mechanism to extract new information from the unification of the three variables rather than from each one separately.

The problem is divided into two sub problems: first, how to represent the variables in such a way they can be used and analyzed holistically, and second, how to use a visual model to represent the integration of the three variables involved.

2.3.1 DEFINITION OF THE POINT OF VIEW

Let $\mathcal{S}\left(G\left(V,E\right),F^{*}\right)$ be a social network, where $G\left(V,E\right)$ is an undirected graph representing the structural variable of the social network, where V is the set of n nodes and E is the set of m edges connecting the nodes, and F^{*} is the composition variable of the social network. The composition information F^{*} is therefore defined as a vector of features. Each actor in the network can be described by one instance of F^{*}.

The features included in F^{*} may vary from context to context: personal information, academic and professional backgrounds or a combination of them, making the vector F^{*} to be defined in a $p-$dimensional space. For simplifying purposes the graph $G\left(V,E\right)$ will be referred as G in the rest of the chapter.

Table 2.1 shows an example of the representation of the features for a set of nodes.

	F^{*}				
	f_1	f_2	$\dots$	f_p	
$node_1$	ξ_{11}				$\rightarrow$ Instance Ξ_1
$node_2$					
$\vdots$					
$node_n$				ξ_{np}	

Table 2.1: Example of a set of features describing actors within an augmented social network

Thus each node is represented by an instance Ξ_i, which is a particular set of values for each feature in its respective domain for a specific node.

It is possible to use the set of instances to describe and to analyze the social network, for example to find groups of similar actors or to identify a set of topics of interest for each actor within the network.

However using all of the p features vector may be inconvenient for two main reasons:

- The number p of dimensions may be large and produce the Hughes effect [Hughes, 1968], also known as the "curse of dimensionality", which states that when the number of dimensions is increased, the statistical significance of each element of the data set is reduced.

- The features representing an actor may represent different aspects, related or not. Using the whole set of features may introduce noise by adding columns not so interesting for some particular application. See data reduction [Kantardzic, 2002].

Instead of using systematically the whole set of features, we introduce the notion of *points of view*. A point of view allows us to analyze the social network from different perspectives that can be useful for a particular domain, for example comparing different configurations of the nodes, producing, different partitions with features applied to the same set of nodes.

Definition 2.3.1 (Point of View PoV_{F^*})**.** *Given a set of features F^*, a point of view PoV_{F^*} is one of the p-combinations of the p elements of F^*.*

Thus, $PoV_{F^*} \in \mathcal{P}(F^*)$. Note that this definition includes the $\emptyset$ and F^* itself as points of view. In the first case, a void point of view represents the non-existence of composition variables, thus the social network is taken just as the social graph. On the second case, there is no restriction on the size or components of the point of view, and since $F^* \in \mathcal{P}(F^*)$, F^* can be used a point of view as well.

Definition 2.3.2 (Augmented social network $\mathcal{S}^+$)**.** *Given a social network $\mathcal{S}(G, PoV_{F^*})$, where G is the graph representing the structural variable, PoV_{F^*} the composition variable as a point of view, the augmented social network is defined as $\mathcal{S}^+(G, PoV_{F^*}, \mathcal{A})$, where $\mathcal{A}$ is the affiliation variable derived from the integration process of G and PoV_{F^*}.*

As a consequence, the augmented social network will include the two variables of the social network and an affiliation variable derived from the other two variables. In this case, note that we use PoV_{F^*} as the composition variable meaning that at this point we have selected one point of view from F^*. Including the point of view here enable us to perform a different analysis of the social network. For example, the nodes can be grouped in several ways: one for each point of view, making it possible to contrast different partitions according to the selected features.

2.3.1.1 AFFILIATION VARIABLES REVISITED

As presented before, one of the types of information in the augmented social network is the affiliation information. Affiliation information represents a partition of the nodes

in the graph. This partition can be obtained by grouping the nodes according to their structure, reducing the number of edges between groups, or using the composition information, obtaining groups of nodes whose information is similar. In the scope of this thesis we will use disjoint partitions rather than fuzzy partitions.

Recall that from each point of view it is possible to generate different partitions of the graph. These partitions represent how the nodes are related under certain characteristics and can be used to differentiate the points of view. For example, given two points of view $PoV^1_{F^*}$ and $PoV^2_{F^*}$ and two partitions $\mathbf{C}_1$ and $\mathbf{C}_2$ derived from these points of view, how to measure the differences between the two points of view?

Comparing directly two points of view may be a complex task: each point of view can be composed of incomparable values e.g., boolean values and real values, or the number of features for each point of view can be different. On the other hand, comparing different partitions may produce better results because these partitions can be represented in the same way.

To manipulate the affiliation variables it is necessary to represent them with some structure that allows to do operations over them. These kinds of networks, including affiliation variables are known as two-mode networks [Wasserman and Faust, 1994]; the affiliation variable in these networks is represented as a binary $n \times k$ matrix $\mathbf{A} = a_{ij}, 1 \leq i \leq n, 1 \leq j \leq k$, where $a_{ij} = 1$ if the node i belongs to the group j, 0 otherwise. This means that for non-overlapping partitions the row sum of $\mathbf{A}$ is equal to 1, meaning each node belongs only to one group.

2.3.1.2 USING THE THREE TYPES OF VARIABLES TO ANALYZE THE SOCIAL NETWORK

The affiliation variable $\mathcal{A}$ of an augmented social network $\mathcal{S}^+$ can be derived either from the structural variable, contained in G or from the composition variable contained in PoV_{F^*}. In the first case, we assume that $PoV_{F^*} = \emptyset$, yielding to the general problem of graph clustering. Graph clustering approaches use only the structure to find cohesive groups where the proportion of edges within each group is higher than the number edges between groups. On the other hand, assuming that $G\,(V, \emptyset)$, only PoV_{F^*} is available, the affiliation variable can be generated using traditional data clustering methods that use (typically) vectorial representations of the data. Using this data these methods produce groups of close elements according some measure distance.

Each one of these approaches results in different partitions configurations whose quality measures have a divergent nature. As presented in [Cruz et al., 2011a], due to the different quality measures applicable to each variable, merging both variables will produce partitions with a modularity and an entropy below the optimum: if the

modularity is optimized, groups will probably have elements with different composition variables, making the partition to have a high entropy. On the other hand, optimizing the entropy to have a partition of groups with nodes described by similar composition information cause the partition to have a low modularity value.

There is a gap between the available clustering approaches designed for each one of these kind of variables. This gap opens a new study field, looking for new ways to generate affiliation variables that integrate the structural and the composition variables.

Therefore the question is: **how to integrate the structural and composition variables (as points o view) to analyze a social network and its associated affiliation variables?**

The first contribution of this thesis is the proposition of a community detection method that integrates the structural and the composition variables to derive a new affiliation variable. Our method doe not require to make any assumption about the data set, making the number of communities to be an output of the algorithm instead of an input.

2.3.2 INTEGRATION AND VISUALIZATION OF THE SOCIAL NETWORK INFORMATION

Affiliation information from a social network can be analyzed using a visual analytics approach, which allows to graphically differentiate the groups present in the partition; however it does not allow extracting information from the other variables that are already incorporated into the affiliation one.

In general, visualization methods for clustered graphs do not naturally include other information than the configuration of each group. Thus, one way to answer the question presented in the previous section is by developing a visual model which integrates the structural variables and different configurations of the composition variables.

Thus, an integrated layout of the communities should:

1. Integrate structural variables: represent the affiliation information in such a way the structure of the social network can still be exploited for analysis.

2. Integrate composition variables: show how the nodes positions are influenced by the selected features.

The first requirement allows us to identify connections between groups by taking advantage of the existing affiliation variable and the structure of the graph. The second requirement seeks for modifying the visual configuration according to the composition variables to reflect this additional information from actors in the network.

The second contribution of this thesis is the proposition of a visualization model that uses the structural and affiliation variables to present the augmented social network in such a way the position of the nodes unveils their structural and compositional similarity. Additionally, this model allows for using the composition variable to observe the influence of this variable on the position of each node.

This visualization model does not include a navigation scheme like focus+context. The graph navigation poses additional challenges for the visual model which are beyond the scope of this PhD thesis, we hence leave it as one of the perspectives.

2.4 Conclusion

In this chapter we presented the context of the thesis and the problem definition. Our general purpose is to integrate the different variables of social networks to extract non-evident information from them.

The studied literature, presented in Chapter 3 reveals that most of the approaches, like community detection, use only one type of the variables at the same time, limiting the scope of the analysis. The integration and the holistic analysis of the variables is a very challenging process.

First, we have proposed a formal approach for modeling the set of composition variables. This formalization divide the composition variable into subspaces that can be used to analyze the social network from different perspectives, called points of view.

The definition of point of view provide us a way to manipulate the composition variables, which can be very rich and variate because of their nature.

Second, we have posed two principal questions:

1. How to integrate structural and composition (as points of view) variables to generate an affiliation variable to induce a partition with groups of well connected and similar nodes.

2. How to design a visual model to analyze the affiliation variables under the light of the other two variables.

These questions guide the development of the present work. We claim, and give here some clues, that the answers to these questions lead to novel methods and tools for identifying new, non-evident information.

Our contributions will be presented in the next chapters: first the community algorithm integrating the structural and composition variables. Second we present the algorithm to produce a layout exploiting the structural and affiliation variables of the clustered graphs. Finally, we present a model to observe the influence of using different points of view on a layout.

Chapter 3

State of the art on community detection and layout of clustered graphs

3.1 Introduction

In social network analysis, different approaches are used to analyze a network. These approaches can be divided into individual, relational or group measures, as presented by [Wasserman and Faust, 1994].

Additionally to these measures, there are other techniques to help the analysis of social networks such as community detection and graph layout. In the first case, the idea is to find partitions of the set of nodes of a graph according to a non-structural criterion, for example the membership of actors to some club or organization, or to a structural criteria such as connectivity characteristics of the nodes. Through the analysis of groups it is possible to extract some non-evident information about the network and the actors within it.

The graph layout models allow to display the network in some readable and interpretable view, ready to perform visual analyses. In general, these techniques use some notion of distance to locate the nodes in order to map their "proximity" using some measure of the distance, from a $\rho-$dimensional space to a $\mathbb{R}^2$ or $\mathbb{R}^3$ space. In the case of clustered graphs, as for graphs partitions, additionally to the node placement, one goal is to clearly represent each group inside the partition.

This chapter presents some important works in the areas of community detection and layout of clustered graphs.

Section 3.2 presents a summary of general data clustering, including elements of quality measures, data types and algorithms for clustering data. These works will are included to take manipulate the composition variable of the social network. Next, Section 3.3 presents a summary of relevant graph clustering and community detection algorithms, including quality measures and partition comparison. These works will serve as reference

for manipulating the structural variable. Finally Section 3.4 presents methods for drawing graphs and clustered graphs; in this section are also included some approaches to graph navigation and exploration.

Thus the structure of this chapter cover both dimensions of the information within a social network and additionally includes works about layout of clustered graphs.

3.2 DATA CLUSTERING

The goal of data clustering is to find a *good enough* partition of the data set, where good partition means, a set of groups with a small intra-cluster distance and a big inter-cluster distance. Although there is not a consensus about what is a good distance measure, but instead it may vary from application to application and from data set to data set. Another feature of the clustering methods is that the input data 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 in $\mathbb{R}^n$, but, despite they may be composed of other types of information such as categorical data or non-numeric values, the typical representation of each element, or pattern, is a vector. In [Gordon, 1999] the following types of data for clustering are defined:

- Binary variables.
- Quantitative variables.
- 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 [Jain and Dubes, 1988], are:

Given the $i-$th and the $k-$th patterns from a data set, a proximity measure, denoted by $d(i,k)$ must satisfy:

1. (a) For dissimilarity: $d(i,i) = 0 \ \forall i$
   (b) For Similarity: $d(i,i) \geq \max_k d(i,k) \ \ \forall i$
2. $d(i,k) = d(k,i) \ \ \forall i,k$
3. $d(i,k) \geq 0 \ \forall i,k$

   According to [Jain and Dubes, 1988], Minkowski-based metrics must satisfy two additional properties:

4. $d(i,k) = 0$ iff, given two patterns $\mathbf{x}_i$ and $\mathbf{x}_k$, $\mathbf{x}_i = \mathbf{x}_k$.

5. $d(i,k) \leq d(i,m) + d(m,k)\ \forall i, m, k$.

The proximity measures between data patterns should be defined in terms of the type of data, i.e., binary, quantitative and nominal and ordinal variables according to [Gordon, 1999].

- **Binary variables:** these are variables which belong only to two states, e.g., 1, 0 or $(+)$, $(-)$. Using the matrix representation proposed by [Jain and Dubes, 1988] we define the p possible values for two patterns $\mathbf{x}_i$ and $\mathbf{x}_j$:

$$\begin{array}{ccc} & \mathbf{x}_j & \\ & 1 & 0 \\ \mathbf{x}_i \quad 1 & a_{11} & a_{10} \\ 0 & a_{01} & a_{00} \end{array}$$

  Thus $p = a_{00} + a_{01} + a_{10} + a_{11}$. Note that a_{11} and a_{00} are the number of agreements between the two patterns. Using the values of p it is possible to define the following measures:

  - **Simple Matching Coefficient:** weights the number of agreements. Is expressed as:

$$s(i,k) = \frac{a_{00} + a_{11}}{p} \tag{3.1}$$

  - **Jaccard Coefficient:** weights the only the 1s of the patterns. This means it only takes into account the values of the patterns which match 1 to 1, but discarding the 0 to 0 matches. It can be expressed as:

$$s(i,k) = \frac{a_{11}}{p - a_{00}} \tag{3.2}$$

    This coefficient can be generalized as the size intersection divided by the size of the union of the patterns compared:

$$s(i,k) = \frac{|\mathbf{x}_i \cap \mathbf{x}_j|}{|\mathbf{x}_i \cup \mathbf{x}_j|} \tag{3.3}$$

– **Dice-Sorensen Coefficient:** this coefficient was first used to compare the ecologic association between species by [Dice, 1945]. The coefficient is given by:

$$s(i,k) = \frac{2 \cdot a_{11}}{2 \cdot a_{11} + a_{01} + a_{10}} \tag{3.4}$$

This coefficient has been used to measure the similarity between fuzzy sets by [Roberts, 1986] and to measure the distance between words by [Rijsbergen, 1979].

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

- **Multi-state nominal and ordinal variables:** these are variables for which there are more than two states or categories [Gordon, 1999]. If those categories are ordered, the variables are called ordinal, otherwise are nominal. This measure sums the contributions of each category over all the variables. This is, defining disagreements indices [Gordon, 1999] between each pair of categories as $\delta_{klm} \geq 0$, where l and m are categories of the kth variable. In the case of nominal variables, $\delta_{klm} = 1$ if $l \neq m$ and $\delta_{kll} = 0$. For ordinal variables, for which the categories are ordered: $\delta_{kml} < \delta_{klr}$ if $l > m > r$ or $l < m < r$. Thus, the contribution to the dissimilarity d_{ij} between the ith and the jth elements that is made by the kth category is defined as $d_{ijk} = \delta_{klm}$. The contribution to the similarity s_{ij} can be defined by $s_{ijk} = 1 - d_{ijk}$.

  – **General nominal / ordinal similarity measure** [Gordon, 1999]:

$$s_{ij} = \frac{1}{p} \sum_{k=1}^{p} s_{ijk} \tag{3.5}$$

where p is the number of variables.

- **Quantitative variables:** it is possible to define a pattern matrix $[x_{ij}]$ such that x_{ij} is the jth feature for the ith pattern. In this kind of variables, the features are continuous. The most common proximity index for these variables are the Minkowski – based measures [Jain and Dubes, 1988]. The Minkowski distance is defined by:

$$d(i,k) = \left(\sum_{j=1}^{p} |x_{ij} - x_{kj}|^r \right)^{\frac{1}{r}} \tag{3.6}$$

where $r \geq 1$, p is the number of features and r is a parameter for changing the way in which the measure is taken. This measure, and it derivations, satisfy the

properties 4 and 5 stated above. Changing the value of r it is possible to obtain other well known measures:

- **Manhattan distance:** this measure is obtained when $r = 1$ and is defined by:

$$d(i,k) = \sum_{j=1}^{p} |x_{ij} - x_{kj}| \tag{3.7}$$

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

$$d(i,k) = \left(\sum_{j=1}^{p} |x_{ij} - x_{kj}|^2 \right)^{\frac{1}{2}} \tag{3.8}$$

- **Chebyshev distance:** this measure is obtained when $r \to \infty$. It is defined by:

$$d(i,k) = \lim_{r\to\infty} \left(\sum_{j=1}^{p} |x_{ij} - x_{kj}|^r \right)^{\frac{1}{r}} = \max_{1\leq j\leq p} |x_{ij} - x_{kj}| \tag{3.9}$$

Figure 3.1 shows an example of the application of three Minkowsky-based measures. Each of these measures can be used in any $n-$dimensional space, however for high values of n there will be the Hughes effect, or *curse of the dimensionality* [Hughes, 1968], which in general refers to the sudden increase of the volume of a hyperplane when the number of dimensions is increased, causing a sparsity in the data and reducing its statistical significancy, and in machine learning applications, with a finite set of training patterns, reducing the prediction capability of the machine.

Other measures for quantitative variables are presented in [Gordon, 1999]:

- **Canberra metric:** this metric is similar to the Manhattan distance, but each term is divided by the sum of the absolute values of each component. This makes this metric to sensible to values close to zero.

$$d(i,j) = \frac{\sum_{k=1}^{p} |x_{ik} - x_{jk}|}{(|x_{ik}| + |x_{jk}|)} \tag{3.10}$$

  where p is the dimension of the points.

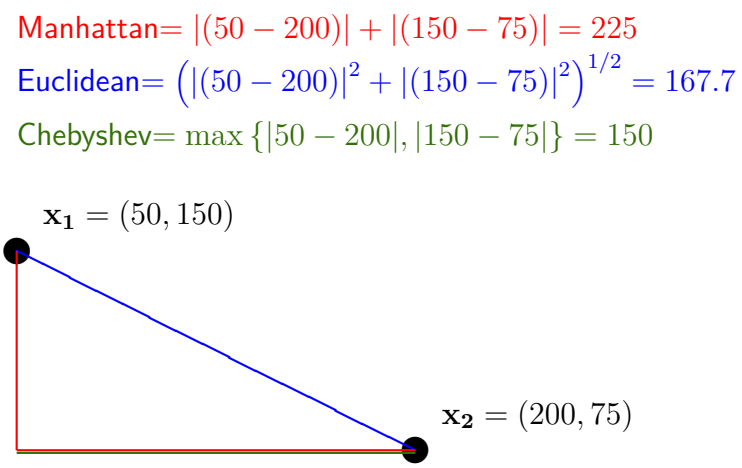


Figure 3.1: Example of the Minkowsky-based measures

– **Angular separation:** this measure is also known as cosine distance. It takes values from -1, meaning the patterns are opposite, to 1, meaning the patterns are equal. It is defined by:

$$d(i,j)=\frac{\sum_{k=1}^{p}x_{ik}x_{jk}}{\left(\sum_{k=1}^{p}x_{ik}^2\sum_{l=1}^{p}x_{jl}^2\right)^{\frac{1}{2}}} \tag{3.11}$$

or in vectorial notation:

$$d(i,j)=\frac{\mathbf{x}_i\cdot\mathbf{x}_j}{\|\mathbf{x}_i\|\|\mathbf{x}_j\|} \tag{3.12}$$

– **Correlation coefficient:** this measure is based on the same vectorial idea from cosine distance. The difference is this takes into account the relation of each variable with the average of the pattern. It is defined by:

$$d(i,j)=\frac{\sum_{k=1}^{p}(x_{ik}-\overline{x}_{i.})(x_{jk}-\overline{x}_{j.})}{\left(\sum_{k=1}^{p}(x_{ik}-\overline{x}_{i.})^2\sum_{l=1}^{p}(x_{jl}-\overline{x}_{j.})^2\right)^{\frac{1}{2}}} \tag{3.13}$$

where $\overline{x}_{i.} = \frac{1}{p}\sum_{k=1}^{p} x_{ik}$.

Thus, clustering techniques will, based on the intrinsic information from a data set, find natural groups according to some similarity or distance measure. Clustering techniques could be divided into agglomerative and partitional [Jain and Dubes, 1988]. Some clustering methods are summarized below.

3.2.1 PARTITIONAL CLUSTERING

These algorithms divide the data set into k groups and then assign each point to one group according to the distance to the group's center of mass. Techniques in this category are known for using computational resources efficiently [Zhao and Karypis, 2002] however, one of their drawbacks is the selection of the initial k value and the initial centroids. Some examples of partitional clustering are:

- **K-means:** This technique presented in [MacQueen, 1967] assign each point from the data set to one of the k groups according to some similarity criterion. Nowadays, this algorithm is widely used [Mirkin, 2005] 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.

- **Fuzzy $c-$means:** This technique presented by [Bezdek, 1981] is based on the same principle as the $k-$means, the difference is that each point has a belonging degree to clusters. This degree is not only used to define the composition of clusters, but to update the centroids of the clusters, which now is calculated as a weighted average of the position. This technique inherits also the two disadvantages of $k-$means.

- **Entropy categorical clustering:** This technique presented by [Li et al., 2004], 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 $\mathbf{C}$ the following expression is used:

$$\mathcal{H}(\mathbf{C}) = \sum_{k=1}^{K} H(C_k) \tag{3.14}$$

  where $H(C_k)$ is the measure of the entropy of the group k of the partition $\mathbf{C}$, and its given by:

$$H(C_k) = -\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} s_{ij} \ln s_{ij} + (1 - s_{ij}) \ln (1 - s_{ij}) \tag{3.15}$$

  Where $0 > s_{ij} > 1$ is a similarity measure between the elements i and j. This technique has been designed to work with categorical data, i.e., with data which not have an inherent distance measure, for example, binary attributes or attributes like colors and names, in this case, s_{ij} can be one of measures for binary variables (equations 3.1, 3.2 and 3.4) or for nominal variables (equation 3.5).

Except for the fuzzy $c-$means, the previous techniques are designed to find disjoint groups, which means that each point will belong only to one cluster. This could reduce the noise robustness by assigning outliers points to some clusters.

3.2.2 AGGLOMERATIVE CLUSTERING

These techniques assign first each point in the data set to its own cluster, then the algorithm fusion pair of clusters according to their distance forming coarser partitions until there is only one group: all the nodes belonging to the same cluster [Kantardzic, 2002]. The main drawback of this type of algorithms is their computational complexity because of the comparison of the distance between the points of the data set. Some examples of agglomerative clustering are:

- **Hierarchical clustering:** these techniques place each object in a single group and gradually merge those atomic groups into larger and larger clusters until all the objects are in a single cluster. This merging process leads to the construction of a dendrogram, which represents the hierarchy of the groups inclusion. Using this dendrogram it is possible to select the number of groups of the partition.

- **Gravitational clustering:** this method, proposed by [Gómez et al., 2003], uses the gravitational theory and the second Newton's motion Law to find the clusters. This technique automatically determines the number of clusters and uses a discrimination technique to make it robust to noise and outliers. This is another agglomerative approach, which means that it is possible to obtain different resolution levels according to different criteria.

- **Kohonen Maps:** this method proposed by [Kohonen, 1997] also known as Self-Organizing Maps (SOM), is used to find groups based on the similarity of the input features, or patterns. In general this method does not need any *a priori* knowledge about the data, just the definition of the size of the grid, which can be derived from the data set size [Chifu and Letia, 2010]; it also has several advantages regarding other clustering methods such as $k-$means, which is less robust to high-dimensional patterns and in which the number of k clusters has to de defined as an input for the algorithm [Kantardzic, 2002]. Additionally, the result of the SOM process can be used to analyze qualitative variables, such as the semantic information by using a 2D representation of the map (U-matrix) and heat maps among others [Kohonen, 1997].

3.2.3 INCREMENTAL CLUSTERING

Another type of clustering is the incremental clustering. Algorithms of this type will create groups as needed according to the distance of the new arriving input to the existing groups.

In general the algorithms start by choosing an element from the data set and creating a group to store it. Then, another element is picked from the data set and is put into an existing group if its distance to that group is less than some threshold, if not, a new group is created.

The advantage of this kind of algorithms, according to [Kantardzic, 2002] is the efficient use of resources, however, these techniques do not satisfy the order-independence of the clustering process: choosing a particular order in the selection of elements will produce different partitions [Kantardzic, 2002].

3.2.4 SUBSPACE CLUSTERING ALGORITHMS

These algorithms can be seen as feature selection algorithms to find clusters that can exist in some dimensions but not in others. In general, by using a projection of the data in different dimensions it is possible to find groups of data that may not exist in other

dimensions or that could not be identified using all the dimensions of the data. Some of this type of algorithms are described below.

- **CLIQUE:** This method proposed by [Agrawal et al., 1998], divides the search space using a grid that is the division of each subspace of the data set into ξ equal length intervals. Each cell of the grid is said to be dense if it contains more than τ percent of elements inside. The algorithm is divided into three steps: (1) identification of subspaces with clusters, (2) identification of clusters and (3) generation of minimal description of the clusters. The first step uses an approach similar to *Apriori* to find dense subspaces. Then, using a measure of coverage, the subspaces are sorted and only some are selected. In the second step the algorithm finds the clusters by selecting adjacent regions which is equivalent to finding connected components in a graph. The last step is the selection of the clusters with a maximal region. From these regions the algorithm removes any redundant region, returning the hyper-rectangular regions that describe the clusters.

- **ENCLUS:** This method presented by [Cheng et al., 1999], is an extension of CLIQUE. It uses the same three steps as CLIQUE, however the first step is changed to use entropy as disorder measure for each dimension. Thus, if a subspace has a cluster, that indicates that the distribution is skewed and the entropy is low, in which case the subspace is selected. The other two steps are the same as in CLIQUE.

- **MAFIA:** The approach presented by [Nagesh et al., 1999] is another extension of the CLIQUE algorithm. In this case the authors propose the use of an adaptive grid instead of fixed values. According to the authors this change produces better results because of the reduction of computation required and the concentration on zone of high density, increasing the likelihood of having clusters. Additionally, this algorithm was designed to be executed on parallel, which improves its performance.

- **Cell-based clustering:** The method proposed by [Chang and Jin, 2002] defines an index to create each hyper-rectangular region to partition the space in an efficient way. Then using a τ threshold parameter, it is decided whether the data within a cell is included into a cluster or not.

Other methods in this category include **CLTree** [Liu et al., 2000], that divides the space using a decision tree that is built by exploring the existing subspaces and calculating the proportion of the data according to a set of predefined classes. The second step is pruning process to identify the actual clusters in the data set. Other method, **Density-based Optimal projective Clustering - DOC** proposed by [Procopiuc et al., 2002]

uses a Monte-Carlo approach to select a discriminant set X that is used to find a pair $(\mathcal{C}, \mathcal{D})$, where $\mathcal{C}$ is a subset of the instances and $\mathcal{D}$ is a subspace of the dimensions that maximizes a quality function.

3.3 COMMUNITY DETECTION IN SOCIAL NETWORKS

Community detection, or graph clustering, is a set of techniques to find groups of nodes according to a similarity criterion; in data clustering the goal is to find groups of points with a small distance within groups and a large distance between groups, in community detection the goal is to find groups of well connected or semantically close nodes or both.

Several methods have been developed to find clusters in a graph, or which is equivalent, to find communities in a social network. In general, those methods have been defined as optimization problems where the objective function is the maximization of some quality index. The indices measure the quality of a partition $\mathbf{C}$ based on the number of edges within the cluster and the number of inter–cluster edges.

3.3.1 PRELIMINARY DEFINITIONS

Let $G = (V, E)$ be a directed graph, with $n = |V|$ the number of vertices, or nodes, and $m = |E|$ the number of edges. A partition $\mathbf{C} = \{C_1, C_2, ..., C_k\}$ where, each C_i, is a non-empty subset of nodes, V, and $k = |\mathbf{C}|$. Thus, $V = \bigcup_i C_i$, $1 \leq i \leq k$; $\overline{C_i}$ is the set of all nodes which do not belong to the cluster i. $E(C_i, C_j)$ is the set of all edges that have their source in C_i and their target in C_j, then the set of intra-cluster edges is $E(\mathbf{C}) = \bigcup_{i=1}^{k} E(C_i, C_i)$, $\overline{E(\mathbf{C})}$ is the set of inter-cluster edges [Gaetler, 2005]. $G(C_i)$ is the subgraph induced by the $i-$th cluster.

In the prior definition, it is assumed that G is a directed and unweighted graph, but it is possible to do a generalization in which the graph is defined as $G = (V, E, \omega)$, where ω is a positive edge weighting $\omega : E \to \mathbb{R}^+$ [Brandes et al., 2008]. Thus, let us denote $\omega(\mathbf{C})$ as the weight of all intra-cluster edges and $\overline{\omega(\mathbf{C})}$ as the weight of all inter-cluster edges.

3.3.2 CONNECTIVITY BASED QUALITY INDICES

These indices are based on the topology of the network, i.e., the number and density of edges within clusters. In general as presented by [Zaidi et al., 2010] clustering indices should be designed to take into account:

- Density: the ratio of edges within the cluster in respect to all the edges of the graph.
- Separation: nodes within a group should have more contacts inside their own group.
- Mutuality: most of the neighbors of nodes within a group are in the same group.
- Compactness: nodes from the same group are close (in terms of geodesic distance).

From this, two types of measures can be identified, one based on the path structure of the graph and another based on the edge density of the clusters.

3.3.2.1 PATH STRUCTURE BASED MEASURES

These measures use the path length between nodes in order to calculate how good is a partition. They are oriented to privilege mutuality and compactness [Zaidi et al., 2010]. The measure proposed is defined as:

$$M(G) = M^{+}(G) - M^{-}(G) \tag{3.16}$$

where $M^{+}(G)$ and $M^{-}(G)$ are positive and negative scores respectively assigned to the clustering according to the criteria defined.

Thus, given a partition $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$, the components of equation 3.16 are defined by:

$$M^{+}(G) = \frac{1}{k}\sum_{i=1}^{k}\frac{1}{AvGPathLen_i} \tag{3.17}$$

where $AvGPathLen_i$ is the average path length of the nodes within the cluster i, and:

$$M^{-}(G) = \frac{2}{k \times (k-1)}\sum_{\substack{i,j=1 \\ i \neq j}}^{k}\frac{E(C_i, C_j)}{|C_i| \times |C_j|} \tag{3.18}$$

Thus, this metric privileges the average path length as measure of compactness and mutuality when used with sparse graphs.

3.3.2.2 DENSITY BASED MEASURES

These indexes are based on the assumption that within each community there is a high connectivity among nodes and a low connectivity between communities. The first is called intra-cluster density and the second is called inter-cluster sparsity.

[Brandes et al., 2008] define a general function framework for measuring the quality of the partitions found by some algorithm. This framework is composed of two independent functions f and g that measure the intra-cluster density and the inter-cluster sparsity respectively. This index is shown below:

$$index\left(\mathbf{C}\right) = \frac{f\left(\mathbf{C}\right) + g\left(\mathbf{C}\right)}{N\left(G\right)} \tag{3.19}$$

where $N\left(G\right)$ is a normalization function usually set to $\max\left\{f\left(\mathbf{C}'\right) + g\left(\mathbf{C}'\right) : \mathbf{C}' \in A\left(G\right)\right\}$, and $A\left(G\right)$ is the set of all possible clusterings [Gaetler, 2005].

These indices have two main purposes: first, rate a partition with respect to clustering paradigms and second, they compare the clusterings regarding quality according to [Gaetler, 2005]. Some of the most used measures based on the equation 3.19 are shown below.

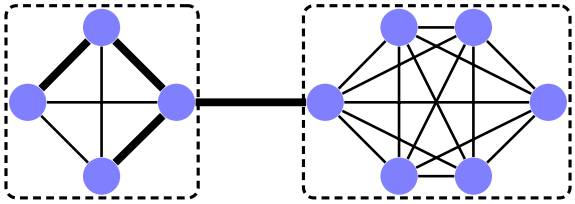
- **Coverage:** $\gamma\left(\mathbf{C}\right)$ measures the weight of all the intra-cluster edges, compared with the weight of all edges [Gaetler, 2005] within the graph. Thus, $f\left(\mathbf{C}\right) = \omega\left(E\left(\mathbf{C}\right)\right)$ and $g\left(\mathbf{C}\right) = 0$. The normalization function $N\left(G\right)$ is given by $\omega\left(E\right)$, then:

$$\gamma\left(\mathbf{C}\right) = \frac{\omega\left(E\left(\mathbf{C}\right)\right)}{\omega\left(E\right)} = \frac{\sum_{e \in E(\mathbf{C})} \omega\left(e\right)}{\sum_{e \in E} \omega\left(e\right)} \tag{3.20}$$

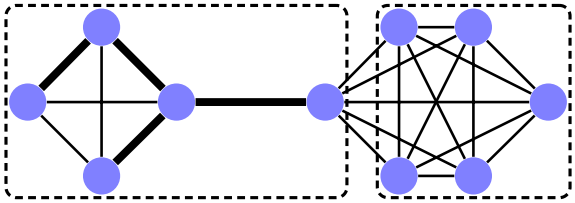
  Coverage just measures the sum of edges weights within the clusters, which can lead to clusterings with a large number of inter-cluster edges and/or a reduced number of intra-cluster edges. An example of a clustering using the coverage measure is shown in the Figure 3.2.

  The Figure 3.2(a) shows the intuitive way to define the clusters in the graph. The thickness of the edges correspond to their weights (weight of bold ones $\gg$ weight of normal ones), thus, using the optimal value for the coverage measure is obtained when the graph is clustered as is shown in the Figure 3.2(b).

- **Conductance:** $\varphi\left(G\right)$ this measure is based on the observation that if a cluster which is well connected then, a large number of edges have to be removed in order to bisect it. Then, the conductance $\varphi\left(G\right)$ of a graph G is the minimum



(a) Intuitive clustering



(b) Best coverage clustering

Figure 3.2: Example of a graph clustering using the coverage measure, adapted from [Gaetler, 2005]

conductance value over all cuts of G [Brandes et al., 2008], this is, the lower possible value of the weight of all edges between the clusters of a partition $\mathbf{C}$.

Along with the graph conductance, there exist two other measures: the intra-cluster conductance $\alpha(\mathbf{C})$ and the inter-cluster conductance $\delta(\mathbf{C})$.

The intra-cluster conductance is the minimum conductance value over all induced subgraphs $G(C_i)$, while the inter-cluster conductance is the maximum conductance over all induced cuts $\left(C_i, \overline{C_i}\right)$. Thus, given a cut $\mathbf{C} = \left(C, \overline{C}\right)$, the conductance of $\alpha(C)$, $\varphi(C)$ and $\varphi(G)$ can be defined as follows [Brandes et al., 2008]:

$$\alpha(C) = \{2 \sum_{e \in E(C)} \omega(e) + \sum_{f \in E\left(C, \overline{C}\right)} \omega(f)\}$$

$$\varphi(C) = \begin{cases} 1, & C \in \{\emptyset, V\} \\ 0, & C \notin \{\emptyset, V\} \wedge \overline{\omega(\mathbf{C})} = 0 \\ \frac{\overline{\omega(\mathbf{C})}}{\min\left(\alpha(C), \alpha\left(\overline{C}\right)\right)}, & \text{otherwise} \end{cases} \tag{3.21}$$

$$\varphi(G) = \{\min_{C \subseteq V} \varphi(C)\}$$

The intra-cluster conductance of a partition $\mathbf{C}$ as:

$$\alpha(\mathbf{C}) = \min_{i \in \{1,\ldots,k\}} \varphi(G(C_i)) \tag{3.22}$$

and the inter-cluster conductance of a partition $\mathbf{C}$ by:

$$\delta(\mathbf{C}) = \begin{cases} 1, & if\, \mathbf{C} = \{V\} \\ 1 - \max_{i \in \{1,\ldots,k\}} \varphi(C_i), & \text{otherwise} \end{cases} \tag{3.23}$$

In order to express the precedent indices in the form of the general framework 3.19, $g = 0$ for intra-cluster conductance, $f = 0$ for inter-cluster conductance and setting $N = f + g = 1$ for both cases.

- **Performance:** this measure defines the quality of a partition based on the "correctness" of the classification of a pair of nodes. This correctness depends on either two connected nodes which belongs to the same cluster or two not connected nodes which belongs to different clusters. Thus, the density function f counts the number of edges within all the clusters while the sparsity function g counts the nonexistent edges between clusters [Gaetler, 2005], this is, the number of not connected pair of nodes among all the clusters. Then, these functions can be defined as:

$$\begin{aligned} f(\mathbf{C}) &= \sum_{i=1}^{k} |E(C_i)| \\ g(\mathbf{C}) &= \sum_{u,v \in V}^{k} [(u,v) \notin E]\, I_{i,j}(u,v) \end{aligned} \tag{3.24}$$

where I is a function which can be 1 or 0:

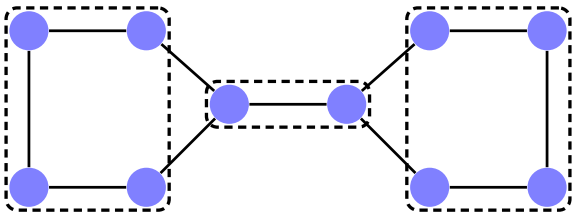
$$I_{i,j}(u,v) = \begin{cases} 1, & u \in C_i \wedge v \in C_j, i \neq j \\ 0, & \text{otherwise} \end{cases} \tag{3.25}$$

Thus, the performance as presented by [Brandes et al., 2008] is:

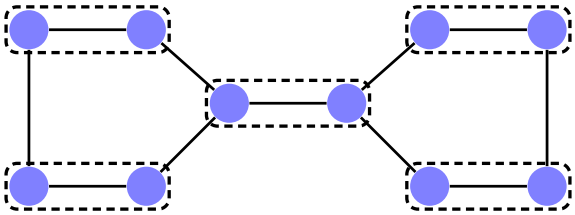
$$perf(\mathbf{C}) = \frac{f(\mathbf{C}) + g(\mathbf{C})}{\frac{1}{2} n(n-1)} \tag{3.26}$$

where n is the number of nodes.

Figure 3.3 shows two possible clusterings of the same graph. Figure 3.3(a) shows an intuitive clustering. Figure 3.3(b) presents the clustering with the best performance.



(a) Intuitive clustering



(b) Clustering with best performance

Figure 3.3: Example of a graph clustering using the performance measure adapted from [Gaetler, 2005]

- **Modularity:** this measure $Q(\mathbf{C})$, proposed by [Newman and Girvan, 2004], compares the fraction of the edges within each cluster with the fraction of edges among clusters, i.e., the intra-cluster edges density versus the inter-cluster sparsity.

  Given a partition $\mathbf{C}$ of k communities, let $\mathbf{e}$ be a $k \times k$ symmetric matrix whose element $e_{i,j}$ is the fraction of the edges in the network that link nodes in the cluster C_i to nodes in the cluster C_j, then the function proposed by [Newman and Girvan, 2004] to calculate the modularity is:

$$Q(\mathbf{C}) = \sum_i \left(e_{i,i} - a_i^2\right) = \mathrm{Tr}(\mathbf{e}) - \left\|\mathbf{e^2}\right\| \tag{3.27}$$

  where $\mathrm{Tr}(\mathbf{e}) = \sum_{\mathrm{i}} \mathrm{e_{i,i}}$, is the trace of matrix $\mathbf{e}$ and $a_i = \sum_j e_{i,j}$ which represent the fraction of edges that connect to community i. In order to put the Equation 3.27 in the notation of Equation 3.19, it is possible to use the notation used by [Blondel et al., 2008]:

$$Q(\mathbf{C}) = \frac{1}{2w} \sum_{i,j} \left[A_{i,j} - \frac{k_i k_j}{2w} \right] I(C_i, C_j) \tag{3.28}$$

where i and j are nodes, $A_{i,j}$ represents the weight between the node i and the node j, $k_i = \sum_j A_{i,j}$ the sum of the weights of the edges attached to node i, $I(C_i, C_j) = 1$ if $i = j$, 0 otherwise and $w = \frac{1}{2} \sum_{i,j} A_{i,j}$.

Thus, since the function I is equal to 1 only if $i = j$, then $k_i k_j$ can be rewritten as k_i^2. Then, the density function f can be expressed as:

$$\begin{aligned} f(\mathbf{C}) &= \omega(E(\mathbf{C})) \\ g(\mathbf{C}) &= -\sum_{i=1}^{k} \left[\frac{\left(\sum_{e \in E(C_i, V)} \omega(e) \right)^2}{\omega(E)} \right] \end{aligned} \tag{3.29}$$

and the normalization function as:

$$N(G) = \omega(E) \tag{3.30}$$

Using equations 3.29 and 3.30, modularity can be expressed as:

$$Q(\mathbf{C}) = \frac{\omega(E(\mathbf{C})) - \sum_{i=1}^{k} \left[\frac{\left(\sum_{e \in E(C_i, V)} \omega(e) \right)^2}{\omega(E)} \right]}{\omega(E)} \tag{3.31}$$

The modularity takes the intra-cluster density minus the expected number of inter-cluster edges in a randomly generated graph. This measure is similar to coverage, the difference is the inclusion of the g function which is 0 for coverage.

This definition of modularity is used to evaluate disjoint partitions, however, in social networks it is possible to have overlapping partitions. To evaluate these kind of clusterings in [Liu, 2010] author proposes the fuzzy modularity Q_f measure. This measure is defined as:

$$Q_f = \frac{1}{2m} \sum_{i=1}^{k} \sum_{u,v \in C_i} \left(\frac{\rho_i(u) + \rho_i(v)}{2} e(u,v) - p_f^E(u,v) \right) \tag{3.32}$$

where $\rho_i(u)$ is the probability of the node u belonging to the ith community and $p_f^E(u,v)$ is the expected stochastic number of edges, given by:

$$p_f^E(u,v) = \frac{d_f(u)d_f(v)}{2m} I(u,v) \tag{3.33}$$

where $d_f(u)$ is the extended degree of node u and $I(u,v)$ is an indicator function which is equal to 1 is u and v belong to the same community, 0 otherwise. The extended degree of a node is the measure of the degree under the probabilistic assumptions. It is given by:

$$d_f(u) = \sum_{z \in C_k} \frac{\rho_k(u) + \rho_k(z)}{2} e\,(u,z) + \sum_{z \notin C_k} \frac{\rho_k(u) + (1 - \rho_k(z))}{2} e\,(u,z) \tag{3.34}$$

where $u \in C_k$. An algorithm for finding fuzzy partitions maximizing Q_f will be shown in Section 3.3.4.

A summary of the previous connectivity based measures is presented in Table 3.1.

Index Name	Expression	Limits	Direction
Coverage	$\gamma(\mathbf{C}) = \frac{\omega(E(\mathbf{C}))}{\omega(E)} = \frac{\sum_{e \in E(\mathbf{C})} \omega(e)}{\sum_{e \in E} \omega(e)}$	$[0,1]$	Max
Graph Conductance	$\varphi(C) = \begin{cases} 1, & C \in \{\emptyset, V\} \\ 0, & C \notin \{\emptyset, V\} \wedge \omega(\mathbf{C}) = 0 \\ \frac{\omega(\mathbf{C})}{\min(\alpha(C), \alpha(\overline{C}))}, & otherwise \end{cases}$, $\varphi(G) = \min_{C \subset V} \varphi(C)$	$[0,1]$	Max
Intra-cluster Conductance	$\alpha(\mathbf{C}) = \min_{i \in \{1,\ldots,k\}} \varphi(G(C_i))$	$[0,1]$	Max
Inter-cluster Conductance	$\delta(\mathbf{C}) = \begin{cases} 1, & if\ \mathbf{C} = \{V\} \\ 1 - \max_{i \in \{1,\ldots,k\}} \varphi(C_i), & otherwise \end{cases}$	$[0,1]$	Min
Performance	$perf(\mathbf{C}) = \frac{\sum_{i=1}^{k} \lvert E(C_i) \rvert + \sum_{u,v \in V}^{k} [(u,v) \notin E] I_{i,j}(u,v)}{\frac{1}{2} m(m-1)}$	$[0,1]$	Max
Modularity	$Q(\mathbf{C}) = \frac{\omega(E(\mathbf{C})) - \sum_{i=1}^{k} \left[\frac{\left(\sum_{e \in E(C_i, V)} \omega(e) \right)^2}{\omega(E)} \right]}{\omega(E)}$	$[-1,1]$	Max
Fuzzy modularity	$Q_f = \frac{1}{2m} \sum_{i=1}^{k} \sum_{u,v \in C_i} \left(\frac{\rho_i(u) + \rho_i(v)}{2} e(u,v) - \frac{d_f(u) d_f(v)}{2m} I(u,v) \right)$	$[-1,1]$	Max

Table 3.1: Connectivity based quality indices summary

3.3.3 METRIC BASED QUALITY INDEXES

In the previous section we presented some connectivity based indices. These indices use the structure of the graph to measure the quality. In this section we present other kind of measures that use the notion of distance between nodes. The distance is used, like vectors in data clustering, to determine whether one node belongs or not to some cluster. For graph clustering, the distance between a pair of nodes could be measured in terms of the similarity between nodes or with the weight of each link.

Let $d_{i,j} = \|v_i - v_j\|$ be the distance measure between nodes v_i and v_j, and let Z be the centroid of the whole node set, given by

$$Z = \{v_k : d_{i,k} = \min d_{i,j} \forall i, j \in E\} \tag{3.35}$$

Z is the node that minimizes the distance to all the nodes in the graph. Similarly, Z_{C_i} is the centroid of the cluster C_i, is the node that minimizes the distance to all the nodes that belong to C_i.

- **Davies – Bouldin index:** this index measures the average ratio between intra-cluster scatter, i.e., the average distance of all nodes to the centroid of the cluster, and inter-cluster separation. This index is defined by [Foggia et al., 2009] as:

$$DB = \frac{1}{|\mathbf{C}|} \sum \max_{i \neq j} \frac{A_i + A_j}{d_{ij}} \tag{3.36}$$

  where $\mathbf{C}$ is a partition of the graph, $i, j \in \mathbf{C}$, A_i is the average distance of members of cluster i to its centroid and d_{ij} is the distance between the centroid i and the centroid j. This measure will assign values between 0 and 1, 0 being the best value. However, in the trivial case when each cluster has one node, the index obtains their minimum value.

- **Dunn index:** this index calculates the ratio between the maximum intra-cluster distance and the minimum inter-cluster distance. In general, the higher the index the better the clustering; this can be achieved by a large minimum inter-cluster distance and/or a small maximum intra-cluster distance. It is defined by [Günter and Bunke, 2003] as:

$$D = \frac{d_{\min}}{d_{\max}} \tag{3.37}$$

  where $d_{\min}$ is the minimum inter-cluster distance, and $d_{\max}$ is the maximum intra-cluster distance. This could lead to indeterminate results when each node represents one cluster and $d_{\max} = 0$.

- **Calinski – Harabasz:** this index is based on the trace of inter and intra-cluster distances [Maulik and Bandyopadhyay, 2002, Foggia et al., 2009]. The trace of the inter-cluster distance is defined as:

$$T_B = \sum_{i=1}^{k} |C_i| \left\| Z_{C_i} - Z \right\|^2$$

where k is the number of clusters, $|C_i|$ is the number of points in the $i-$th cluster, Z_{C_i} is the centroid of the $i-$th cluster and Z is the centroid of the whole set.

The trace of the intra-cluster distance is defined by:

$$T_W = \sum_{i=1}^{k} \sum_{v \in C_i}^{|C_i|} \left\| v - Z_{C_i} \right\|^2$$

Thus, the CH index can be defined as:

$$CH = \frac{T_B / (|\mathbf{C}|-1)}{T_W / (n-|\mathbf{C}|)} \tag{3.38}$$

where n is the total number of nodes. This index could lead to undesirable results when number of clusters is equal to the number of nodes or when there is only one cluster.

- **Xie – Beni index:** is used to measure the quality of fuzzy clusters. This uses the notion of compactness and separation [Xie and Beni, 1991, Foggia et al., 2009] of a determined partition. This index is defined by [Maulik and Bandyopadhyay, 2002] as:

$$XB = \frac{\sum_{i=1}^{k} \sum_{v \in V} S_i(v)^2 \left\| v - Z_{C_i} \right\|^2}{n \min_{i,j} \left\| Z_{C_i} - Z_{C_j} \right\|^2} \tag{3.39}$$

where n is the number of nodes, $S_i(v)$ is the membership function of the node v to the cluster i, and $\left\| Z_{C_i} - Z_{C_j} \right\|$ is the distance between the centroids of clusters i and j. This index leads to undesirable results when, whether each node belongs to a singleton cluster or there is only one cluster with all the nodes in it.

- **C index:** this index is based on the computation of three distance measures as presented by [Hubert and Schultz, 1976]. It is defined as:

$$C = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{3.40}$$

where S is the sum of distances over all pairs of nodes from the same cluster of cardinality m. S_{min} is the sum of the m minimum distances over all the node pairs and S_{max} the sum of m maximum distances over all the mode pairs. This index provides small values for good clusters, however it is appropriate if the clusters are of similar size [Günter and Bunke, 2003] and, for trivial situations like each node belongs to a singleton cluster or when all the nodes belong to a giant cluster, the index is indeterminate. This index was created for evaluating data sets of elements represented as vectors, some authors like [Foggia et al., 2009] have introduced the idea of geometric distance to be used with graphs.

- $\mathcal{I}$ **index:** this index takes into account three features of a graph: the number of clusters, the compactness of the clusters and their separation [Foggia et al., 2009]. This index is intended to measure the quality over fuzzy clusters. It is defined by:

$$\mathcal{I} = \left(\frac{1}{k} \times \frac{P_1}{P_k} \times D_k\right)^2 \tag{3.41}$$

where $|C_i| = k$ is the number of clusters, the compactness:

$$P_k = \sum_{i=1}^{k} \sum_{v \in V} S_i\left(v\right)^2 \left\|v - Z_{C_i}\right\|^2$$

and the separation between clusters:

$$D_k = \max_{i,j=1}^{k} \left\|Z_{C_i} - Z_{C_j}\right\|^2$$

With the first factor, the index decreases as k is increased. The second and third factors increase the index as k is increased. This index will assign high values for good clusterings, however reaches an indetermination when all the nodes belong to a single giant cluster.

Table 3.2 shows a summary of the distance based quality indices presented in this section.

Index Name	Expression	Limits	Direction	Indetermination
Davies – Boulin	$DB = \frac{1}{\lvert\mathbf{C}\rvert} \sum \max_{i \neq j} \frac{A_i + A_j}{d_{ij}}$	$[0, 1]$	Min	–
Dunn	$D = \frac{d_{\min}}{d_{\max}}$	$[0, 1]$	Max	Each node is a cluster $\therefore d_{max} = 0$
Calinski – Harabasz	$CH = \frac{\sum_{i=1}^{k} \lvert C_i \rvert \left\lVert z_{C_i} - z \right\rVert^2 / (\lvert\mathbf{C}\rvert - 1)}{\sum_{i=1}^{k} \sum_{v \in C_i}^{\lvert C_i \rvert} \left\lVert v - z_{C_i} \right\rVert^2 / (m - \lvert\mathbf{C}\rvert)}$	$[0, 1]$	Max	Each node is a cluster, or all the nodes belong to one cluster
Xie – Beni	$XB = \frac{\sum_{i=1}^{k} \sum_{v \in V}^{m} S_i(v)^2 \left\lVert v - Z_{C_i} \right\rVert^2}{m \min_{i,j} \left\lVert Z_{C_i} - Z_{C_j} \right\rVert^2}$	$[0, 1]$	Min	All the nodes belong to one cluster
C	$C = \frac{S - S_{min}}{S_{max} - S_{min}}$	$[0, 1]$	Min	Each node is a cluster, or all the nodes belong to one cluster
$\mathcal{I}$	$\mathcal{I} = \left(\frac{1}{k} \times \frac{P_1 \times \max_{i,j=1}^{k} \left\lVert Z_{C_i} - Z_{C_j} \right\rVert^2}{\sum_{i=1}^{k} \sum_{v \in V}^{m} S_i(v)^2 \left\lVert v - Z_{C_i} \right\rVert^2} \right)^2$	$[0, 1]$	Max	Each node is a cluster

Table 3.2: Distances based quality indices summary

3.3.4 GRAPH CLUSTERING

Community detection is the process of grouping nodes according to different criteria, the ones based on a distance, like in data clustering, or the ones based on the structure, using the number of edges within a group versus the number of edges between the groups. However, the community detection process is not well defined due to some ambiguities in the definitions of partition and community and the quality measures [Fortunato, 2010].

One of the most important assumptions in community detection is the sparsity of the graph [Fortunato, 2010], which means the number of edges m is in the order of the number of nodes n. This assumption is due to the sparse nature of social networks [Wasserman and Faust, 1994].

3.3.4.1 GRAPH PARTITION METHODS

Graph partition methods divide the nodes set of a graph into a predefined number k of groups. The groups are calculated in such a way the number of edges between them is minimal, i.e., the minimum cut size. Several methods have been developed to find graphs partitions using these techniques, specially for circuit designing [Fortunato, 2010, Kernighan and Lin, 1970] to minimize the number of connections between boards or the identification of Web communities [Flake et al., 2000] among others.

In the work presented by [Kernighan and Lin, 1970], authors have proposed a heuristic for partitioning arbitrary graphs into k groups of size p. One of the goals of the method is to keep the uniformity of the partition, i.e., to have groups of similar size. This impose a set of restrictions in terms of the final solution, causing that other approaches, like linear programming, the Ford–Fulkerson algorithm [Ford and Fulkerson, 1956] and graph clustering techniques, are not suitable to solve this particular problem. In the first case, the problem can be stated as a linear programming problem, however, according to the authors, it needs a large number of constraints to assure the uniformity of the partition. In the second case, the Ford–Fulkerson algorithm finds the maximal flow in a network with the minimal cut. Despite the correctness of the method in dividing the nodes set, it does not assure equally, or at least similarly, sized groups. Finally, in the third case, graph clustering approaches are not suitable because of the way the groups are constructed: the main idea of graph clustering is to find groups of strongly connected nodes according to some criteria, whether structural or external, and not to taking into account the size of the groups.

[Barnes, 1981] presents a heuristic partitioning algorithm based on the spectral properties of the Laplacian matrix A of the graph G. First, the author defines k column vectors $\mathbf{x}_j,\ j = 1, \ldots, k$ representing each partition, i.e., $\mathbf{x}_{ij} = 1$ if node i belongs to

the group j, $\mathbf{x}_{ij} = 0$ otherwise. Additionally, each column vector satisfies the following constraints:

$$\sum_{i=1}^{n} x_{ij} = n_j, j = 1, \ldots, k \tag{3.42}$$

where n_j is the number of nodes in the group j and n is the number of nodes, and

$$\sum_{j=1}^{k} x_{ij} = 1, i = 1, \ldots, n \tag{3.43}$$

where n is the number of nodes. This constraint says that each node in G belongs only to one group. Finally, given the k eigenvectors $\mathbf{u}_j$ from the k largest eigenvalues, the author describes the problem as equivalent to solving a transportation problem minimizing:

$$\min \sum_{j=1}^{k} \sum_{i=1}^{n} \frac{u_{ij}}{\sqrt{n_j}} x_{ij} \tag{3.44}$$

subject to constraints 3.42 and 3.43.

In general these methods require to define beforehand the number of groups in order to calculate the minimum cut. Additionally, according to [Fortunato, 2010], these methods are not suitable for community detection because of the restrictions imposed by the iterative splitting of the graph, which may lead to groups formed of parts from past divisions, which is not reliable.

3.3.4.2 HIERARCHICAL CLUSTERING

These methods were conceived to cluster data, as presented in Section 3.2, however it is possible to use them for graph clustering. As in data clustering, there are two categories for hierarchical clustering techniques:

- **Agglomerative algorithms:** assigning first each node to one cluster, each cluster is merged according to some similarity measure i.e., if the similarity between the two clusters is high enough.

- **Divisive algorithms:** having all the nodes into a single group, the algorithm searches and removes those edges below some similarity threshold. This leaves groups with a high similarity.

Let us note that both cases need the definition of a (dis)similarity measure to decide whether merge or split a pair of clusters, however, the notion of node (dis)similarity is not well defined. One way to measure the dissimilarity between two nodes is by measuring the geodesic distance, which according to [Wasserman and Faust, 1994] and [Rattigan et al., 2007] is one of the most used in the field, because it allows introducing naturally a distance notion in a graph. This distance represents the shortest path between the nodes u and v as: $d(u,v) = \min_p A^{[p]}_{uv} > 0$ where $A^{[p]}$ is the power matrix p.

However, this measure does not provide an idea about the neighborhood of the nodes. [Fortunato, 2010] presents the Jaccard distance 3.2, which is transformed to use the neighbors of each node, i.e., this measures the overlapping between the neighborhood of two nodes u and v. Thus, if the neighborhoods of two nodes u and v contain the same elements, the distance will be 0, i.e., the nodes will be totally similar in terms of their local structure. [Thiel and Berthold, 2010] propose also two node similarity measures based on spreading activation processes. These measures are based on neighborhood overlap and the activation patterns of each node.

According to [Fortunato, 2010] and [Hastie et al., 2009], hierarchical approaches for graph clustering have several advantages such that previous knowledge about the number of groups is not needed and they may reveal the hierarchical structure of the graph, if it has one. On the other hand, these methods have some drawbacks: if they exists, the nodes with one neighbor are assigned to separated clusters and the methods do not say what is the best number of groups [Newman, 2004a].

Additionally, since there is not a natural way to calculate the (dis)similarity between two nodes, the typical approach is to sort in decreasing order the $O(n^2)$ (dis)similarities which is on the order of $O(n^2 \log n)$. Since the construction of the dendrogram can be performed in linear time [Newman, 2004a], the whole complexity of these techniques is bounded by $O(n^2 \log n)$ in sparse graphs [Newman, 2004a].

3.3.4.3 PARTITIONAL CLUSTERING

Partitional clustering methods for graphs use the same principle as those presented in Section 3.2 for data clustering. In the case of graphs, the difference lies on the way how the centroids and the distance of a node to them are calculated. [Rattigan et al., 2007] propose a node indexing, which is a set of annotations combined with a distance measure based on the notion of distance to zone. This indexation process reduces the calculation time of $k-$medoids algorithm according to [Rattigan et al., 2007].

An algorithm to detect communities in large–scale social networks is presented by [Du et al., 2007]. Their method is based on the enumeration of all the maximal cliques, i.e., a complete subgraph which is not contained in any other complete subgraph. After

all the maximal cliques are enumerated, they generate different, so called kernels, which are the "centers" of the communities; then, each node of the graph will be assigned to one of the kernels according to its distance to the center in $k-$means fashioned way. Finally, they try to optimize the modularity obtained by moving nodes from one community to another. This algorithm has a complexity of $O(n^2)$, where n is the number of nodes.

Except for the maximal cliques enumeration algorithm, the main inconvenience of these techniques is again the need to define in advance the number of groups of the partition [Fortunato, 2010], which in community detection is not the general case. Moreover, including a node distance notion is not natural in graphs.

3.3.4.4 SPECTRAL METHODS

Spectral clustering methods are based on the definition of pair-wise dissimilarity, or distance matrices. The dissimilarities are measured using some of the measures presented in Section 3.2, which since they follow the four discussed properties, they produce positive-definite matrices, used during the clustering process.

However, for those measures, the data must be represented in suitable way for arithmetic and algebraic operations, i.e., to make possible the use of arithmetic and set algebra operations. In the case of graphs it means, to produce a symmetric, positive-definite dissimilarity matrix Δ [Fortunato, 2010]. First, the data from the graph has to be transformed into another representation, then, the matrix is used to extract coordinates from the points, generally using their eigen-vectors and eigen values, and finally using some classic clustering algorithm such as $k-$means.

[Yen et al., 2009] propose an algorithm for nodes clustering using a spectral approach. They define a two-step process: first, define a similarity or kernel matrix to capture the similarity between nodes, then use a kernel clustering algorithm to find groups of nodes according to the precedent similarity matrix.

To calculate the similarity they propose the use of the sigmoid commute-time kernel, based on the average commute distance $\mathbf{CT}$, defined also in [Luxburg, 2007], which is defined as the average number of steps a random walker takes to go from a node i to a node j, with $i \neq j$. Once the similarity matrix is defined using $\mathbf{CT}$, the authors transform the kernel applying a sigmoid transformation. This transformation reduces the number of outliers, however, may produce a non positive semi-definite matrices, yielding to eigen-vectors and eigen-values in the complex plane.

After this kernel is constructed, the authors use $k-$means and fuzzy $c-$means algorithms to cluster the nodes.

[Luxburg, 2007] presents a tutorial on spectral clustering in which he discuses different approaches for the creation of the kernels, using several distances and similarity

metrics. He additionally addresses the problem of choosing k, the number of groups to be used during the second step of the clustering. He proposes a heuristic called *eigengap*. This heuristic measures the distances between the l smallest eigen-values and select those with a large difference. This heuristic works fine with graphs with a well defined community structure such as social networks, but in other cases, the difference between eigen-values becomes blurry.

3.3.4.5 DIVISIVE ALGORITHMS

These algorithms are designed to separate clusters by removing edges from the graph. The selected edges to be removed are chosen according to one, appropriate, property of similarity. Compared to hierarchical clustering, the removed edges are not selected using some similarity measure between nodes, but using a measure of the importance of the edge in connecting communities.

The most used measure for divisive algorithms is the *edge centrality* [Fortunato, 2010], which measures how important is a given edge to some process running on the graph.

This notion was used by [Girvan and Newman, 2002, Newman and Girvan, 2004] when they proposed their algorithm. This algorithm is focused on a generalization of the betweenness measure proposed by [Freeman, 1977]: the betweenness of the edge, which counts the times the edge participates on a process. The general steps of the algorithm are:

1. Compute the betweenness centrality of all edges
2. Remove the edge with the highest centrality
3. Recalculate the centrality of the edges
4. Go to step 2 until no edge remains

The idea then, is to find those high-central edges, connecting clusters of nodes, and by removing them, discover the underlying network structure of the graph.

Searching the most central edge has a complexity of $O(mn)$, where m is the number of edges and n the number of nodes, but since the centrality has to be calculated in each iteration, the worst case is $O(m^2n)$. However, in sparse graphs, e.g., social networks, the complexity can be reduced to $O(n^2)$ according to [Fortunato, 2010].

[Tyler et al., 2003] propose a modification of the Girvan and Newman's algorithm to test if a group of nodes fall into the same community in different iterations. Additionally, they use the method proposed by [Brandes, 2001] to find the betweenness centrality

faster than the Newman's version. This method, which is similar to that the one proposed in [Newman, 2001] allows to calculate the betweenness centrality in $O(mn)$ for unweighted graphs and $O\left(mn + \log^2 n\right)$ for weighted graphs.

The algorithm proposed by [Newman, 2001] iteratively finds and removes the edge with the highest betweenness score. This process allows finding groups which are loosely connected with each other and with well connected nodes within the group. The partition with the highest modularity gives the best partition. The main drawback of this approach is the complexity of the calculation of the betweenness, the general algorithm will take $O(mn^2)$ for m edges and n nodes, its cost for huge graphs is prohibitive.

In general, these methods use an external stop criterion, such as connectivity based measures like coverage, conductance, performance and modularity (Section 3.3.2.)

3.3.4.6 MODULARITY BASED METHODS

As presented in Section 3.3.4.5, the modularity, introduced by [Newman and Girvan, 2004], was used as a stopping criterion for divisive algorithms, however, other methods may used this measure as an element to affect the execution of the algorithm, i.e., as an objective function in an optimization problem.

[Newman, 2004b] proposes an approach, using the community detection algorithm presented by [Newman and Girvan, 2004] and the modularity Q as a measure of the quality of the partition: he constructs the partition using his divisive algorithm and besides the partition, an accompanying dendrogram representing the hierarchy of the clusters. Thus, by moving down through the dendrogram, the author searches for local high peaks of modularity. Those high modularity values represent good node partitions. The complexity of this algorithm for sparse graphs is $O(n^2)$ where n is the number of nodes.

[Clauset et al., 2004] continue under the idea of optimizing the modularity. In their algorithm, they change some implementation details and procedures of the Newman and Girvan's algorithm by assuming the sparseness of the graph. Thus, the algorithm begins by storing only the variation ΔQ of the modularity for each row of the adjacency matrix of the graph. The general steps of the algorithm are:

1. Calculate the values of ΔQ_{ij} (The Q variation for joining communities i and j.)
2. Join the communities i and j maximizing ΔQ_{ij} and increment Q by ΔQ_{ij}.
3. Repeat step 2 until only one community remains.

The general complexity of this algorithm is $O\left(n \log^2 n\right)$. This complexity was calculated assuming that the graph is sparse, therefore a high number of modularity variations

were not taken into account. A variation of the Clauset's algorithm is proposed in by [Wakita and Tsurumi, 2007]. In this algorithm authors inherit the same idea but selecting as ΔQ_{ij} communities with similar sizes, reducing the consolidation time, which was identified by authors as the bottleneck of the algorithm.

[Blondel et al., 2008] have used another approach where the modularity is computed iteratively. They propose an algorithm executed in two stages:

1. Find a partition of the nodes.
2. Create a graph of meta-nodes and meta-edges from the groups found at step 1.
3. Repeat step 1 until no modularity improvements can be made.

During the first step, the algorithm will put each node in one community, and then, a random node is selected and put in another random community. This is repeated until all the communities have been tested; for each change the variation of the modularity is tested, and the node will be put in the community which has the highest positive variation, or is returned to its original community if no better/positive change is possible.

At the end of this step, each node belongs to one community, and this partition has the maximum possible modularity. Then, each community is transformed to a meta node: meta-edges are composed by the edges between communities.

With this new meta-graph the process is repeated. This process assures that the modularity value is monotonically increasing. On the complexity side, the very first step of the algorithm is the most time consuming: the process deals with the complete graph, but once the communities are first detected, the graph is collapsed into meta-graphs with less elements. Authors claim the complexity is $O(m)$, where m is the number of edges, for sparse graphs.

In section 3.3.2 a fuzzy modularity index was presented. Using this index, [Liu, 2010] presents an energy based algorithm to calculate fuzzy partitions maximizing Q_f. This algorithm initializes the energy of the system to $E = -Q_f$, then using an optimization approach the energy E is minimized. This minimization implies the maximization of Q_f. The algorithm starts selecting a random number of groups $N\prime \backsim U[N_{min}, N_{max}]$. Then, the nodes memberships to each of the $N\prime$ groups is randomly assigned. The algorithm iterates changing the memberships values of each node until a threshold temperature is achieved.

Despite the quality of the partitions obtained using the modularity as a quality function, it presents some limitations regarding the optimal partition of a graph. These limitations make the produced partition to be difficult to analyze. A good study and discussion about the limitations of the modularity as quality measure for graph clustering is presented in [Good et al., 2010].

3.3.4.7 OTHER METHODS

Approaches using genetic algorithms were proposed by [Pizzuti, 2008a] and [Pizzuti, 2008b]. These algorithms are designed to optimize a fitness function based on the volume of edges within each group.

As usual in genetic algorithms approaches, three operators were defined: selection, cross-over and mutation. The first one is done using a roulette mechanism, the second is done by simple interchanging genes from two parents, producing child per offspring; crossover is executed with a probability p. Finally, the mutation, having a probability $q = 1 - p$, is used to create valid individuals in the context of the problem.

Another work from [Pizzuti, 2009] is a genetic algorithm to identify overlapping communities in a graph. The design of the genetic algorithm is the same as in [Pizzuti, 2008b], but instead of using the graph, she proposes the use of line graph $L(G)$, which is a graph such that each node of $L(G)$ represents an edge of G and an edge in $L(G)$ represents two edges in G with a common endpoint.

Thus, using $L(G)$ as individuals the algorithm detect overlapping communities, but the fitness function is still calculated over the original graph G.

An agglomerative genetic algorithm approach is proposed by [Lipczak and Milios, 2009]. In this case, the individuals are represented as a string of groups, which is a vector of size n containing the number of items in each of the k groups. During the selection and the crossover operations, the genes are selected according to the potential improvement of the fitness function they may give.

The fitness function, in fact, is composed of three measures, the normalized cut, proposed by [Shi and Malik, 2000], the Newman's modularity [Newman and Girvan, 2004], and the silhouette width, proposed by [Rousseeuw, 1987]. In general, the complexity of this method is in the order of $O(n^2 \times \lambda)$, where n is the number of nodes and λ the number of iterations of the simulation.

3.3.4.8 METHODS USING ADDITIONAL INFORMATION

In general, previous works use only the structure of the graph to detect the communities. This information is mostly used to find groups of nodes strongly connected. However, there are other approaches which combine that structural information with information associated to the nodes of the graph. Nodes represents persons or organizations and the edges represent interactions. Generally, as claimed in our current work, a social networks contains additional data, like interests or associated topics; this additional information can be used to create groups. As far as we know, this data is not yet fully exploited. We list here few works in this direction while writing this thesis.

[Pathak et al., 2008] propose a method which uses the information contained inside the edges of the network, treated as topics of interest shared between people. Authors claim that classic graph clustering methods find groups of well connected nodes but related to very dissimilar topics. Their method is the construction of a Bayesian model to infer the communication model; the assumption made is that two persons in the same group talk about topics interesting both of them but also their community. The model generates for each node a degree of membership to a community or communities and can generate fuzzy partitions.

[Sachan et al., 2012] present a method based on that presented by [Pathak et al., 2008]. The main difference is the addition of the topological aspect of the graph. Thus, the algorithm generates partitions in which nodes may belong to several groups and each group may contain different topics. To measure the quality of the partition authors use the fuzzy modularity Q_f.

[Yang et al., 2009] present a method to combine links and content for community detection in directed graphs. Authors define a link model based on the popularity of the nodes, creating a model in which the edge weight is defined by the probability of the node u to cite the node v. The probability model is changed to include the content through the definition of a vector of membership to each element of the content. This method calculates the probability of each pair of nodes to be connected and not only those which are connected by an edge.

[Zhou et al., 2009] propose a community detection approach using structural and attribute similarities. They use a random walk throughout a predefined set of k clusters, and try to maximize the distance between clusters by moving nodes according to their similarity. First, they create an augmented graph from the node attributes, then they execute the random walk over the transition matrix generated by the augmented graph. This leads them to find k groups of semantically close nodes. To measure the clustering from a structural point of view, they use the density of edges within the clusters.

3.3.5 CLUSTERING ANALYSIS AND EVALUATION

In general graph clustering methods produce different results for the same data set. It comes from the way the data is represented and manipulated by each approach. Additionally, algorithms based on modularity optimization (Section 3.3.4.6) and implementing genetic algorithms (Section 3.3.4.7) may produce different partition configuration in different executions.

There are different studies to measure and reduce these discrepancies between algorithms and executions. One of the first works in analyzing and comparing clusters is the one presented by [Rand, 1971].

In this work, Rand states the problem of clustering evaluation. Given two partitions C_1 and C_2 of the same data set, how similar those partitions are? The Rand index $c\left(C_1, C_2\right)$ is given by:

$$c\left(C_1, C_2\right) = \sum_{i<j}^{n} \frac{\gamma_{ij}}{\binom{n}{2}} \tag{3.45}$$

where γ_{ij} is 1 if the elements i and j are either placed in the same group in both partitions C_1 and C_2 or if they are placed in different groups in both partitions, i.e., there is an agreement in both partitions; γ_{ij} is 0 in other case.

The agreements between two partitions C_1 and C_2 can be summarized as a contingency table in which each element $n_{ij} = |C_{1i} \cap C_{2j}|$, i.e., the number of common elements of groups i and j from partitions C_1 and C_2 respectively. Table 3.3 presents a general example of a contingency table.

		Paritition C_2				
	Class	v_1	v_2	$\dots$	v_s	Sums
Partition C_1	u_1	n_{11}	n_{12}	$\dots$	n_{1s}	$n_{1\cdot}$
	u_2	n_{21}	n_{22}	$\dots$	n_{2s}	$n_{2\cdot}$
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	u_r	n_{r1}	n_{r2}	$\dots$	n_{rs}	$n_{r\cdot}$
	Sums	$n_{\cdot 1}$	$n_{\cdot 2}$	$\dots$	$n_{\cdot s}$	n

Table 3.3: Contingency table of the agreements of two partitions C_1 and C_2

Thus, using the previous table, the rand index RI can be defined as:

$$RI = \frac{\binom{n}{2} + \sum_{i=1}^{r}\sum_{j=1}^{s} n_{ij}^2 - \frac{1}{2}\left(\sum_{i=1}^{r} n_{i\cdot}^2 + \sum_{j=1}^{s} n_{\cdot j}^2\right)}{\binom{n}{2}} \tag{3.46}$$

However, this index has some problems. According to [Hubert and Arabie, 1985] this index is not corrected for chance, i.e., the index does not take into account how the partitions were chosen.

This makes the index values to be not normalized to lie between some bounds, making it difficult to compare two results. To overcome this issue, in [Hubert and Arabie, 1985] authors propose a general solution to adjust comparison indices:

$$\text{Adjusted Index} = \frac{\text{Index} - \text{Expected Index}}{\text{Maximum Index} - \text{Expected Index}} \tag{3.47}$$

Thus, using the equation 3.47, the equation 3.46 can be rewritten as:

$$ARI = \frac{\sum_{i=1}^{r}\sum_{j=1}^{s}\binom{n_{ij}}{2} - \left[\sum_{i=1}^{r}\binom{n_{i\cdot}}{2}\sum_{j=1}^{s}\binom{n_{\cdot j}}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_{i=1}^{r}\binom{n_{i\cdot}}{2} + \sum_{j=1}^{s}\binom{n_{\cdot j}}{2}\right] - \left[\sum_{i=1}^{r}\binom{n_{i\cdot}}{2}\sum_{j=1}^{s}\binom{n_{\cdot j}}{2}\right]/\binom{n}{2}} \tag{3.48}$$

which is the Adjusted Rand Index ARIpresented in [Hubert and Arabie, 1985]. This index can take values between 0 and 1.

These indices serve to compare and study the behavior of clustering methods, for example as presented by [Rand, 1971], it is possible to measure how sensitive is a method to missing data, to noise and how different are the results of two methods.

[Kwak et al., 2009], present a study on graphs partition evaluation. This is similar to use a partition similarity index such as Rand Index, but different in the sense of the application: here authors want to have consistent partitions of the same data in different runs. Thus, they evaluate three modularity based algorithms (Section 3.3.4.6):

- Newman's and its variances
- Wakita's
- Blondel's

They define the following index of consistency:

$$\mathcal{C} = \frac{\sum_{(i,j)\in E}(p_{ij} - 0.5)^2}{|E|} \times \frac{1}{0.5^2} \tag{3.49}$$

where p_{ij} is the probability of nodes i and j to be placed in the same community across multiple executions of the algorithm.

Using this index, authors have observed that certain partition configurations may give better results in terms of modularity. By calculating the best partitions over N runs, they change the weights of those edges which give a better consistency index, by this way, clustering algorithms will privilege these winning configurations.

3.4 LAYOUT OF GRAPHS AND OF GRAPHS OF COMMUNITIES

Graph layout refers to a set of techniques designed to represent nodes and edges entities within a $2D$ or $3D$, usually limited, space. In general, the graph layout techniques are created according to different objectives such as those summarized by [Di Battista et al., 1994] and by [Purchase et al., 1997]:

- Display symmetry
- Reduce edge crossings
- Reduce edge bending
- Keep edge lengths uniform
- Distribute vertices uniformly
- Keep edges parallel to coordinates axes
- Maximize the angle between two consecutive edges incident to a node.

These objectives help to improve the readability and interpretation of the graph, however, [Di Battista et al., 1994, Garey and Johnson, 1983] have reported these criteria to be NP-hard optimization problems. Additionally to complexity issues, these aesthetic objectives can not be optimized at the same time: optimizing one may be detrimental for the other objectives [Di Battista et al., 1994].

Regardless specific aesthetic objectives, graph layout algorithms try to produce visual representations which can be analyzed and in general, that are well suited to a particular problem, for example, layout of UML diagrams [Purchase et al., 2002], visualization and analysis of large semantic graphs [Wong et al., 2006] or visual analysis for discovering information in organizations [Perer et al., 2011] among others.

Clustered graphs layout, on the other hand, are a set of methods designed to extract and use the groups structure given. Although, in some cases the layout algorithm have been conceived to find a partition of the graph at the same time it is being laid out. The main idea is to represent the differences between groups and in some cases to reduce the visual complexity of the drawing.

3.4.1 FORCE DIRECTED METHODS

One of the typical approaches to layout graphs is modeling nodes and edges as a physical system which needs to be stabilized.

One important work in force directed methods was presented by [Eades, 1984]. In this work, the author modeled the graph as a spring-mass system: each node is replaced by a steel ring and each edge by a steel spring. Then, given some initial configuration, the spring-mass system is released and the spring forces take it to a minimal energy state, however, the spring forces do not follow the Hooke's law.

A version of the Eades' algorithm is proposed by [Kamada and Kawai, 1989]. In this work, authors model the system following the Hooke's law and solving a set of partial derivatives to reduce the energy of the system. Additionally, contrary to Eades, authors model an optimal distance between nodes which are not neighbors , which is proportional to their geodesic distance.

Another typical algorithm was proposed by [Fruchterman and Reingold, 1991]. This algorithm has been designed to draw neighbor nodes close to each other but keeping a minimum distance between them. This means, the layout should take advantage of the available drawing area. To do this, the system is modeled using an analogy of subatomic particles or planetary configurations: nodes connected by an edge will be attracted but, all nodes repel each other. The stopping criterion is calculated from a cooling function based on the movement of the nodes after each iteration.

3.4.2 HYPERBOLIC LAYOUT

This technique has been developed to draw trees and graphs in a hyperbolic space. Embedding a graph into a hyperbolic space has two main advantages [Munzner, 1998]: exponential room for drawing and outsider's view. Mathematical foundations of hyperbolic spaces and stereographic projections can be found by [Anderson, 2005].

According to [Herman et al., 2000] first papers in hyperbolic layout were presented by [Lamping et al., 1995] for drawing hierarchical structures using a focus+context method. They use a Poincare's disk to represent the visible, or focused, hyperbolic space.

In the work proposed by [Munzner, 1998], the author proposes a 3D hyperbolic approach to browse large graphs. The graph hence is drawn on the surface of a sphere, maximizing the use of the space.

In the work proposed by [Mohar, 1999], the author proposes some primitives for drawing graphs in a hyperbolic space. These primitives allows to simplify the drawing process using directly the geometrical properties of the space rather than just translate coordinates from a Cartesian space.

3.4.3 EDGE CLUTTER REDUCTION

Most graph clustering methods deal with how to place nodes and use neighborhood notions, as in the force directed methods, to take into account the edges. However, when drawing edges some issues may arise, for example, given a large graph, the edges may cover the drawing up making virtually impossible for the user to understand its structure. Some works have been developed to overcome, or at least reduce the impact of this problem.

In the work presented by [Wong et al., 2003] is a method called EdgeLens. This method is designed to curve the edges out of the zone where the user is currently focusing. This is called by the authors *edge congestion reduction*. Thus, the method is inspired from the fisheye visualization, but instead of distorting the whole image, the algorithm change edges outside the zone of interest, leaving the nodes in their original position and with their original size; doing this, the semantics given by the node position is kept.

In [Huang et al., 2005], authors propose a framework to filter, cluster and layout large graphs. Filtering is performed by removing noisy nodes according to an importance score, obtained from the number of appearances of a node in different communication paths; thus, if a node is near to isolated nodes, its importance score is low. Then, they cluster the graph, using $k-$means, to obtain summaries for the nodes to display less number of elements, therefore reducing the cluttering. Finally, they produce an interactive view in which the user can choose between different clustering level views to explore the graph.

In [Holten, 2006] an edge clutter reduction technique is presented. This technique reduces edge cluttering by bundling edges of adjacent nodes according to the graph structure, i.e., using the hierarchy of the graph to bend and group the edges. Additionally, the method renders the graph putting the nodes on top and using different opacity levels according to the length of the edges.

In [Ellis and Dix, 2007], authors propose a taxonomy of clutter reduction techniques in information visualization. This taxonomy defines a matrix of clutter reduction techniques and clutter reduction criteria; this matrix thus assess how different methods fulfill, or not, the different visualization criteria. Table 3.4 presents the classification of methods while Table 3.5 presents the criteria for cluttering reduction made by [Ellis and Dix, 2007].

Thus, this taxonomy may help users to match different techniques to solve particular information visualization problems.

In [Henry et al., 2008] a node duplication approach is proposed to improve the readability of graphs and clustered graphs. It is based on NodeTrix [Henry et al., 2007],

Type	Technique
Appareance	Sampling
	Filtering
	Point size
	Opacity
	Clustering
Spatial distortion	Point/line displacement
	Topological distortion
	Space-filling
	Pixel-plotting
	Dimensional rendering
Temporal	Animation

Table 3.4: Cluttering reduction methods

which represents clusters as visual adjacency matrices. Thus, the algorithm is designed to duplicate nodes to represent connections between different communities within a given time frame. Authors propose two ways to duplicate: cloning or splitting. In the first case, the node and all its connections are replicated, causing an increase on the number of edges; in the second case, the node is copied but its connections are divided into the original and the duplicated nodes. This helps also in central actors identification.

3.4.4 LAYOUT OF CLUSTERED GRAPHS

A clustered graph is a special case of hierarchical graphs, where an undirected graph is associated to an inclusion tree representing groups of nodes. A more formal definition is given by [Eades and Feng, 1997]. Thus, authors define a clustered graph $\mathcal{G}$ as a graph $G(V, E)$ and a rooted tree $T(\nu)$ such that the leaves of T are the node set $V \in G$. Additionally, they define the height $h(\nu)$ of a cluster as the depth of the sub-tree ν in the rooted tree, and the span of an edge $\lambda = |\nu_1 - \nu_2|$ in T as the connection between two different levels of the rooted tree. In the case of clustered graphs each level of T is connected only with one level above or below, i.e., each edge has a span of 1.

Figure 3.4 shows an example of a clustered graph and its inclusion tree. In Figure 3.4(a), the nodes are surrounded by ellipses representing groups, and in Figure 3.4(b), each ellipse is represented by a level on the tree. Note that each edge in T has a span λ of 1.

Using this definition, [Eades and Feng, 1997] propose a layout method to exploit this structure in a recursive way using a force directed algorithm and a layered 3D

Technique	Rationale
Avoid overlapping	Ability of identify patterns
Keep spatial information	Position of nodes may have a meaning
Localized	Examine in detail some portion of the data while keeping the context
Scalability	Data set size limitation
Adjustability	Control of visual aspects
Show point/line attributes	Color, size, opacity controlled by the user
Discriminate point/lines	Differentiate between elements in a crowded display
Measure overlapping density	The user are aware that there is data hidden because of the clutter

Table 3.5: Cluttering reduction criteria

visualization. First, the nodes within each community are placed in the first layer of the 3D representation using a force directed algorithm. Then another layer is added above the first one. In this layer a circle representing each community is drawn in the centroid of the nodes placed in the previous step. This operation is repeated throughout all levels in the inclusion tree. Figure 3.5 shows an example of the final representation.

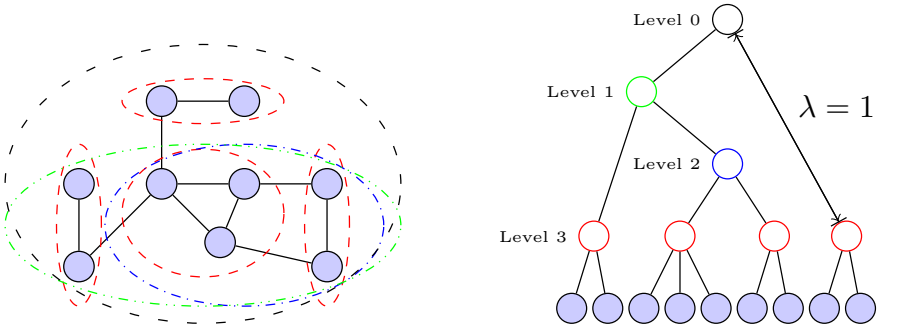
Some works in this field come from the need of layout VLSI[1] circuits. One of them is presented by [Tamassia, 1987], where authors propose an algorithm which uses rectangles to represent nodes and straight lines with right angles to represent links. This kind of layout is used also by [di Giacomo et al., 2007] to present a synthetic view of Web graphs. In this case the rectangles are big enough to contain the nodes from each community, which can be problematic with huge graphs. Figure 3.6 shows an example of this representation.

The LinLog approach is presented in [Noack, 2003]. This is a force directed algorithm for connected graphs, which uses a linear model to represent the attraction of each pair of adjacent nodes and a logarithmic model for representing the repulsion between non connected nodes. The proposed energy model is given by:

$$U_{LinLog}(p) = \sum_{e(u,v)\in E} \|p_u - p_v\| - \sum_{u,v\in V^{(2)}} \ln \|p_u - p_v\|$$

where $e(u,v)$ is the edge between nodes u and v, p_x is the position of the node x and

[1]Very-Large Systems Integration



(a) Example of clustered graph $C(G,T)$

(b) Inclusion tree T derived from a clustered graph

Figure 3.4: Example of a clustered graph and its associated inclusion tree

$\|p_u - p_v\|$ is the Euclidean distance between nodes u and v. This approach does not use a clustered graph but instead, it uses the LinLog model to visually find a partition of the graph.

In [Brockenauer and Cornelsen, 2001], authors define the quotient graph as the result of shrinking groups of nodes into a single node. More formally, given a partition of nodes $\mathbf{C} = (C_1, C_2, \ldots, C_k)$ of a graph $G(V,E)$, the quotient graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ is composed of a set of k meta-nodes $\mathcal{V} = C_1, C_2, \ldots, C_k$ and a set of meta-edges $\mathcal{E} = e(C_i, C_j) : \exists v \in C_i, u \in C_j \wedge e(u,v) \in E$.

Thus, the meta-nodes are composed by each group in the partition, and each meta-edge is the group of edges connecting groups; Figure 3.7 shows an example of a clustered graph and the derived quotient graph.

The clustered graph definition given by [Eades and Feng, 1997] and the notion of quotient graph is used by [Bourqui et al., 2007]. Authors propose a model oriented to deal with weighted clustered graphs. They define four constraints for graph layout algorithms:

1. Forbid overlapping groups to ease the groups' detection.
2. Save the inclusion tree to visualize the hierarchy produced by the clustering algorithm.
3. Use a convex polygon to contain each group.
4. Respect the weighted graph distances using energy minimization functions.

Thus, they begin by placing nodes (meta-nodes) from the level 0, and then, placing the nodes from the subsequent levels. The initial placing occurs by setting each node

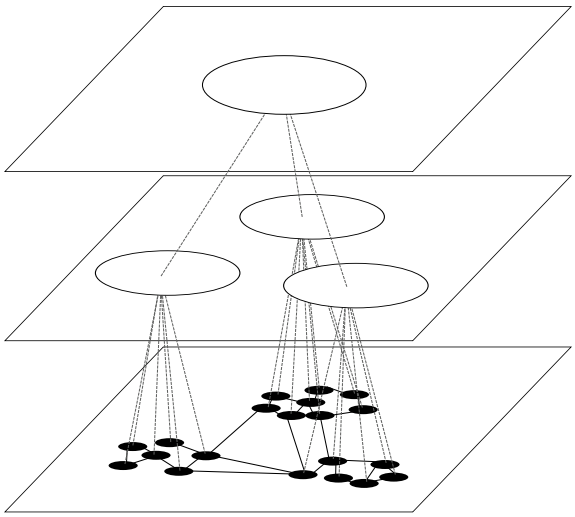


Figure 3.5: Three dimensional layered clustered graph representation. Adapted from [Eades and Feng, 1997].

position such that the distance between them is the same. This is only possible if there are at least three nodes in the level; if there are not enough nodes, they take nodes from the next level to do the operation. After all the levels have been computed, the placement refinement phase begins, using a force directed algorithm. To ensure that the drawn nodes do not overlap, they use Voronoi cells defined by the ancestors of the previous level. Figure 3.8 shows an example of this visualization.

In general, most clustered graphs layout algorithms use general graph layout classic techniques, suited to fit into a hierarchical structure. These methods are not aimed to analyze interactions between communities, therefore, we propose an approach to differentiate nodes according to their role in the community. This approach will be presented in Chapter 5.

Aligned to this perspective [Santamaría and Therón, 2008] present a layout method for overlapping groups in co-authorship networks. This method use a modification of force directed algorithms, modeling the attraction force between connected nodes as a spring and the repulsion force between non-connected nodes as a gravitational force. The difference is that in this algorithm the links between nodes are not shown, drawing instead a convex hull for each group. Since this method has been conceived for overlapping groups, those nodes belonging to several communities will be placed at the border of the group and will be drawn as pie charts, showing the affiliation percentage to each community. Figure 3.9 shows an example of the representation proposed by [Santamaría and Therón, 2008].

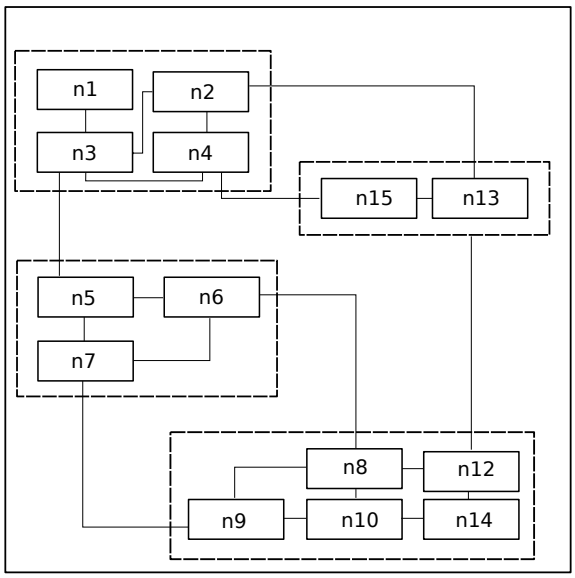


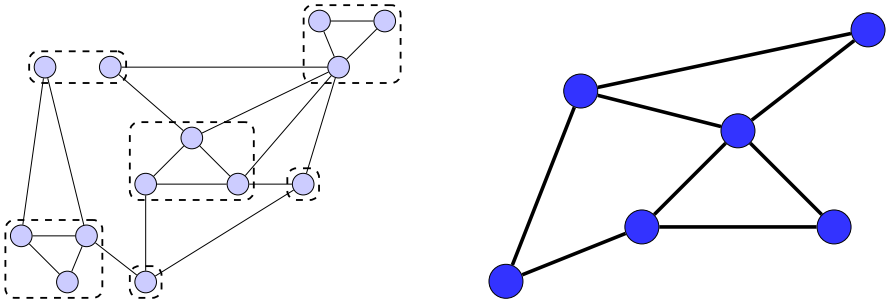
Figure 3.6: Orthogonal representation of clustered graphs. Adapted from [Tamassia, 1987].

In [Mun and Ha, 2005] authors propose an approach to draw hierarchical structures in a radial fashion. The layout is composed by a set of concentric circles, each one representing one level of the hierarchy.

Figure 3.10 presents an example of the radial layout of hierarchical structures. In the figure, non-black nodes represent non-leaf nodes, while black nodes are the leaves of the graph. This representation is oriented to display the inclusion tree rather than the graph itself.

In the work presented by [Batagelj et al., 2011], authors propose a hybrid method to represent large clustered graphs. Their approach consists on combine different layout algorithms for different levels of representation of the graph; authors use first the clustered graph to summarize the information and reduce the amount of elements drawn. For this level they use the algorithm proposed by [Tamassia, 1987], which is well suited for planar graphs. Then, the user can go further into the details of each community by expanding the clusters. The elements into the cluster are displayed as adjacency matrices or using other layout algorithms like circular layout or with the Eades' algorithm.

In [Shi et al., 2009], authors propose a system for visualization and exploration of clustered graphs is proposed. The layout algorithm is a modification of the Kamada-Kawai algorithm [Kamada and Kawai, 1989] in which a term is introduced to take into account the nodes representing groups. This algorithm is applied recursively throughout



(a) Example of graph of node clusters surrounded by rectangles

(b) Quotient graph of the previous clustered graph

Figure 3.7: Example of a clustered graph and a quotient graph. Adapted from [Brockenauer and Cornelsen, 2001]

all levels in the hierarchy.

An algorithm called TopoLayout is presented by [Archambault et al., 2007b]. This algorithm exploits the topological features of the graph to create a group hierarchy and locate the nodes. The algorithm is composed of four general steps:

1. Decomposition: in this phase the topological attributes are identified and the hierarchy is constructed.

2. Feature layout: in this phase each level of the hierarchy is drawn using an algorithm for the specific features being used.

3. Crossing reduction: node-edge and edge-edge reduction (not elimination).

4. Overlap elimination: in this phase nodes a relocated, if it is necessary, to eliminate nodes overlaps in the final drawing.

Thus, this algorithm combines a series of layout and topological features identification techniques to produce a group hierarchy and a final explorable layout.

3.4.5 VISUAL EXPLORATION AND NAVIGATION

Exploration and navigation are important features of information visualization algorithms, specially for visual analytics purposes. These features allow users to study the data using different levels of detail, and even, to change rendering details for a better understanding of the drawing. A good review of overview+detail and focus+context is presented by [Cockburn et al., 2009].

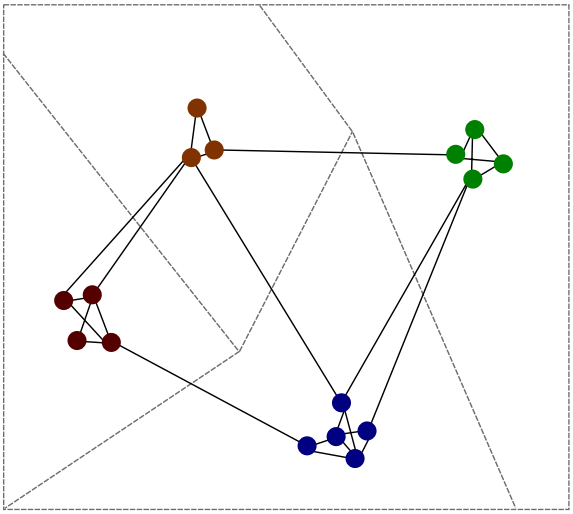


Figure 3.8: Example of a weighted clustered graph layout. Adapted from [Bourqui et al., 2007].

In the work presented by [Huang and Eades, 1998], authors propose an interactive system to visualize and explore large graphs. The system divides the process into four layers: a large graph G, a clustering C of G, an abridgment C' of C and a picture of C'. These layers represent different stages of the process and of the interaction with the user. The clustering and abridgment stages summarize the content of the graph in such a way the content be readable.

In [Li and Takatsuka, 2004], authors adapt the visualization techniques of degree of interest and logical fisheye view to clustered graphs. With these approaches, the user focuses its attention on some fragments of the graph. Authors propose a transformation of the classic interaction techniques such as filtering and browsing for this kind of graphs. Thus, the algorithm uses the subjacent hierarchical structure to find the user's points of interest and present the information in an intuitive way.

In the work presented by [Gansner et al., 2005], authors propose a model inspired on fisheye views but not just from a geometric perspective but taking into account topological considerations. Using the topology of the graph, the model reduces the number of displayed elements, i.e., reducing the level of details in zones outside the focus of interest. Thus, this method preserves the topological and geometric characteristics of the graph, keeping the semantic meaning of it.

The method proposed by [Kumar and Garland, 2006] is a graph visualization algorithm for time-varying graphs. The algorithm uses a hierarchical approach to structure the graph and uses a force-directed layout algorithm; once the graph is structured, nodes

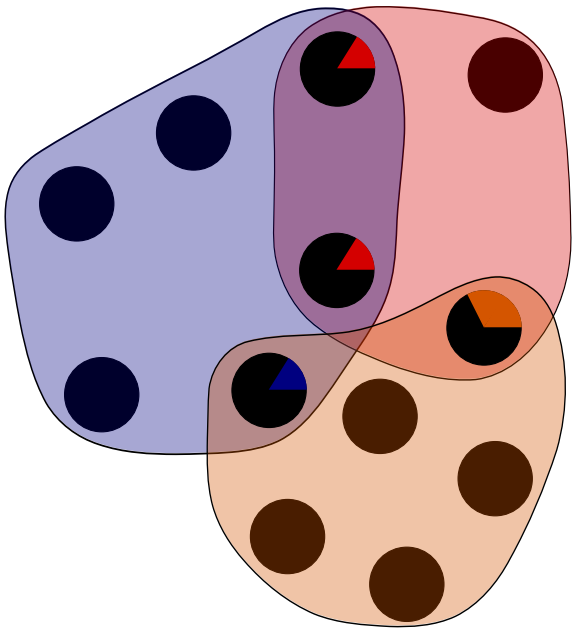


Figure 3.9: Layout of a clustered graph of overlapping groups

and edges are filtered using an interactive tool and allows the user to explore the graph taking advantage of its hierarchical structure. Additionally, the algorithm puts element from the graph according to temporal variables, maintaining the coherence of those elements.

The work presented by [Shen et al., 2006] is a visual analysis model for heterogeneous networks, i.e., networks of nodes of different types. This method uses the LinLog (Section 3.4.4) approach to layout the graph and define the communities. Then it proposes some node importance measures to filter the network using semantic and structural aspects of it.

Besides to the clustered graphs layout algorithm proposed by [Shi et al., 2009], authors propose an interaction system oriented to display graph elements according to certain degree of importance in a given moment. One operation is the semantic zoom-in/zoom-out, which is designed not just to change the size of selected items but to remove "distracting" and not interesting nodes and edges. Other operation is the hierarchical drill-in/roll-out which allows to navigate the graph in different hierarchies, revealing or hiding elements.

In [Archambault et al., 2007a, Archambault et al., 2008] authors propose a method to dynamically display graphs and their associated hierarchy. The method is based on interactive feature exploration, which allows to dynamically lay out the graph as the

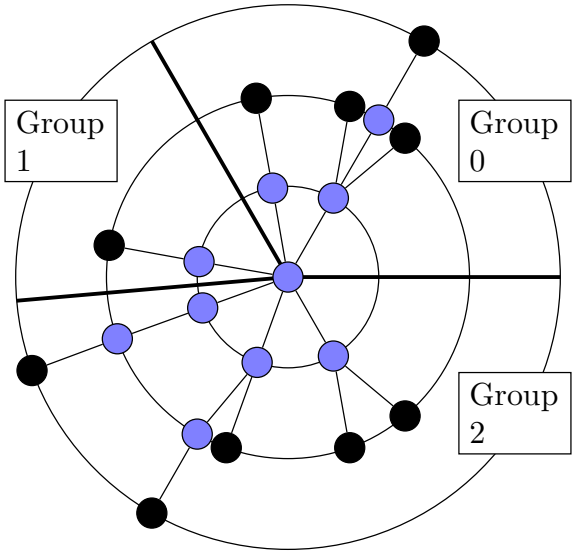


Figure 3.10: Layout of a hierarchical graph in a radial structure. Adapted from [Mun and Ha, 2005].

user explores the hierarchy. This feature is useful to reduce the computation time of the layout since it is only required to place a reduced number of nodes compared to the total size of the graph.

In the work presented by [Baur and Brandes, 2008] authors present a layout algorithm for micro/macro graphs, which can be seen as clustered graphs in the following sense: the macro level will represent the groups of the partition and the micro level contains the nodes and the edges within each group. This representation complies the restriction presented by [Eades and Feng, 1997] in which the distance between two groups within the hierarchy must be 1. The algorithm has been designed for reducing the number of crossing between the macro level nodes by placing the micro level nodes in such a way the crossing between the edges of this level do not intersect any macro level edge. This algorithm takes into account therefore any partition of the graph and aims for reducing the number of crossings.

3.4.6 GRAPH VISUALIZATION AND MANIPULATION TOOLS

The goal of this thesis is not the development of a new graph or social network manipulation and analysis software; an application with these capabilities is far beyond the scope of this work.

Rather we want to propose a set of models for detecting and visualizing communities

in a social network. These methods have to be implemented in order to be tested. Thus, we would use an existing graph manipulation engine that can be extended through plugins. Table 3.6 presents a non-exhaustive set of tools and libraries for manipulating graphs.

Type	Name	Language	Capabilities
Library	GraphViz [Ellson et al., 2001]	C++	This is a graph visualization software that implements some general graph layout algorithms. It is not designed to be used in an interactive environment.
	OGDF [Chimiani et al., 2007]	C++	This library contains several graph drawing and manipulation algorithms, providing a graph manipulation engine.
	Prefuse [Heer et al., 2005]	Java	This is an extensible framework for manipulating graphs in interactive ways. The framework includes connections to Web search engines and SQL storage interfaces.
Software	Cytoscape [Shannon et al., 2003]	Java	This is a software project created for analyzing protein interaction and molecular networks. However, it can be used with complex networks. It is extensible through plugins.
	Gephi [Bastian et al., 2009]	Java	This is a graph manipulation tool created to provide users with a simple but powerful tool. It includes several layout and partitioning algorithms and graph measures as well. It is extensible via plugins.
	Tulip [Auber, 2003]	C++	This is also a graph manipulation tool created to manipulate any kind of network. It provides a set of general manipulation algorithms, including layout, partitioning and coloring. It is extensible via plugins.

Table 3.6: Tools and libraries for manipulating graphs and complex networks

The libraries and toolkits can be used as the engine of a complete software tool that allows interacting with and visualizing the graph and the data associated to the graph. On the other hand, the software tools already have an interaction and manipulation layer besides the graph engine. Thus, we stick with the extensible software tools.

Thus, since all the presented software tools are extensible through plugins, we would like to select the one that best fits with our needs. We need to, not only manipulate the graph structural variable but store and manipulate the composition variable. We found that Tulip is more flexible in terms of plugin development: it allows to implement different views of the graph, making possible to implement a complete analysis workflow. Additionally, being Tulip written in C++ makes the plugins to be closer to the hardware and thus facilitating if it is required, implement parallel operations for example.

3.5 CONCLUSION

In this chapter relevant works in community detection and visualization were presented. Before the definition of community detection, some concepts from data clustering were included; several of these concepts are inherited by the graph clustering approaches, including quality and proximity measures, algorithms structure and the idea of intra-group density versus inter-group sparsity.

Then, some graph clustering quality measures were introduced, presenting their objectives, complexities and where available, their limitations in order to test partitions. The measures were divided into two main groups: connectivity based and metric based. The first group uses the idea of reducing the number of edges connecting groups and increasing the number of edges within groups, which is the typical way to calculate the "goodness" of a graph partition. The second group borrows the ideas of distance from data clustering and transforms the graph into a set of $p-$dimensional points in order to calculate distances and metric measures. In general, there are no consensus about what is a good partition or how to calculate the goodness because it is highly dependent of the type graph and the application, making it difficult to find a generalization.

Once the quality measures were presented, several relevant graph clustering algorithms were introduced. The algorithms were divided according to the approach used to calculate the partitions. In general the graph partitioning algorithms have been designed to find groups of nodes based just on the topological configuration of the graph, however, as some author pointed out, social networks contain more information than just nodes and edges but rather, over those nodes and edges may lay semantic information which can be useful to identify groups of well connected and semantically close people.

Table 3.7 shows some of the classic community detection algorithms found in literature. As presented in Chapter 3, most of them use only the structural variable to find the groups within the social network. The last one uses both the structural and the composition variables to find partitions of well connected and similar nodes, however the method needs to make assumptions about the number of communities existing on the social network.

Finally, a set of methods for drawing clustered graphs were presented. These methods exploit the inherent, hierarchical structure of a clustered graph. These methods inherit ideas from information visualization and data clustering in which the main goal is to represent the groups well separated from other groups, i.e., represent the idea of the low intra-cluster and high inter-cluster distances.

Table 3.8 presents a list of clustered graph layout algorithms and their different approaches and capabilities regarding the variables required by the algorithm to lay out the clustered augmented social network.

In general most algorithms to lay out clustered graphs do not use more information than the nodes composing each group to place these nodes, and do not take into account, or use, the composition information to place, for example nodes with similar composition information close each other. Three special cases are shown in Table 3.8: methods which do not use the affiliation information. These methods use the structure of the graph to visually, produce an affiliation variable while it is being depicted.

To the best of our knowledge, few methods have been developed to try to address this problem, which is becoming important because of the explosion of social networking sites. In Chapter 4 will be presented a novel method for merging the structural and composition variables to find partitions of well connected and similar nodes.

On the other hand, clustered graph visualization techniques are designed to represent the groups and the hierarchy of the partition, not taking into account other elements of the social network analysis like central actors identification. In Chapter 5 will be presented a clustered graphs layout algorithm, which besides of presenting the bounds of each group, highlights the connections between each community.

Type	Examples	Variable
Hierarchical clustering	Agglomerative and divisive algorithms [Fortunato, 2010],[Hastie et al., 2009]	Structural
Partitional clustering	Node Indexing [Rattigan et al., 2007]	Structural
	Cliques enumeration [Du et al., 2007]	Structural
Spectral methods	Sigmoid commute-time kernel generation [Yen et al., 2009]	Structural
Divisive Algorithms	Betweenness generalization [Girvan and Newman, 2002, Newman and Girvan, 2004]	Structural
	Consistency testing [Tyler et al., 2003]	Structural
	Hihgest betweenness removal [Newman, 2001]	Structural
Modularity optimization	Hierarchical approach [Newman, 2004b]	Structural
	Modularity variation store [Clauset et al., 2004]	Structural
	Modularity variation for similar communities [Wakita and Tsurumi, 2007]	Structural
	Fast unfolding of communities [Blondel et al., 2008]	Structural
Other methods	Genetic algorithms [Pizzuti, 2008a, Pizzuti, 2008b]	Structural
	Edges as topics of interest and fuzzy partitions [Pathak et al., 2008, Sachan et al., 2012]	Structural+
	Random walk for structural and semantic similarity [Zhou et al., 2009]	Structural + Composition

Table 3.7: Comparison of classic community detection algorithms

		Variables		
Method	**Approach**	**Structural**	**Composition**	**Affiliation**
[Eades and Feng, 1997]	Multi-level/ Force directed	No[4]	No[5]	Yes
[Tamassia, 1987]	Rectangles & straight lines	No[4]	No	Yes
[Noack, 2003]	LinLog/ Force model	Yes	No	No
[Brockenauer and Cornelsen, 2001]	Hierarchic / Quotient graph	No[4]	No	Yes
[Bourqui et al., 2007]	Multi-level/ Force directed for weighted graphs	Yes	No[5]	Yes
[Santamaría and Therón, 2008]	Overlapping clustered graphs	No[4]	No[5]	Yes
[Mun and Ha, 2005]	Radial representation of the hierarchy	Yes	No	No
[Batagelj et al., 2011]	Hierarchical/ visual clustering	Yes	No	No
[Shi et al., 2009]	Kamada-Kawai based	No[4]	No	Yes
[Archambault et al., 2007b]	Topological features based layout	Yes	No	Yes
[Baur and Brandes, 2008]	Multi-Circular of micro/macro graphs	Yes	No[5]	Yes

[4]The structural variable has been used to derive the affiliation variable: there is an implicit use.
[5]Despite it has not be formally defined, it could be possible to include composition information.

Table 3.8: Comparison of clustered graphs layout algorithms

Chapter 4

Integration of variables in an augmented social network

4.1 Introduction

As presented in Chapter 2, a social network contains three variables: a structural variable, describing the links or connections between the members of the network; a composition variable containing information about each actor of the network; and an affiliation variable, mapping each actor to a group.

The structural variable can be used to describe the social network as a whole by analyzing only the connections between actors. On the other hand, the composition variable allows to analyze each actor according to its individual information, such like food preferences, professional competences, languages and preferred books. This information can be used to create categories of the actors of the network, but it is in general, independent of the structure. The third variable is more special; it maps the assignation of each node to one category: this variable is derived from the previous two. If the affiliation variable is derived from the structure, then the affiliation represents a partition of the nodes strongly connected, or communities; if it is derived from the composition variable, then the affiliation describes categories of nodes grouped according their similarity.

This chapter presents the part of the model devoted to integrate the variables composing the social network. The integration is made through the use of the affiliation variables generated by the structural variable combined with the different points of view that can be derived from the composition variable.

But first, let us introduce some basic notations to be used on this chapter. Let $\mathcal{S}^{+}\left(G, PoV_{F^{*}}, \mathcal{A}\right)$ be an augmented social network as in Definition 2.3.2; $\mathcal{A}$ represent a partition $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$ of the set V into k disjoint subsets; $e\left(u, v\right) \in E$ is the edge between the nodes u and v, $e\left(u\right)$ is the set of all edges to or from the node u. Let $E\left(C_i, C_j\right), 1 \leq i, j \leq k$ be the set of edges connecting groups i and j.

$E(C_i, C_i) = E(C_i), 1 \leq i \leq k$ is the set of edges connecting nodes only within the group i. Note that $E(C_i, C_j) \subset E$.

Let $\mathcal{A}_G$ and $\mathcal{A}_{PoV_{F*}}$ be two auxiliary affiliation variables derived from the structural and the composition variables respectively that will be used during the variable integration process. These affiliation variables represent two partitions, the first one with groups of well connected nodes and the second one with similar nodes.

Note that the partitions described by $\mathcal{A}_G$ and $\mathcal{A}_{PoV_{F*}}$ are created using different grouping criteria, hence their quality measures are different. This issue will be reviewed later during the integration of both affiliation variables.

The chapter is organized as follows: Section 4.2 some clustering quality measures and their behavior are presented. Next in Section 4.3, the proposed approach for integrating the structural and composition variables of a social network into a single affiliation variable is presented, and in Section 4.4 some experiments regarding the quality of the partition are performed before the conclusion.

4.2 REVIEW OF CLUSTERING QUALITY MEASURES

As presented in Chapter 3, there are several measures of the quality of a partition, or in terms of the social network variables, the quality of the affiliation variable.

As introduced in the previous section, each affiliation variable in the model has a different set of quality measures to be used. This difference lies on the criteria used to create the partitions: in the case of $\mathcal{A}_G$, the proportion of edges within each group compared to the number of edges connecting groups, and in the case of $\mathcal{A}_{PoV_{F*}}$, the overall similarity of elements in each group.

It is somehow straightforward to measure the quality of each affiliation variable, however, measuring the quality of a partition involving both variables at the same time is quite complicated due to the nature of each of those variables.

In the next sections first we review some quality measures for structural and composition derived partitions, then we discuss about the divergent behavior of those measures.

4.2.1 MEASURING THE QUALITY OF STRUCTURAL PARTITION

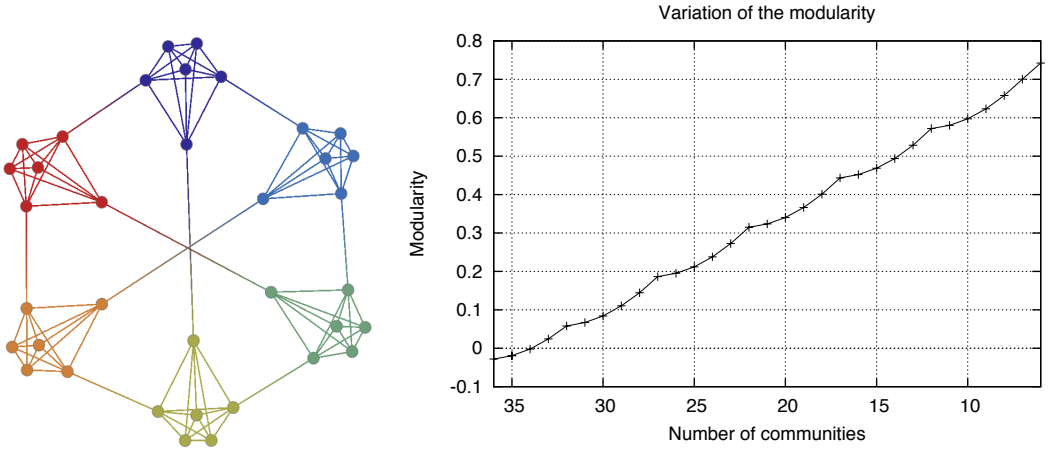
Most of these measures are based on the notion of intra-cluster density versus inter-cluster sparsity, which is in general the ratio of edges within each group compared to the number of edges connecting groups: a graph has a community structure if the number of edges connecting communities is low. This notion has been captured into the equation 3.19.

According to [Fortunato, 2010], the most used measure for assessing the quality of a graph partition is the modularity Q, we focus hence on this measure.

The modularity, as expressed in equation 3.27, represent the proportion of the edges within the groups minus the number of edges connecting each group.

The trace of the matrix $\mathbf{e}$ represents the fraction of edges falling within groups. When the trace is maximized the groups will concentrate the edges of the graph, reducing the number of edges outside groups. Thus, given a partition with a maximal Q, any further change of the configuration of the partition will reduce the modularity value because it will move edges out of the groups.

To show the behavior of the modularity we use an artificial graph with 36 nodes and 198 edges. The affiliation variable $\mathcal{A}_G$ derived from the structure is composed of 6 groups of nodes. Figure 4.1(a) presents the described graph.



(a) Representation of the graph used on this example.

(b) Variation of the modularity

Figure 4.1: Example of the variation of the modularity for a community graph

Table 4.1 presents the matrix $\mathbf{e}$ with the fraction of edges connecting communities for the graph presented above.

Figure 4.1(b) shows the variation of the modularity during the community detection process. During the first step of the algorithm each node is assigned to one community, the subsequent steps merges the communities if that merge increases the modularity. Thus, the algorithm finds a partition of 6 communities and a modularity value of $Q = 0.7424$. Note that the modularity found in this case is maximal, and cannot be improved any further; instead any change of the partition configuration may reduce the modularity.

	1	**2**	**3**	**4**	**5**	**6**
1	0.152	0.005	0	0.005	0	0.005
2	0.005	0.152	0.005	0	0.005	0
3	0	0.005	0.152	0.005	0	0.005
4	0.005	0	0.005	0.152	0.005	0
5	0	0.005	0	0.005	0.152	0.005
6	0.005	0	0.005	0	0.005	0.152

Table 4.1: Community edges proportion matrix $\mathbf{e}$ for the graph presented in Figure 4.1(a)

4.2.2 MEASURING THE QUALITY OF THE COMPOSITION PARTITION

Composition variable describes individually the actors of the network, i.e., the nature of this information is not related to the structure of the graph. This provides a sort of freedom degree regarding the representation and extent of the features describing each actor.

In general, the composition variable of each actor can be represented in a vectorial form in which each element of the vector is a feature of the actor. Additionally, features can be defined in different spaces, as presented in Section 3.2.

To measure the quality of a partition derived from the composition variable it is necessary to define a similarity function for the given sets of features. There are several possibilities to calculate the quality of a given partition, however, each option depends on the way the features are expressed. As presented in Section 3.2, the most common used types of data are:

- Binary variables.
- Quantitative variables.
- Nominal and ordinal variables.

And for each one of those types there is a similarity measure. Since the composition information is diverse, a general measure of quality is the entropy. This measure, instead of measuring distances between elements, measures how ordered is a partition: a highly disordered set has a high entropy.

The general expression for calculating entropy in a given set C is:

$$H(C) = -\sum_{i=1}^{r} p_i \ln p_i + (1 - p_i) \ln (1 - p_i) \tag{4.1}$$

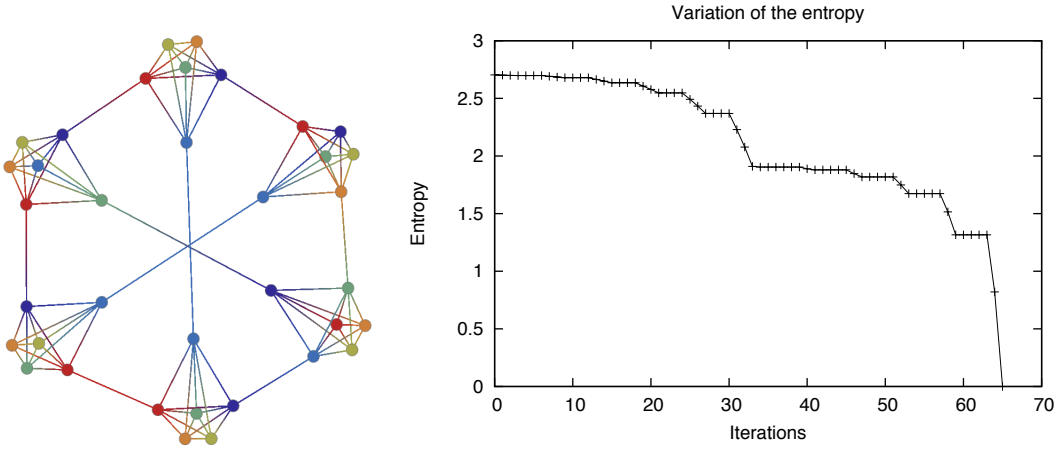
where r is the number of categories and p_i is the proportion of elements in C assigned to category i. Thus, the overall entropy of a partition $\mathbf{C}$ is:

$$\mathcal{H}(\mathbf{C}) = \frac{1}{k} \sum_{C_i \in \mathbf{C}} H(C_i) \tag{4.2}$$

where k is the number of groups in $\mathbf{C}$. Minimizing the value of the equation 4.2 will produce a good quality partition. [Li et al., 2004] present an algorithm for data clustering based on entropy minimization.

The next example is made using the same graph presented in Section 4.2.1. We create a composition variable from a list of six hobbies, and then assigning each node to one of those hobbies, we can define an affiliation variable. Therefore, this new affiliation variable define a partition of six groups, one for each hobby. For the sake of the example, we use the partition found in the previous example, that has the optimal modularity value. We assign to each node within each community a semantic category. This initial assignation makes the partition to have the highest entropy.

Figure 4.2(a) shows each community and their nodes: each color represents one category of the composition variable; there is one node from each category in the optimal modularity partition.



(a) Representation of the graph used on this example. Each color represent one category of the composition variable

(b) Variation of the entropy

Figure 4.2: Example of the variation of the entropy for a community graph

Figure 4.2(b) shows the decay of the entropy when nodes are grouped according to

their category. The initial entropy $\mathcal{H}$ is 2.703, and using a variation of the algorithm proposed by [Li et al., 2004], the entropy goes to 0. This value indicates that each new group in the resulting partition contains only equal elements.

The cost of optimizing the entropy is the reduction of the modularity, which means that most edges in the partition represented by $\mathcal{A}_{PoV_{F^*}}$ will be outside the groups, i.e., nodes in each group will not be well connected.

This behavior, reported by [Cruz et al., 2011b] and by [Zhou et al., 2009], is expected because of the opposite nature of each measure. However, is addressed, or at least taken into account while merging $\mathcal{A}_G$ and $\mathcal{A}_{PoV_{F^*}}$ in next sections.

4.3 INTEGRATING STRUCTURAL, COMPOSITION INTO AFFILIATION VARIABLES IN AUGMENTED SOCIAL NETWORKS

Our challenge is to merge the structural and composition variables to obtain an unified affiliation variable, i.e., an affiliation variable that represent groups of well connected and similar nodes. This operation deals with the divergent nature of the measures as presented in the previous section.

Instead of just merging the affiliation variables derived from the structural and composition variables, it is possible to influence the affiliation variable derived from the structure by changing this variable according to the configuration of the affiliation variable derived from the composition one. Figure 4.3 presents the general diagram of the variables integration.

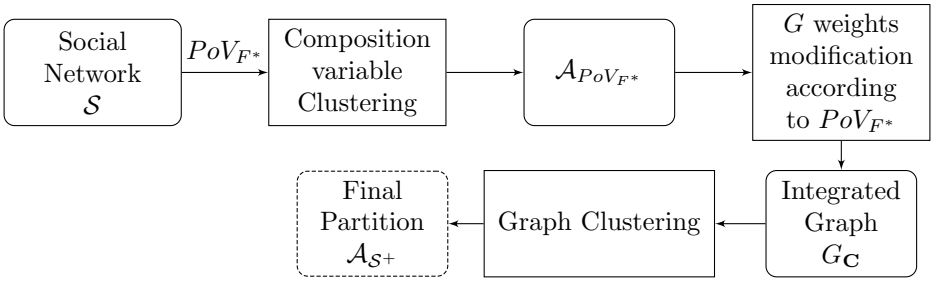


Figure 4.3: Diagram of the structural and composition variables integration

First, using one point of view from the social network the model finds an auxiliary affiliation variable $\mathcal{A}_{PoV_{F^*}}$, that contains groups of similar nodes. Then the weights of G are changed to reflect the information contained into $\mathcal{A}_{PoV_{F^*}}$: the weight of the link connecting two similar nodes is increased according to their composition variables.

Thus, the integration of $\mathcal{A}_{PoV_{F^*}}$ into G, along with a graph clustering process produces a new affiliation variable $\mathcal{A}_{\mathcal{S}^+}$ that contains information from both variables. This

affiliation variable $\mathcal{A}_{\mathcal{S}^+}$ is the third element of the augmented social network.

Each one of the clustering processes outlined here will be presented in detail in next sections.

4.3.1 CLUSTERING OF THE COMPOSITION INFORMATION

The composition information describes each actor individually. The features of the composition variable may vary according to their provenance: textual analysis from comments, general information such age and sex, formation and competences among others. Additionally each value can take values from different domains, e.g., real, integer, binary or defined by non-categorical variables.

In the literature there exists a vast number of data clustering algorithms which, most of the times, have been designed to cope with some specific problem; these methods are borrowed by researches to solve other problems generalizing them.

The first phase of our approach is the clustering of the composition variable in order to produce an affiliation variable based on the individual description of actors. We have selected three data clustering techniques for testing purposes: $k-$means, because it is a widely used clustering technique within the literature, entropy optimization, because it does not need a distance measure, making it more robust to the curse of the dimensionality, and self-organizing maps, because it does not require a final number of clusters as an input from the user.

4.3.1.1 k-MEANS ALGORITHM

$k-$means algorithm is a clustering algorithm that divides a set of n elements into k disjoint groups. The main idea is to assign each element to one of $k \leq n$ centroids.

In general, the problem solved by this algorithm is stated as: given a set $\mathcal{P}$ of observations, where each observation is represented as a $p-$dimensional real vector, find a partition $\mathbf{C} = C_1, C_2, \ldots, C_k$ according to:

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - \mu_i\|^2 \tag{4.3}$$

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 [Mahajan et al., 2009], however several heuristic approaches have been developed to find, good enough, local

minima solutions.

```
   Data: P, k
   Result: C = C_1, C_2, ..., C_k
 1 minDist ← ∞;
 2 while !stable do                    // The groups do not change any more
 3 |   C ← select_centroids(k);
 4 |   for p ∈ P do
 5 |   |   for c ∈ C do
 6 |   |   |   d ← dist(c, p);
 7 |   |   |   if d < minDist then
 8 |   |   |   |   minDist ← d;
 9 |   |   |   |   winner ← c;
10 |   |   |   end
11 |   |   end
12 |   |   assign( p, C_winner );                    // 1 ≤ winner ≤ k
13 |   end
14 end
```

Algorithm 1: k-means classic algorithm

The algorithm 1 presents the general $k-$means algorithm. This algorithm 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 [Mardia et al., 1979] to select k is:

$$k \approx \sqrt{n/2}$$

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

4.3.1.2 Entropy optimization

The problem statement for this kind of algorithms is to minimize the entropy value for the partition using the equation 4.1. The criterion used is similar to that presented in Section 4.2.2. Typically an entropy optimization algorithm will use a Monte-Carlo approach to find the best location for the n elements of a data set.

Let us note that the entropy equation only takes into account the proportion of each feature present in each group in the partition, it does not need to define a similarity, or dissimilarity measure, which is an advantage for clustering the composition variable since

it can be composed of different types of features: nominal, binary and real variables for example.

Data: $\mathcal{P}$, k
Result: $\mathbf{C} = C_1, C_2, \ldots, C_k$
$H_0 \leftarrow$ Initial entropy;
while *!stable* **do**
  $x \leftarrow 1 + u \times (|\mathcal{P}| - 1)$; // $u \backsim U[0, 1]$
  Select a target cluster B;
  Put x into B;
  $H \leftarrow$ total_entropy();
  if $H \geq H_0$ **then**
    Put x into its original cluster;
    $H \leftarrow H_0$
  end
  $H_0 \leftarrow H$
end

Algorithm 2: Entropy minimization algorithm as presented by [Li et al., 2004]

Algorithm 2 shows the entropy based algorithm presented by [Li et al., 2004]. This algorithm performs a Monte-Carlo simulation to find a partition with a minimal entropy. Similar to k-means, this algorithm requires the expected number of k groups, having the same issues to find it.

4.3.1.3 SELF-ORGANIZING MAPS – SOM

Self Organizing Maps (SOM) or Kohonen maps [Kohonen, 1997], is an unsupervised clustering method that uses an adaptive neural grid to obtain groups based on the similarity of the input patterns according to the distribution of the points in the data set. In general this method does not need any *a priori* knowledge about the data [Chifu and Letia, 2010], just the definition of the size of the grid, which can be derived from the data set size; it also has several advantages regarding other clustering methods such as $k-$means, which is less robust to high-dimensional patterns and in which the number of k clusters has to be defined as an input for the algorithm [Kantardzic, 2002]. Additionally, the result of the SOM process can be used to analyze qualitative variables, such as the composition information by using a 2D representation of the map (U-matrix)

or heat maps among others [Kohonen, 1997].

Data: $\mathcal{P}$
Result: $\mathbf{C} = C_1, C_2, \ldots, C_k$
while *!stable* **do**
  for $p \in \mathcal{P}$ **do**
    winner $\leftarrow$ competition (p);
    cooperation $(p, winner)$;
  end
end

Algorithm 3: Self-Organizing maps algorithm

Algorithm 3 outlines the SOM clustering process. At each step the selected pattern is assigned to one neuron of the grid. The algorithm has two main steps: first, a competition step, in which the distance between a pattern p and all the neurons in the grid is calculated; the neuron with the smaller distance is marked as the winner and the pattern p is assigned to that neuron.

The second step is the cooperation. During this step the weights of the winner neuron are updated according to the following rule:

$$\mathbf{w}^i = \mathbf{w}^{i-1} + \lambda \times \left(p - \mathbf{w}^{i-1}\right) \tag{4.4}$$

where $\mathbf{w}$ is the weight vector of the neuron and λ is the learning rate. This adjustment will make the neuron grid to self-adapt to the dataset. The algorithm is said to be stable when there is no changes on the assignation of patterns to the neurons from one iteration to another.

4.3.1.4 CLUSTERING ALGORITHM BENCHMARKING

So far we have presented three data clustering algorithms. These algorithms use different approaches and measures to divide a set of n elements into k disjoint groups. The behavior of each algorithm depends on the type of data used and on the assumptions made to set up the parameters of each one.

In the case of $k-$means and entropy optimization algorithms it is necessary to define the expected number of groups of the partition, and in the case of SOM, it is necessary to define the size of the network. These parameters will impact the performance and the final result of the clustering, which may be poor.

Since any clustering algorithm can be used, we want to test the performance of the three previous algorithms in terms of quality of results and execution time. To test the algorithms we will use the Adjusted Rand Index – ARI [Rand, 1971, Hubert and Arabie, 1985],

which gauges the similarity between two data clusterings.

Thus, given a confusion matrix $\mathbf{N}$ with its rows representing the groups of one partition and the columns, the groups of the other partition, and such that each $\mathbf{N}_{ij}$ represent the number of common elements between groups i and j; the adjusted rand index has been defined on equation 3.48.

The ARI vary from 0, indicating that the two partitions are completely different, to 1, indicating that elements in each partition were assigned to the same groups.

To calculate the ARI we need to generate three confusion matrices, one for each clustering algorithm: $\mathbf{N}_{k-means}$, $\mathbf{N}_{Entropy}$ and $\mathbf{N}_{SOM}$. Each algorithm will produce a partition that will be represented by an inclusion matrix $\mathbf{A}$; as presented in Chapter 2 this binary matrix indicates whether an element belongs to a group or not.

We will use the Zoo dataset from the UCI ML Repository [Frank and Asuncion, 2010], which contains 101 elements divided into 7 categories and 16 features. The last feature is the type, so we will not use it during the algorithms training, however it will be used to generate a reference inclusion matrix $\mathbf{A}_r$. Confusion matrices can be determined using the inclusion matrices of each partition as presented in equation 4.5.

$$\mathbf{N}_{\cdot} = \mathbf{A}_r^T \mathbf{A}_{\cdot} \tag{4.5}$$

where $\mathbf{A}_r$ is the affiliation matrix of the reference partition, $\mathbf{N}_{\cdot}$ is the confusion matrix of some clustering algorithm and $\mathbf{A}_{\cdot}$ is the affiliation matrix of the partition produced by the clustering algorithm.

$\mathbf{N}_{k-means}$

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	21	0	0	0	17	0	3
2	0	20	0	0	0	0	0
3	0	0	2	0	0	0	3
4	0	0	0	0	0	0	13
5	0	0	4	0	0	0	0
6	0	0	0	8	0	0	0
7	0	0	1	5	0	4	0

$\mathbf{N}_{Entropy}$

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	0	0	0	0	19	22
2	0	11	0	0	9	0	0
3	1	0	3	1	0	0	0
4	0	0	0	13	0	0	0
5	0	0	0	4	0	0	0
6	7	0	1	0	0	0	0
7	2	0	2	6	0	0	0

$\mathbf{N}_{SOM}$

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	30	0	0	8	0	0	3
2	0	20	0	0	0	0	0
3	1	0	1	0	1	2	0
4	0	0	0	0	0	0	13
5	0	0	0	0	4	0	0
6	0	0	0	0	8	0	0
7	0	0	4	0	6	0	0

Table 4.2: Confusion matrices for each algorithm

Table 4.2 shows the confusion matrices for the partitions obtained by each algorithm on the Zoo data set. For example, in the matrix $\mathbf{N}_{k-means}$, the position n_{11} indicates that there are 21 elements that were grouped in the same group in both partitions: the reference one and the one obtained by the $k-$means algorithm. Also note that the sum of each row has the same value in each matrix: this sum is the number of elements of the same type in the reference partition.

Now, using the equation 3.48 we calculate the index for each partition. The result are presented in Table 4.3.

Algorithm	**ARI**
k-Means	0.5983
Entropy optimization	0.5246
SOM	$\mathbf{0.6582}$

Table 4.3: Adjusted Rand Indices for each partition respect to the reference partition

According to the results the SOM produces the closest partition to the reference partition: this means that this algorithm produces more consistent groups with those originally provided by the data set.

This data set is quite small to test the time performance of the algorithms, therefore this test is executed using an artificial data set with 2000 elements and 100 features. The test has been performed by progressively adding these 100 features one after the other in each execution; in this way we can assess the sensibility of each algorithm to the number of dimensions, or the number of features composing the point of view.

Figure 4.4 present the results of the time performance for the algorithms.

The execution time for each algorithm, at least for this data set, is linear. The difference lies in the constants involved. The best algorithm, in terms of execution time, is the entropy optimization algorithm. This comes from the way the entropy is calculated: it is the percentage calculation of each dimension of the elements present in each group, which in general needs less operations than the calculation of the euclidean distance. This is the primordial difference with the other two algorithms, which use a Chebyshev based measure, such as the Euclidean distance, that requires more operations. The worst execution time is for the $k-$means algorithm, which grows linearly, with a greater slope than the SOM.

Thus, according to the results of the ARI and the execution times, we chose SOM as clustering algorithm for the structural and composition variables. Additionally to these results, one advantage of SOM is that it does not need any *a priori* information, hence the researcher does not need to make any assumption about the communities, as preferred for community detection algorithms [Newman and Girvan, 2004], and given its nature, it is more robust to a high number of dimensions, as can be the case for the composition variable.

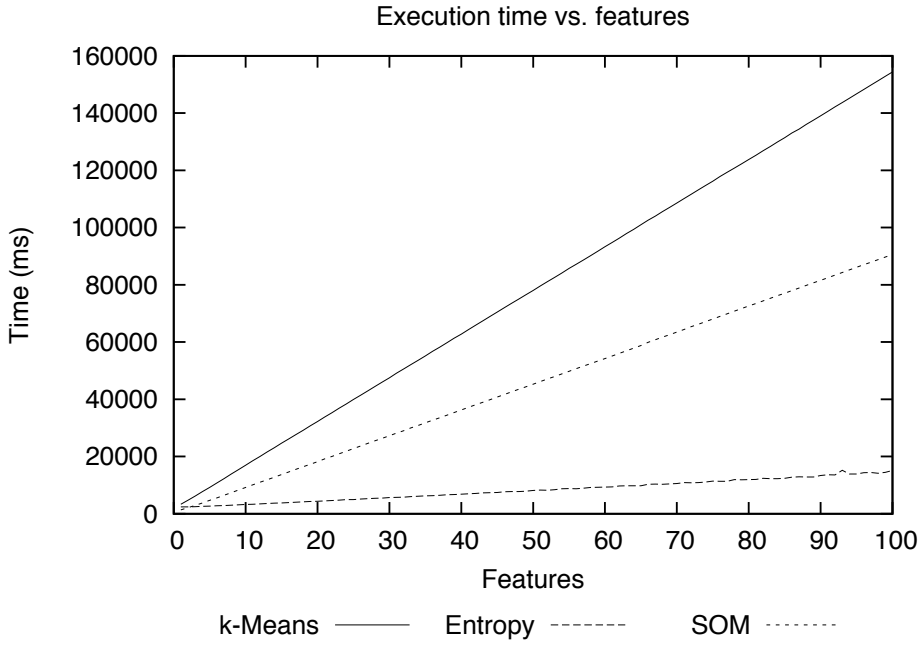


Figure 4.4: Execution time for the three clustering algorithms increasing the number of features composing the point of view

4.3.2 INFLUENCE OF THE COMPOSITION VARIABLE ON THE GRAPH STRUCTURE AND GRAPH CLUSTERING

The partition generated from the composition variable describes the nodes of the graph only in terms of individual information and in general, without any information related to relationships between actors. The composition variable is divided into different points of view, which allows generating several partitions, one for each point of view.

Thus, given a point of view PoV_{F^*}, each node can be characterized by its vector of features, or an instance ξ of the point of view. Each instance ξ is an input pattern for the clustering. The SOM network groups the nodes according to the similarities of their features. The SOM network $\mathcal{N}$ has been implemented using a square lattice of $l \times l$ neurons, where $l = \left\lceil \left(\sqrt{n/2}\right)^{1/2} \right\rceil$ and n is the number of nodes of G.

Algorithm 4 presents the algorithm for integrating the structural and composition variables to produce a new affiliation variable. The algorithm takes as parameters an augmented social network and a parameter α that is used for changing the weights of the graph.

The first step of the algorithm is to select a point of view from the set of features F^*

Data: $\mathcal{S}^{+}\left(G\left(V,E\right),F^{*},\mathcal{A}\right),\alpha$
Result: $\mathbf{C}_{PoV_{F^{*}}}$
$PoV_{F^{*}} \leftarrow$select_point_of_view(F^{*});
$\mathbf{C}_{SOM} \leftarrow$SOM_algorithm($PoV_{F^{*}}$); // Part. from the composition variable
for $e \in E$ **do**
  $n_s \leftarrow e.source$;
  $n_t \leftarrow e.target$;
  $e.w = 1$;
  if $C\left(n_s\right) = C\left(n_t\right)$ **then** // $C\left(n_s\right), C\left(n_t\right) \in \mathbf{C}_{SOM}$
    $e.w+ = \alpha\left(1 - d\left(\mathcal{N}_{n_s,n_t}\right)\right)$;
  end
end
$\mathbf{C}_{PoV_{F^{*}}} \leftarrow FU\_algorithm(G\left(V,E,w\right))$;
return $\mathbf{C}_{PoV_{F^{*}}}$

Algorithm 4: Structure and composition variables integration algorithm.

of the augmented social network. This point of view $PoV_{F^{*}}$ is the composition variable of the social network and it is used by the SOM algorithm to find group of similar nodes, i.e., an affiliation variable $\mathcal{A}_{PoV_{F^{*}}}$ derived from the composition variable.

The groups in this partition contain actors which are semantically close, i.e., they are similar. To measure the quality of the partition we use entropy of the partition as presented in Section 4.2.

Once the semantic partition $\mathcal{A}_{PoV_{F^{*}}}$ has been found it is possible to begin the second step of the proposed method. Using this affiliation variable $\mathcal{A}_{PoV_{F^{*}}}$ the weights of all the edges $e \in E$ are changed according to:

$$w_{ij} = 1 + \alpha\left(1 - d\left(v_i, v_j\right)\right)\delta_{ij} \tag{4.6}$$

where $\alpha \geq 1$ is a constant value, $d\left(v_i, v_j\right)$ is the distance between the nodes i and j in terms of the selected point of view, and $\delta_{ij} = 1$ if v_i and v_j belong to the same partition in $\mathcal{A}_{PoV_{F^{*}}}$, $\delta_{ij} = 0$ otherwise.

After the weights are changed according to Equation 4.6, a partition based on the variable$\mathcal{A}_{\mathcal{S}^{+}}$ is found using the Fast Unfolding algorithm. This partition contains the final set of communities, which has both, the composition information and the structural information.

The graph clustering algorithm is based on the modularity optimization algorithm proposed by [Blondel et al., 2008]. This is an agglomerative algorithm which maximizes

the modularity Q, presented by [Newman and Girvan, 2004], by locally changing the composition of the communities. The algorithm has two steps: i) first, the modularity optimization, and ii) a community aggregation to create a graph of communities. These two steps are repeated until the modularity cannot be improved.

By using Equation 4.6 we transform the graph into a weighted graph. In this case, the modularity is calculated from the sum of the weights and not only counting how many edges fall inside a community versus the edges falling outside. This change also indicates that nodes which are "semantically" close have a stronger connection between them.

The parameter α will therefore change the structure of the network bay adjusting the weights of the graph according to exogenous information. From the community structure point of view, this adjustment will induce a change on the community structure such that the modularity optimization algorithm will find different partitions as presented by [Fan et al., 2006] and observed in this work.

In this work the new weights are calculated from the auxiliary affiliation variable $\mathcal{A}_G$ and not are generated as in [Berry et al., 2009] with the objective to overpass the resolution limit of the modularity. Therefore the implications of selecting a different α rely on how much the community structure is changed when $\alpha \to \infty$.

To measure this change when $\alpha \to \infty$ we propose a simple experiment that will be presented in in Section 4.4.

4.4 EXPERIMENTS AND RESULTS OF THE COMMUNITY DETECTION

In this section we perform some experiments to test the performance of the community detection algorithm and the integration of the structural and composition variables.

Experiments were performed using two graphs: one of DBLP authors, used by Zhou et al., in [Zhou et al., 2009], that contains 10000 authors and 65734 edges; this graph represents a co-authorship network. The second graph is a personal social network downloaded from Facebook using NameGenWeb [Hogan, 2011]. This network corresponds to people known by its owner in different contexts: family, university, high-school, coworkers, PhD studies and a former research project. This graph contains 334 nodes and 5394 edges.

Additionally to graphs there are a set of points of view. For the first graph, we use a point of view describing the topics in which researches in the network are involved with. For the second graph we use two points of view, the first one, PoV_{COMP}, describing the competences of each actor in some knowledge field, and the second one, PoV_{SPOR}, describing each actor in terms of its preferred sport. Note that for each graph exists

a trivial point of view with no features; this null point of view, PoV_{NULL}, allows the community detection algorithm to behave as if only the structural variable is available. It is used during experimentation for comparing purposes. The points of view are described in Table 4.5.

4.4.1 IMPLICATIONS OF THE α PARAMETER

The parameter $\alpha \geq 0$ is used to modify the weights of the graph according to the affiliation variable $\mathcal{A}_{PoV_{F^*}}$. The modification of the weights will influence the modularity optimization algorithm in such a way it will find groups of well connected and similar nodes. However, this influence will also change the community structure of the graph.

To measure how much the structure may change we made a simple experiment with the first graph and the two points of view presented on Table 4.5.

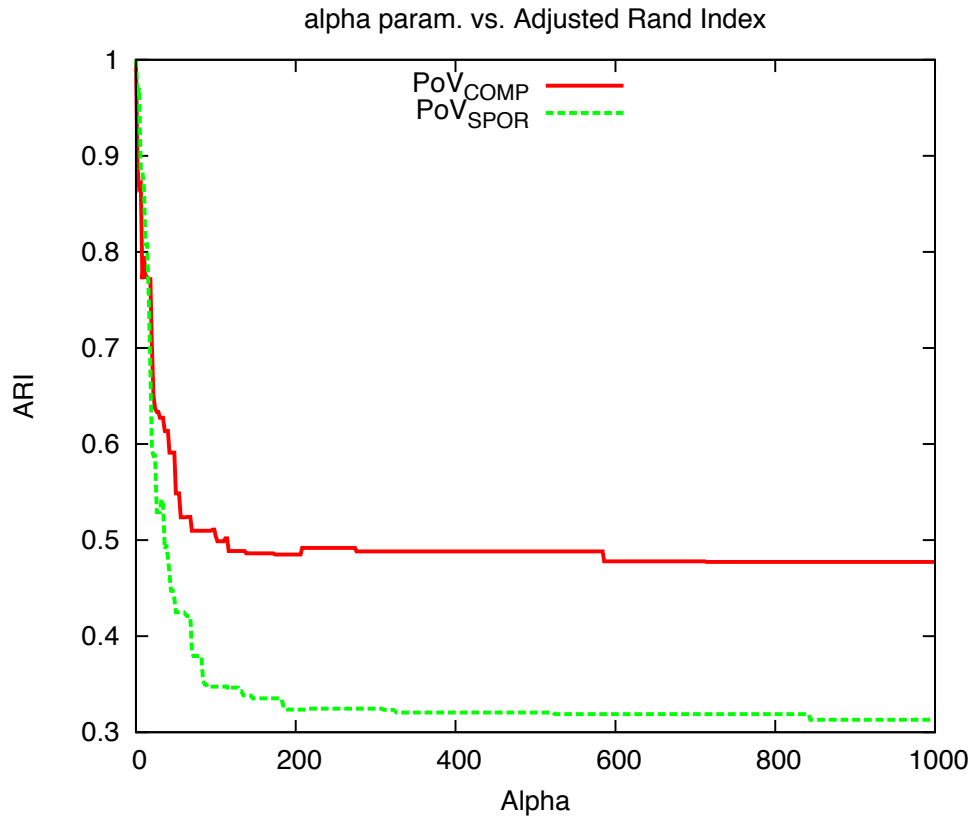


Figure 4.5: Behavior of the ARI when changing the value of α

To have a reference partition, we first obtain a partition using the PoV_{NULL}: this

partition has a high modularity value and has not been influenced. Then, each point of view will produce a new auxiliary affiliation variable $\mathcal{A}_{PoV_{F^*}}$ that will be used with a specific α parameter to modify G.

The experiment consists in change the value of α from 0 to 1000 and measure the ARI between the obtained partitions and the reference partition. Figure 4.5 shows the results of the experiment. In both cases the behavior is similar: the differences between partitions, measured by the ARI, grows as the value of α is increased. However, after a high value the ARI seems to be stabilized, showing on each case an asymptotic ARI value.

This help us to choose a value for α at least following a simple heuristic. In this case, we want to have a new partition that integrates the structural and the composition variables, which we found is a trade-off between those two variables, therefore we say that an α that produces an ARI greater than 0.5 is a good value to be used with the algorithm. We therefore for the following experiments use an $\alpha = 20$.

4.4.2 MEASURING THE QUALITY OF PARTITIONS

The execution of the community detection algorithm will produce a partition $\mathcal{A}_{\mathcal{S}^+}$ of the social network. Since the produced affiliation variable integrates the structural and composition variables, each group within this partition have two main dimensions:

- Groups of well connected nodes.
- Groups of similar nodes in terms of their composition variables.

Thus, the quality measures should include both dimensions; for this we use the measures used by [Zhou et al., 2009]: edges density for the first one and entropy for the second one. The edge density ρ for a partition $\mathbf{C}$ is defined as:

$$\rho(\mathbf{C}) = \sum_{i=1}^{k} \frac{|E(C_i)|}{|E|} \tag{4.7}$$

and the entropy is used as defined in equation 4.1.

4.4.3 EXPERIMENTS WITH THE CO AUTHORSHIP NETWORK

The graph used during this set of experiments is composed by 10000 nodes and 65734 edges. It represents a DBLP co-authoring network extracted from four different fields of computer sciences: data bases, data mining, information retrieval and artificial intelligence. Additionally, each author is described by two fields: one describing how

productive is the author, and two, the field topic that best fits to the author. This dataset has been configured by [Zhou et al., 2009], who performed the topic extraction process and assessed the productivity of each author.

Table 4.6 present the results of the clustering process for each point of view. In Section 4.2.1 were presented two simple experiments to show the nature of the quality measures for the graph variables. Since the PoV_{NULL} is performed using only the structural variable, the partition has the best modularity, hence the best density, and will give the initial entropy value.

For a non null PoV, the community detection process improves the entropy but at the cost of reducing the modularity of the partition.

The partition found by the community detection algorithm using the point of view PoV_{SCCO} has 862 communities, meaning that some of the original groups have been divided; this impacts the density value because now there are more edges connecting communities that were unified before. However, the entropy of the partition has been reduced by nearly 50% of its original value, which means that the groups are more homogeneous in terms of the composition variable of their nodes.

These results shows a better performance than those presented in [Zhou et al., 2009] using the same dataset. In that work the density for 800 groups is ≈ 0.51 and an entropy of ≈ 0.36. One of the main drawbacks of their approach is that the algorithm requires the number of groups, making the results difficult to compare.

4.4.4 Experiments with the Facebook network

The second test is performed with a graph extracted from Facebook. This graph is composed of 334 actors and 5394 edges between them. Since this social network came from a social networking site, the composition variable may be rich in terms of the profile information.

The fast unfolding algorithm find partitions with an optimal modularity, as presented in equation 3.27. It takes into account the proportion of edges within each group in its first term; hence an optimal modularity partition yields to a high density value. Here again, with PoV_{NULL} the optimal modularity is reached and the density value can be used as a reference for the other points of view. Any change on the partition configuration for the same graph will impact this value as discussed by [Cruz et al., 2011b], hence when using the other points of view, the density value is expected to decrease.

The partition found using only the structural information contains 6 communities, each one with people from different environments in which the network's owner has been involved into:

1. Group of former coworkers in a research project. This group includes persons from

administrative staff from the university where the project was developed.

2. Family and family friends.

3. Group of students and researchers from the university where the network's owner made its undergraduate and graduate studies.

4. Group of former students from the school and high school.

5. Group of people from a consulting firm where the network's owner worked before starting its PhD.

6. Group of people known during PhD studies.

This partition hence divide the nodes set in such a way the proportion of edges outside the groups is small.

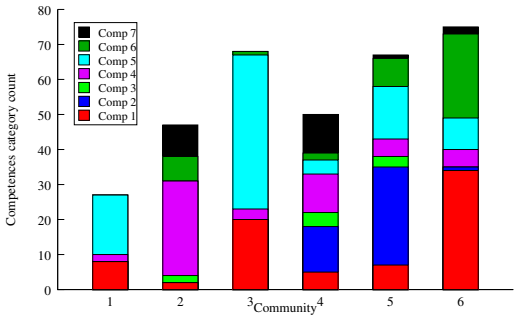
The community detection process is therefore applied to the remaining points of view: PoV_{COMP} to describe the social network under the competences perspective and the PoV_{SPOR} to influence the community discovery from the sport preferences information (Table 4.5.)

The results of this phase are presented in the table 4.7. The partitions calculated with the PoV_{COMP} and PoV_{SPOR} have, as expected, a lower density. However let us note that the modularity is greater for each additional point of view. This fact is explained because of the intensification of the edges' weights when influencing the graph with the composition information. The community detection algorithm computes the modularity using the weight of the edges and not only by counting the number of edges within the groups, making the affiliation variable $\mathcal{A}_{PoV_{F^*}}$ to describe groups with nodes that are more similar in terms of their composition variables.

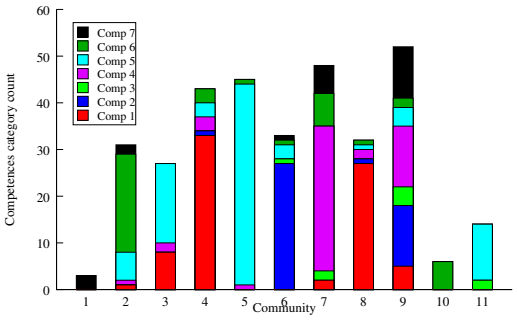
Figure 4.6 shows the distribution of competences in the partitions defined by PoV_{NULL} and PoV_{COMP}. The first partition, Figure 4.6(a) presents the partition derived from the structural variable; since this partition has been generated using only the structure of the graph, each community contains actors from almost each competence. On the other hand, the integrated partition presented in Figure 4.6(b) shows how each group is more homogeneous.

The community detection algorithm splits structural communities according to the composition variables of their members producing these homogeneous groups. This homogeneity is observed on communities 1 and 10 with 100%, 5 with 96% and 11 with 86% of their members belonging to the same competence category.

Similarly, the composition of sport preferences for the point of view PoV_{SPOR} for partitions $\mathbf{C}_{NULL}$ and $\mathbf{C}_{SPOR}$ are presented in figures 4.7(a) and 4.7(b) respectively. In

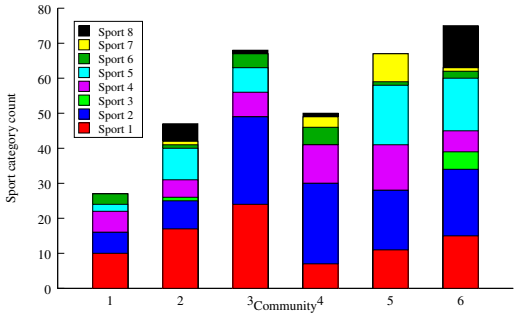


(a) Composition of competences communities for partition $\mathbf{C}_{NULL}$

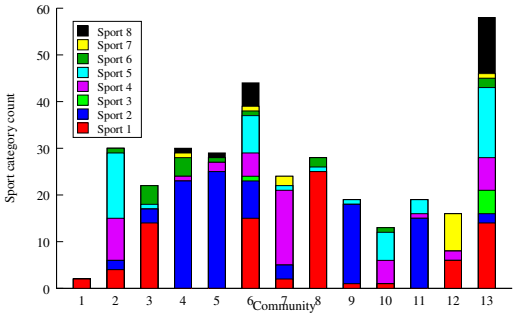


(b) Composition of competences communities for partition $\mathbf{C}_{COMP}$

Figure 4.6: Composition of communities for the first point of view



(a) Composition of sport communities for partition $\mathbf{C}_{NULL}$



(b) Composition of sport communities for partition $\mathbf{C}_{SPOR}$

Figure 4.7: Composition of communities for the second point of view

this case, in the first partition the sports seem to be more equally distributed while in the second case there are communities with dominant sports. For example, in groups 4, 5, 8, 9 and 11 in $\mathbf{C}_{SPOR}$, more than 75% of their members prefer the same sport. This shows how by influencing the community detection process actors in each group are more similar.

The principal feature of the model is the integration of the information used to create the new partition; this impose some restrictions on the quality of the clusters i.e., they do not have an optimal modularity nor an optimal average distance but instead, there is a trade-off between them as presented by [Cruz et al., 2011c].

There is a particularity regarding the points of view used during these experiments. In the first case, the PoV_{COMP} is close to the PoV_{NULL}: it is because most of the groups reflect people belonging to the same organization, having they therefore, similar

competences, which is reflected on the FB friendships. On the contrary, the community 5 contains actors from all categories: this community contains students from school and high school, where the competences among students may be diverse. The community 3 contains acquaintances from the university, who are most either computer scientist or software engineers.

On the other hand, the PoV_{SPOR} define a set of preferences for each person. In this network, communities are not defined by sportive preferences, therefore each group will have representatives from each sport; this variety can be seen in Figure 4.7(a).

4.4.5 COMPLEXITY AND TIME EXECUTION CONSIDERATIONS

This section presents the community detection algorithm in terms of complexity and execution time, which is in general the sum of the complexities of the involved clustering algorithms.

After the selection of a point of view PoV_{F^*}, the next step in the community detection algorithm (Algorithm 4, line 2) is the clustering of the nodes according to that selected point of view. This process compares each one of the n input patterns to the $\eta = l \times l$ neurons of the network $\mathcal{N}$ and the weights of the winner neuron are recalculated.

This process is repeated until the $\mathcal{N}$ converges, i.e., the weights do not change or until a maximum number of iterations λ. The worst case occurs when all the iterations are performed, making the number of operations proportional to $\eta \times n \times \lambda$. In general the complexity is:

$$T_{SOM}(n) = O\left(|PoV_{F^*}| \times l^2 \times n\right)$$

The next step is the weight change process. During this step each edge is checked and its weight changed. The complexity is:

$$T_{\Omega}(n) = O(m)$$

The last step is the final graph clustering. This step is performed using Blondel's algorithm whose complexity $T_Q(n)$ has been reported to be linear in the number of nodes for sparse adjacency matrices [Blondel et al., 2008].

Since each step is performed sequentially, the overall complexity of the community detection algorithm is $T_{SOM}(n) + T_{\Omega}(n) + T_Q(n)$.

The tests are performed using two graphs, first a graph extracted from Twitter, with 5389 nodes and 46440 edges. The density[1] of the graph δ_G of this graph is $\delta_G \approx 0.32\%$,

[1]The density of the graph measures the number of edges versus the number of nodes of the graph, which is different of the partition density presented in the precedent sections.

it corresponds to a sparse graph. Second, the DBLP graph used in the precedent section. In both cases we use an artificial point of view composed by 100 features. The idea is to measure the impact of the size of the number of selected features on the execution time, accordingly the experiment starts with zero features, adding one by one the remaining features of the set.

Figure 4.8 shows the execution time of the algorithm for points of view of different size. The test was performed using the same graph and increasing by one the number of features used to calculate the partition generated from the composition variable.

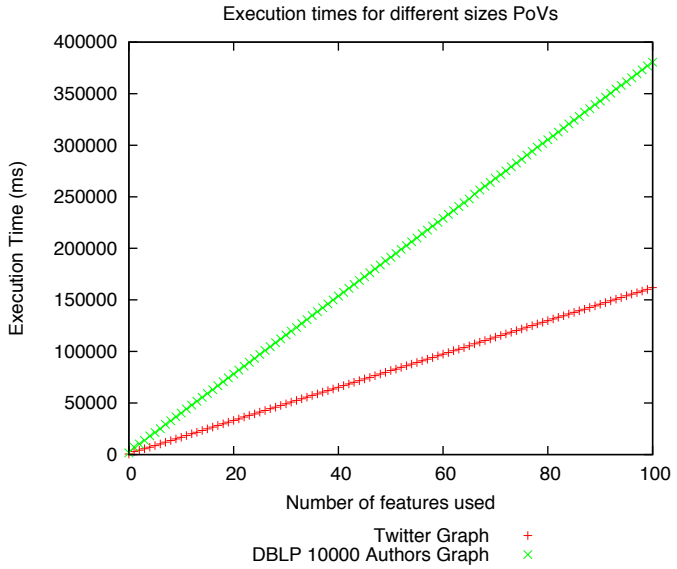


Figure 4.8: Execution times for each data set when changing the number of features included on the point of view

The increase in the execution time has a linear order whose slope changes in function of the number of features composing the point of view. The difference on the slopes is due to the different size of the data sets.

PoV	Graph	Components	Rationale
NULL	1 & 2	None	This is the null point of view, thus there are no categories, the community detection algorithm uses only the structure of the graph. This point of view produces the variable $\mathcal{A}_G$
SCCO	1	1. Type of author Values $\in [0, 2]$ 2. Topic cluster Values $\in [1, 99]$	This PoV describes each author in terms of its type: Highly prolific, prolific and little prolific (according to its publications.) Each actor belongs to one of 99 cluster of topics. This description is used in [Zhou et al., 2009, Zhou et al., 2010]
COMP	2	1. Math and science 2. Business administration 3. Law 4. Social sciences 5. Software eng. 6. Other eng. fields 7. Arts	These categories are defined according to the areas of expertise of each actor in the network.
SPOR	2	1. None 2. Football 3. Rugby 4. Tennis 5. Jogging 6. Basketball 7. Baseball 8. Swimming	These sports are defined according to the general preferences of each actor in the network.

Table 4.5: Summary of the three points of view used during experimentation

PoV	Groups	Density ρ	Entropy $\mathcal{H}$
NULL	843	0.8324	Respect to PoV_{SCCO} : 0.2744
SCCO	862	0.8245	0.1461
Zhou et al.,	800	≈ 0.51	≈ 0.36

Table 4.6: Summary of results of the community detection for the DBLP graph

PoV	**Groups**	**Q**	**Density** ρ	**Entropy** $\mathcal{H}$
NULL	6	0.7730	0.9718	Respect to PoV_{COMP} : 0.4637 Respect to PoV_{SPOR} : 0.5780
COMP	11	0.8197	0.9583	0.3161
SPOR	13	0.8270	0.8135	0.3966

Table 4.7: Summary of results of the community detection

4.5 CONCLUSION

This chapter presented an approach to integrate the structural and composition variables in a social network. Each of these variables can be used to generate two affiliation variables: $\mathcal{A}_G$ and $\mathcal{A}_{PoV_{F^*}}$, representing respectively a structural partition and a feature partition of the node set.

Each affiliation variable contains different type of information which should be measured and analyzed in different ways: the variable $\mathcal{A}_G$ defines groups of nodes which are well connected, therefore the proportion of edges connecting groups is lower than those within the groups. The variable $\mathcal{A}_{PoV_{F^*}}$ define groups of nodes that are similar according to their individual features. These affiliation variables are in general, not comparable because of how they are generated.

This juxtaposition of affiliation variables makes difficult, or at least not evident, to find partitions of well connected AND similar nodes.

The proposed approach in this chapter uses the variable $\mathcal{A}_{PoV_{F^*}}$ to influence the community detection process in such a way that makes the partition to have both features: groups of well connected nodes with similar composition variables. To the best of our knowledge this method is original and performs the variable integration better than that presented by [Zhou et al., 2009], which is one of the first works in this direction.

To generate the variable $\mathcal{A}_{PoV_{F^*}}$ we use a data clustering algorithm that finds a partition of the nodes according to the similarity of their composition variables. Three classic clustering algorithms were tested: k-means, entropy optimization algorithm and self-organizing maps. To test the performance of these algorithms two datasets were used: the first one is the "Zoo" dataset from the UC Irvine Machine Learning Repository[2] which contains 101 elements with 15 features and 7 classes. The second dataset is an artificial dataset with 2000 patterns of 100 binary features. The first dataset was used to test the performance of the algorithms in terms of quality, regarding the exiting classes of the dataset. The second dataset was used to test the performance of the algorithms in terms of execution time.

The results have shown, for the first case, that the partition found by SOM was closer to the optimal partition. This was found using the Adjusted Rand Index – ARI; regarding the execution time the best performance was obtained by the entropy optimization algorithm followed by the SOM. Thus provided the SOM had obtained good results and it is expected to perform in an acceptable time the clustering of the composition variable, it was selected to be used to find $\mathcal{A}_{PoV_{F^*}}$ during the first phase of the community detection algorithm. However, in other applications where a large amount of features or dimensions, other kind of algorithms can be considered, for example those of subspace

[2] http://archive.ics.uci.edu/ml/

clustering presented in Section 3.2.

Using $\mathcal{A}_{PoV_{F^*}}$ it is possible to change the weights of the graph according to the mutual assignation of the two nodes connected from an edge to the same $\mathcal{A}_{PoV_{F^*}}$ group. This change allows the algorithm to privilege nodes in the same category to be placed into the same community. Note that given an edge whose nodes belong to the same group in $\mathcal{A}_{PoV_{F^*}}$ will not be placed in the same group automatically, instead, the algorithm will use that information to decide later if the grouping optimizes the graph clustering criteria.

The change of the weights is made using a parameter α that allows for setting the strength to be used to influence the modularity optimization algorithm. Despite we presented a simple heuristic to select the a value for α, how to measure the impact on the community structure given certain value remains as an open question. This question is in part due to the intrinsic nature of the modularity and its resolution limits which study is beyond the scope of this work.

To test the whole community detection model we used two datasets: one modeling a 10000 nodes co-authorship network extracted from DBLP[3] by [Zhou et al., 2009]; actors in this network are described by two features: the production amount of papers and articles (three categories) and the topic most treated by the author (1 out of 100.) The second dataset is a network extracted from Facebook composed with 334 actors; each actor is described by a set of features. From this network two points of view were created, one with the academic competences, and another with the sportive preferences of each actor.

To measure the quality of the partition the edges density and the entropy of each group were used. A high density means groups of well connected node, while a low entropy means groups of similar nodes.

The results obtained for the first graph show a better performance in terms of the previous measures that those reported by [Zhou et al., 2009] with the additional advantage of not predefining the final number of communities.

For the second graph using the partition $\mathbf{C}_{COMP}$ makes it possible to extract information which is not evident before. For example, groups 2 and 7. In the first case this is a group of people who were in the group 2 of $\mathbf{C}_{NULL}$, which is the family and family friends group, but their competence (arts) is very specific compared with the other members. The second case corresponds to a group of people who were in the group 4, which contains former colleagues and researchers from an old project. Thus, these persons are strongly connected and they share the same competence.

On the other hand, the partition $\mathbf{C}_{SPOR}$, in spite it has a better modularity than $\mathbf{C}_{NULL}$, the average semantic distance is quite high. This is due to high variability of

[3] http://dblp.uni-trier.de/

the sport preferences of the actors in each group, i.e., each actor has a preferred sport independently of its environment in contrast to the partition $\mathbf{C}_{COMP}$ in which each group it is more probable to have communities with similar competences.

Thus, the model creates a more fine grained partition than $\mathbf{C}_{NULL}$, which reveals non evident information contained within the graph according to the selected point of view. This happens thanks to the generation of different partitions, one for each point of view: the definitions of points of view from the composition variable allows analyzing the social network from different perspectives making visible non-evident knowledge.

Chapter 5

Visualization of communities and their interactions

5.1 Introduction

Given an augmented social network $\mathcal{S}^+ (G, PoV_{F^*}, \mathcal{A})$, the affiliation variable $\mathcal{A}$ may be derived from the structural or the composition variable or both of a social network, we want to create a model to visualize the communities within the partition. This visual model uses the affiliation variable to layout the partition while highlighting the structural variable of the social network to focus on communities interactions.

The affiliation variable can be generated, among others, using the structural or the composition variable or both. In the previous chapter we generated an affiliation variable that integrates the structural and the composition variables. The visualization model should be able, besides of showing the groups, to show the existence of the another variable involved and its relation with the communities.

Differently of most graph layout algorithms which focus on showing each group of the partition separated from each other, the model presented in this chapter it is oriented to highlight the interactions between communities and to reveal some important roles of the nodes concerning these interactions. Note that those roles are defined by the partition and the local structure of each node, being completely independent of the layout: the layout just helps to identify them out.

The first step is to find those nodes in charge of connecting the groups, i.e., those nodes that are connected with other groups; these nodes can be seen as connecting bridges between communities. To do this, the set of nodes is divided into two types: first, the nodes whose edges start and finish in their own community, called *inner nodes*, and second the nodes which have at least one edge starting, or finishing in other community than their own, called *border nodes*.

The nodes will be placed according to a similarity measure, meaning that two similar nodes will be located close to each other meanwhile dissimilar nodes will be placed away.

To do this, it is necessary to define a similarity measure between nodes, that can be issued from the geodesic distance between the nodes or from another measure such as the comparison of neighborhoods.

Thus, using a partition of k communities, the division of nodes produces $k+1$ subsets: one of border nodes with nodes from all communities, and k subset of inner nodes, each with nodes from one community. During the algorithm, for each of the $k+1$ subsets one similarity matrix is produced, and then the algorithm will find the position of the nodes according to those matrices.

Due to the division of the nodes and the way the algorithm is executed, the resulting complexity is reduced, allowing to perform some parallel operations where it is possible.

Before presenting the algorithm we introduce basic notations required in the next sections. Given a graph $G(V,E)$ where V represents the set of nodes and E represents the set of edges, and given a partition $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$, let $e(u,v) \in E$ be the edge between the nodes u and v, $e(u)$ be the set of all edges to or from the node u. Let $E(C_i, C_j), 1 \leq i,j \leq k$ be the set of edges connecting groups i and j. $E(C_i, C_i) = E(C_i), 1 \leq i \leq k$ is the set of edges connecting nodes only within the group i. Note that $E(C_i, C_j) \subset E$.

Definition 5.1.1 (Inner node v^-)**.** *Given a group $C_i \in \mathbf{C}$, a node v is an inner node of C_i iff $\forall \varepsilon \in e(v),\ \varepsilon \in E(C_i)$.*

Therefore, an *inner node* has edges only to/from nodes in its own community. An example of the location of these nodes is presented in Figure 5.1(a).

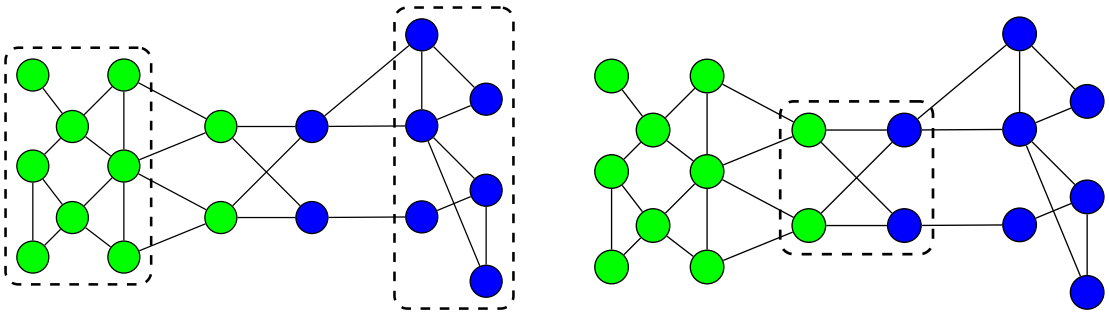
Definition 5.1.2 (Border node v^+)**.** *Given a group $C_i \in \mathbf{C}$, a node v is a border node of C_i if $\exists \varepsilon \in e(v) : \varepsilon \notin E(C_i)$.*

Therefore, a *border node* has at least one edge to/from a node in a different community. An example of the location of these nodes is presented in Figure 5.1(b).

The node set is hence divided into $k+1$ subsets: k subsets $V_i^-, i = 1, 2, \ldots, k$ containing the *inner nodes* from each community and one subset V^+ composed of border nodes from all communities.

This division is the cornerstone of the layout algorithm, helping the visualization of interactions between communities in the social network.

This chapter is organized as follows: Section 5.2 the flow and implementation of the layout algorithm is discussed. Section 5.3 presents a description of roles that can be found in a graph partition; then in Section 5.4 some experiments and results are presented; this section presents also some discussions about the complexity and execution times of the algorithm. Finally the conclusion is presented in Section 5.5.



(a) Example of the inner nodes location

(b) Example of the border nodes location

Figure 5.1: Example of the position of each type of node in a social network. Each color represent one group of the partition.

5.2 PRESENTATION AND DESCRIPTION OF THE ALGORITHM

The algorithm divides the set of nodes into $k+1$ subsets V_i such that:

$$V = V^{+} \cup \bigcup_{i=1}^{k} V_i^{-} \tag{5.1}$$

where V^{+} is the set of border nodes and $V_i^{-}, i = 1, 2, \ldots, k$ is the subset of inner nodes from community i. The objective of the algorithm is to place the border nodes in such a way they are differentiated from the inner nodes. The place where the border nodes will be placed is called the interaction zone and the inner nodes will be placed outside this zone as presented in Figure 5.2.

In our proposed layout the drawing zone is a unitary circle and the interaction zone is represented by a concentric circle of radius $r = 0.5$. This forms a ring shaped area where the inner nodes of each community will be paced.

Figure 5.3 presents the general process of the layout model. Once the node set is divided, a process for finding the coordinates of each type of nodes is started; first the algorithm is applied to the border nodes set and for each set of inner nodes. This is performed in such a way the nodes are placed according to a dissimilarity measure defined by the neighborhood of the nodes. These initial coordinates are centered in $(0, 0)$ and lay within the interval $(-\infty, \infty)$ for both x and y, thus in order to manipulate them, they are scaled to be within the interval $[-1, 1]$.

The coordinate integration step performs a scaling procedure to change the coordinates center of each set of inner nodes in such a way they can be placed near the border nodes of the community.

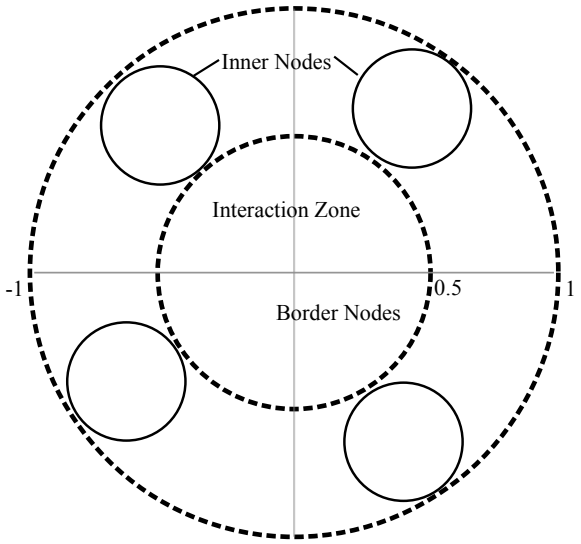


Figure 5.2: Placement of the elements of the layout model

5.2.1 DISSIMILARITY MEASURES

In the proposed model we need to measure the dissimilarity between nodes. This can be done by measuring the geodesic distance, which, according to [Wasserman and Faust, 1994], is one of the most used in the field. This distance represents the shortest path between the nodes u and v as: $d(u,v) = \min_p A^{[p]}_{uv} > 0$ where $A^{[p]}$ is the power matrix p.

However, this measure does not provide any idea about the neighborhood of the nodes. In [Melançon and Sallaberry, 2008] authors present the Jaccard distance δ_J to measure the overlapping between the neighborhood of two nodes u and v. This distance is given by:

$$\delta_J(u,v) = 1 - \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} \tag{5.2}$$

where $N(u)$ is the set of neighbors of the node u. Other measures are presented in the same work, however as reported by the authors, having similar properties, the differences between those measures are not perceptible at least at a local scope.

In general if the neighborhoods of two nodes u and v contain the same elements, the distance will be 0, i.e., the nodes will be totally similar in terms of their local structure. The goal of the algorithm is to place similar nodes near and dissimilar nodes away from each other.

Thus, given a set U of n nodes, a dissimilarity matrix $\mathbf{\Delta}_U$ is a $n \times n$ matrix defined

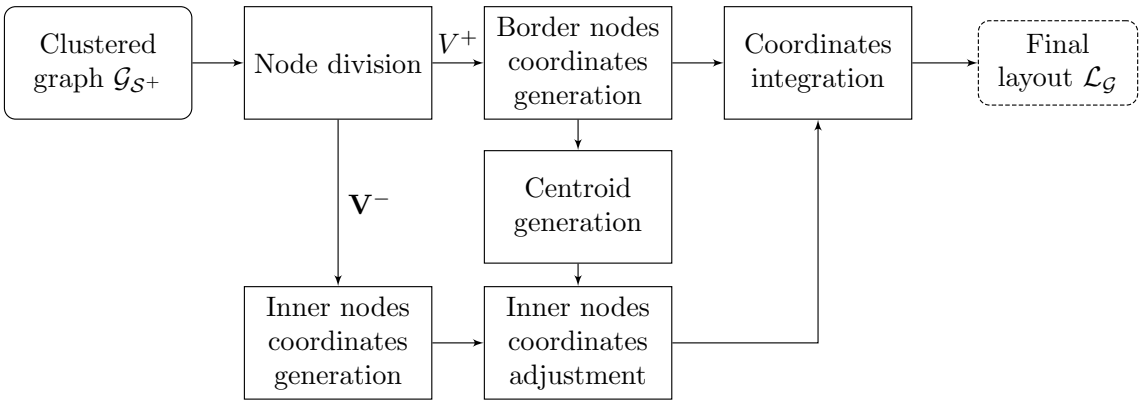


Figure 5.3: Diagram of the clustered graphs layout process

as:

$$\mathbf{\Delta}_U = \{\delta_J(i,j)\} \tag{5.3}$$

where $i, j \in U$. Since the Jaccard distance can be considered as a metric, $\mathbf{\Delta}_U$ is a symmetric matrix with zero diagonal.

This matrix will be generated for each subset of nodes after the division presented in the previous section.

5.2.2 MULTIDIMENSIONAL SCALING

Multidimensional scaling (MDS) is a technique to represent similarities or dissimilarities among objects in a space and map them as distances into another ρ-dimensional space. For a graphical representation, ρ is either 2 or 3.

There are several implementations of the MDS algorithm which can be classified, according to [Ingram et al., 2009], into three types: classic methods, stress majorization and optimization of force directed heuristics.

1. **Classic methods:** these methods search for an analytical solution by minimizing an strain function. This is made by computing a singular value decomposition of $\mathbf{\Delta}$, which is in general an $O(n^3)$ operation.

2. **Stress majorization:** these techniques use a gradient descent approach to optimize a non-linear stress function. These methods have in general a complexity of $O(n^2)$ improving the general time execution of classic approaches.

3. **Force based approaches:** these methods use a simulated spring-mass system to minimize the stress function. The basic implementation has a complexity of $O(n^3)$: $O(n^2)$ per iteration executing n iterations.

In this work we use the SMACOF (Scaling by MAjorizing a COmplicated Function) algorithm, proposed by [de Leeuw, 1977], which belongs to the second category, and which according to [Ingram et al., 2009], has the most approximate results to the classic MDS.

In order to measure the proximity between the distances and the dissimilarities a stress function is used. This function $\sigma(\mathbf{X})$ is given by:

$$\sigma(\mathbf{X}) = \sum_{i<j} w_{ij} (d_{ij} - \delta_{ij})^2 \tag{5.4}$$

where $\mathbf{X}$ is the point matrix, d_{ij} is the Euclidean distance and δ_{ij} is the dissimilarity between objects i and j. SMACOF is outlined in the Algorithm 5. The matrix $\mathbf{B}(\mathbf{Z})$ used in the algorithm is given by:

$$\begin{aligned} b_{ij} &= \begin{cases} -\frac{w_{ij}\delta_{ij}}{d_{ij}} & \text{for } i \neq j \text{ and } d_{ij} \neq 0 \\ 0 & \text{for } i \neq j \text{ and } d_{ij} = 0 \end{cases} \\ b_{ii} &= -\sum_{j=1, j\neq i}^{n} b_{ij} \end{aligned} \tag{5.5}$$

where $w_{ij} \geq 0$ is the weight between i and j. The weights w_{ij} are used in SMACOF algorithm to handle the non-existence of some points, reducing in this way their impact on the stress function σ; in this case all the weights are set to 1.

SMACOF algorithm will run until the difference between the stress from two consecutive iterations is less that some predefined $0 < \epsilon < 1$.

Since SMACOF uses a gradient descent approach, initial guess of the points in $\mathbf{X}$ may change the final result. This means that the solution is a local minima depending on the starting point. This may pose problems to the user due to the impossibility of reproducing results. [Borg and Groenen, 1997] recommend that the centroid initial coordinates should be $(0, 0)$.

Hence, a matrix $\mathbf{X}^{[0]}$ is defined to contain the initial coordinates of the nodes to be placed over the perimeter of a circle of radius $r = 1$ and center $c = (0, 0)$. The position of each node is:

$$\begin{aligned} \mathbf{X}_x^{[0]}(i) &= \cos\theta_i \\ \mathbf{X}_y^{[0]}(i) &= \sin\theta_i \end{aligned} \tag{5.6}$$

Data: $\boldsymbol{\Delta}$
Result: $\mathbf{X}$
$\mathbf{X}^{[0]} \leftarrow$Initial guess ; /* Eq. 5.6 */
$\mathbf{Z} \leftarrow \mathbf{X}^{[0]}$;
$\sigma^{[0]} \leftarrow \sigma\left(\sigma\left(\mathbf{X}^{[0]}\right)\right)$;
$k \leftarrow 0$;
repeat
  $k \leftarrow k + 1$;
  $\mathbf{X}^{[k]} \leftarrow \frac{1}{n}\mathbf{B}\left(\mathbf{Z}\right)\mathbf{Z}$;
  $\mathbf{Z} \leftarrow \mathbf{X}^{[k]}$;
  $\sigma^{[k]} \leftarrow \sigma\left(\sigma\left(\mathbf{X}^{[k]}\right)\right)$;
until $\sigma^{[k-1]} - \sigma^{[k]} < \epsilon$;
return $\mathbf{X}^{[k]}$

Algorithm 5: Basic implementation of the SMACOF algorithm

where $\theta_i = (i-1)\,\frac{2\pi}{k}$ and $i = 1, 2, \ldots, k$.

By this way the initial points will have the same coordinates at each execution, as advised by [Borg and Groenen, 1997], making the algorithm to obtain the same final coordinates.

5.2.3 PLACING THE BORDER NODES V^+

These nodes are in charge of the interaction between communities, hence they should be placed near all the communities. To do it, this set will be placed at the center of the drawing, surrounded by the inner nodes of the communities (5.2).

The process starts calculating the dissimilarity matrix Δ_{V^+}, which represents the dissimilarities between all the border nodes. This matrix is used then as input for the SMACOF algorithm which will find a coordinate for each node in the set. These coordinates are produced in such a way that are close to the provided similarities.

The function *scale_and_center* will place the nodes inside a circle of radius 0.5 and center $c = (0, 0)$.

The algorithm 6 shows the process to place the border nodes. This set is taken as a whole and contains nodes from all the communities. Thus, the idea is to show the relationships between communities according to the position of each node.

Since the dissimilarity function is calculated from the neighborhood structure of each node, a proximity in the layout reflects a structural proximity in the graph G.

Data: V^+
Result: $\mathbf{X}_{V^+}$
1 $\Delta_{V^+} \leftarrow d_J\left(\forall u, v \in V^+\right)$; /* Eq. 5.2 */
2 $\mathbf{X}_{V^+} \leftarrow$SMACOF($\Delta_{V^+}$);
3 $\mathbf{X}_{V^+} \leftarrow$ scale_and_center $\left(\mathbf{X}_{V^+}, c\left(0,0\right)\right)$;
4 **return** $\mathbf{X}_{V^+}$

Algorithm 6: Border Nodes Set Layout

The algorithm returns the x and y coordinates of each node. These coordinates $\mathbf{X}_{V^+}$ are used in the next step to calculate the position of nodes in the inner nodes sets to place them in such a way they are placed close to the nodes of their own community in the interaction zone.

5.2.4 PLACING THE INNER NODES OF EACH COMMUNITY V_i^-

These are the nodes of each community i which interact only with nodes within their own community. Thus, each subset is placed in front of the border nodes of the same community. To do this the center $\mathcal{P}_i$ of the community is given by:

$$\begin{aligned} \mathcal{P}_i^x &= \sqrt{\frac{1}{m_i^2+1}} \times r \\ \mathcal{P}_i^y &= \mathcal{P}_i^x \times m_i \end{aligned} \tag{5.7}$$

where

$$r = \begin{cases} 0.75 & \text{if } \bar{x}_i > 0 \\ -0.75 & \text{if } \bar{x}_i < 0 \end{cases}$$

and m_i is the slope of the vector formed by the centroid of the community border nodes i and the origin. Thus, $m_i = \frac{\bar{y}_i}{\bar{x}_i}$, where $\bar{x}_i$ and $\bar{y}_i$ are:

$$(\bar{x}_i, \bar{y}_i) = \sum_{u \in C_i \cap V^+} \frac{u_x, u_y}{|C_i \cap V^+|} \tag{5.8}$$

where u_x and u_y are the coordinates x, y of the node u.

Therefore, each of those points is the centroid of its respective community and will be used by the algorithm 7 to place each group.

The algorithm finds the position of each inner node from each community. At this point the coordinates are centered in the origin and in the interval $[-1, 1]$ for both x and y. So in the next step the coordinates of the nodes in each community have to be

Data: $\mathbf{V}^-, \mathcal{P}$
Result: $\mathbf{X}_{\mathbf{V}^-}$
for $V_i^- \in \mathbf{V}^-$ **do**
  $\Delta_{V_i^-} \leftarrow d_J\left(\forall u, v \in V_i^-\right)$ (Eq. 5.2);
  $\mathbf{X}_{V_i^-} \leftarrow$SMACOF($\Delta_{V_i^-}$);
  $\mathbf{X}_{V_i^-} \leftarrow$ scale_and_center $\left(\mathbf{X}_{V_i^-}, \mathcal{P}_i\right)$;
end
return $\mathbf{X}_{V^-}$

Algorithm 7: Inner Nodes Sets Layout

transformed to be centered on the point defined by equation 5.8 and normalized to fit into the annulus of inner radius 0.5 and outer radius 1. This is performed by the function *scale_and_center* whose input is a set of coordinates and the centroid of the group.

Similar to the border nodes placement algorithm, for each inner node set a dissimilarity matrix is calculated. This matrix reflects the dissimilarities of each inner nodes set individually, i.e., there are no elements from other communities in these matrices.

Once the dissimilarity matrix is calculated, the initial coordinates are calculated using SMACOF, then these coordinates are scaled and centered in the provided centroid.

The algorithm 8 summarized the whole layout process. The whole process hence produce a layout $\mathcal{L}_\mathbf{C}$ that contains the coordinates for each node in V.

Data: A partition $\mathbf{C}$
Result: A layout $\mathcal{L}_\mathbf{C}$
$V^+ \leftarrow \bigcup v^+ \in \mathbf{C}$;
$V^- \leftarrow \bigcup v^- \in \mathbf{C}$;
$\mathbf{X}_{\mathbf{V}^+} \leftarrow$Border Nodes Set Layout($V^+$) {Alg 6};
foreach $V_i^+ \in V^+$ **do**
  $\mathcal{P}_i \leftarrow$ centroid_calculation $(i, \mathbf{X}_{\mathbf{V}^+})$; `/* Using equation 5.8 */`
end
$\mathbf{X}_{\mathbf{V}^-} \leftarrow$Inner Nodes Sets Layout($\mathbf{V}^-, \mathcal{P}$) {Alg 7};
return $\mathcal{L}_\mathcal{G}$

Algorithm 8: Layout algorithm

Figure 5.4 presents the position of the nodes for the final layout $\mathcal{L}_\mathcal{G}$. In the center of the drawing, in the interaction zone, there are the border nodes and outside this zone, the inner nodes from each community.

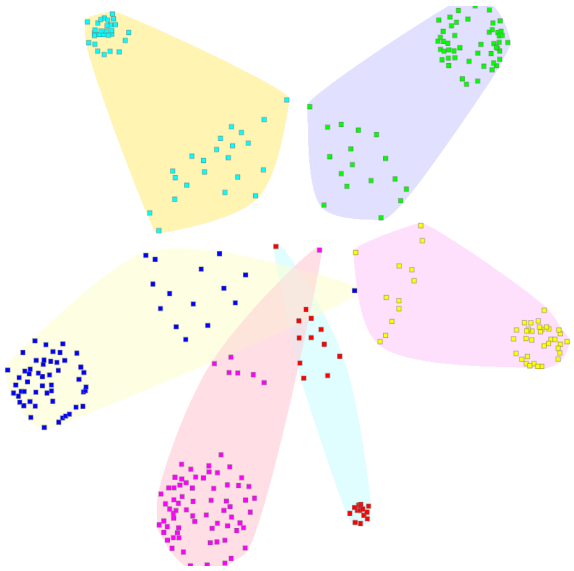


Figure 5.4: Location of the inner nodes according to the placed border nodes

Each community in the Figure 5.4 is inside a convex hull. Note that the inner nodes are placed in front of the border nodes of their respective community.

5.2.5 COMMUNITIES LAYOUT ALGORITHM COMPLEXITY

The complexity of the visualization algorithm, which is executed $k+1$ times, where k is the number of communities in the partition, makes the execution time to vary according to the size of each category (border nodes and inner nodes.)

The SMACOF algorithm iteratively reduces a stress function to find a set of points mapping dissimilarities among objects. The most costly operation of SMACOF is the matrix multiplication its basic calculation has a complexity of $T(n) = O(n^2 L)$, where L is the target dimension of the points, in this case, $L = 2$. However, since in this algorithm we divide the nodes into different groups, the complexity is not the same.

The complexity values $T_B(n)$ and $T_I(n)$ for the algorithm 6 and the algorithm 7 respectively are given by:

$$T_B(n) = O\left(2|V^+|^2\right), \quad T_I(n) = O\left(\sum_{i=1}^{k} 2|V_i^-|^2\right)$$

If $\bigcup_{i \in \mathcal{C}} V_i^- \neq \emptyset$ then the expected size of each inner node set is $\frac{n-|V^+|}{k}$, thus the complexity is:

$$T_I(n) = O\left(\frac{(n-2|V^+|)^2}{k^2}\right)$$

where k is the number of communities. Therefore, the overall complexity is:

$$\hat{T}(n) = O\left(\frac{(n-2|V^+|)^2}{k^2} + 2|V^+|^2\right) \tag{5.9}$$

Note that $\hat{T}(n) < T(n)$. Nonetheless, the worst case is when $\bigcup_{i\in\mathcal{C}} V_i^- = \emptyset$, i.e., all the nodes belong to the border nodes set. In this case, the complexity is $O(2n^2)$.

This particular case is not very probable in the social networks context. According to [Newman and Girvan, 2004], a social network has a community structure which means that the number of edges within communities is notably greater than the edges between communities.

The most expensive step of the algorithm, in terms of computational cost, is the matrix multiplication. For that matter all the matrix multiplication operations were done using an implementation of the level 3 BLAS [Dongarra et al., 1990] which is able to perform calculations using multi-core architectures when available.

To test the execution of the algorithm and the way it grows when the number of border nodes is increased, the algorithm was tested using a graph extracted from Twitter, with 5389 nodes and 46440 edges and divided into 12 groups using the fast unfolding algorithm.

The experiment starts by assigning all the border nodes to inner nodes in their respective communities, then, iteratively, nodes from all communities are transformed into border nodes and the new execution time is calculated. This is done until all the nodes in the graph are border nodes, having the maximum execution time.

Figure 5.5 shows the execution time growth according to the number of border nodes set size. The worst execution time, as expected according to equation 5.9, occurs when all the nodes belong to the border nodes set. Note that the time is effectively increased following a quadratic growth, but it has a low execution time when the number of border nodes is low. For this graph, clustered using the fast unfolding algorithm, the original percentage of border nodes is near 24% of the total number of nodes.

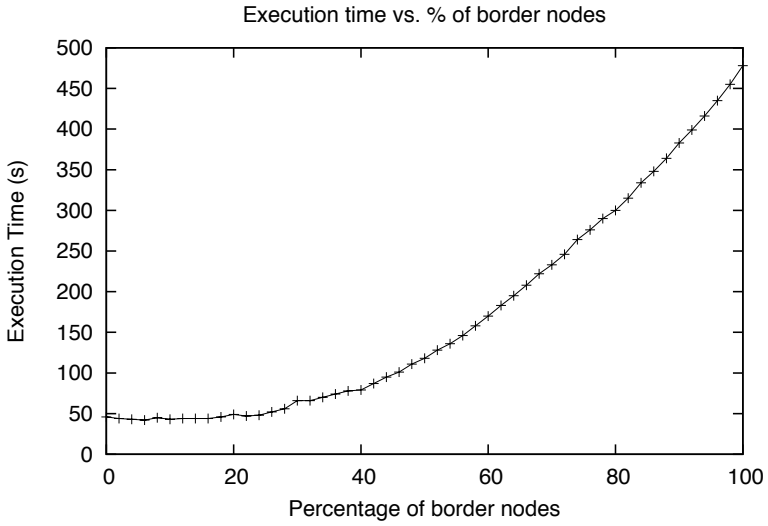


Figure 5.5: Execution time for different border nodes set size

5.3 ROLES IN COMMUNITY NETWORKS

One important aspect of social network analysis is measuring the importance of some actors in the network. This importance can be referred to the capacity of some nodes to disconnect two or more communities, or to concentrate the flow of messages between different teams. This aspect has been studied in both academic and industrial scenarios like those presented by [Aldrich and Herker, 1977], by [Guimera and Amaral, 2005] and by [Cross and Parker, 2004].

In the work of [Aldrich and Herker, 1977], authors present, from a sociological and managerial perspective, the boundaries of an organization as a defining characteristic of it. They define the boundaries as the organization's points of contact with its environment.

Thus actors with a boundary role may perform two functions, one of representation of the organization, and other of processing incoming information; the first one to organization's responses to environmental changes and inquiries. The second one is a sort of filter of the incoming information in that way the information do not overwhelms the organization. All these concepts can be extrapolated to a community graph. In this case each community can represent an organization, and the environment can be represented by the interaction between communities.

On the other hand, [Cross and Parker, 2004] present a set of roles concerning existing groups in a company. They propose four basic roles:

- **Central nodes**, which have a high degree i.e., most of the information in the network passes through them.
- **Boundary spanners**, which provide critical links between two or more communities.
- **Information brokers**, which can be seen as experts in some domain and, for example help to leverage expertise in a corporate network.
- **Peripheral people**, those persons who do not interact with other groups, or who are not central regarding the configuration of the network. Into this class are located those nodes without any connection, i.e., degree 0.

These roles hence can describe in more detail the nodes in a community graph. These previous definitions have been made from a managerial context and they are not formal in a mathematical sense. [Guimera and Amaral, 2005] have proposed a set of seven roles according to the position and interaction volume of the nodes in a community graph.

To do that they define two indices: the within-module degree z score and the participation coefficient. The z score is a measure of how well connected is a node; thus, given a node $u \in C_i$, the $z-$score of the node u is given by:

$$z_u = \frac{|N^-(u)| - \mu_{d_{C_i}}}{\sigma_{d_{C_i}}} \tag{5.10}$$

where $|N^-(u)|$ is the number of inner neighbors of u, $\mu_{d_{C_i}}$ is the average degree in community C_i and $\sigma_{d_{C_i}}$ is the standard deviation of the degree in community C_i. This score measures how well connected is the node u to other nodes in the graph.

The second measure, the participation coefficient P_u measures the type of node regarding its connections to other communities. This coefficient is defined by:

$$P_u = 1 - \sum_{i=1}^{k} \left(\frac{|N(u, C_i)|}{|N(u)|} \right)^2 \tag{5.11}$$

where $|N(u, C_i)|$ is the number of edges from the node u to the community C_i and $|N(u)|$ is the total degree of u. P_u hence will vary from 1 if the neighbors of u are uniformly distributed among the k communities, to 0 if all the neighbors of u are in its own community.

Using these two indices, [Guimera and Amaral, 2005] define seven roles according to the value obtained by each node, the first four for nodes with $z < 2.5$ and the last three for nodes with $z \geq 2.5$.

- **Role 1:** ultra peripheral nodes. Nodes with $P \approx 0$.
- **Role 2:** peripheral nodes. Nodes with at least 60% of inner neighbors. $P \in (0, 0.62)$.
- **Role 3:** non-hub connectors. Nodes with a degree greater than 4 and a 50% of inner neighbors. $p \in [0.62, 0.8)$.
- **Role 4:** non-hub kinless nodes. Nodes with less than 35% of inner neighbors, this is $P \in [0.8, 1]$.
- **Role 5:** provincial hubs. A node with a degree $\gg 1$ which has at least $5/6$ of inner neighbors; $P \in [0, 0.3)$.
- **Role 6:** connector hubs. A node with a degree $\gg 1$ which has at least 50% of inner neighbors. $P \in [0.3, 0.75)$.
- **Role 7:** kinless hubs. A node whose has less than 50% of its neighbors in its own community. $P \in [0.75, 1]$.

These roles give an idea about the distribution of roles within the community graph and how to measure their importance.

Aldrich and Herker	Cross and Parker	Guimerà and Amaral	Type
N/D	Peripheral people	Ultra peripheral nodes	I
		Peripheral nodes	B
	N/D	Non-hub connectors	B
	N/D	Non-hub kinless nodes	B
Boundary spanners type I & II	Central nodes	Provincial hubs	I, B
	Spanner nodes	Connector hubs	B
	Information brokers	Kinless hubs	B

Table 5.1: Comparative summary of roles definitions in community networks

Table 5.1 presents a comparative summary of the roles that can be found analyzing a community graph. These roles, in general, help the researcher to find important actors regarding the communication between communities rather than identifying, for example structural holes. The last column indicates the type of node (I for inner nodes, B for border nodes) that can hold the respective role. Most of the roles are hold by border nodes, hence appearing within the interaction zone. The roles assigned to inner nodes are those that can have participation indices equal to 0.

The set of roles defined by Guimerà and Amaral will be used during the experiments in next sections.

5.4 EXPERIMENTS OF COMMUNITY GRAPH LAYOUT ALGORITHM

The goal of the experiments is to test the algorithm in terms of scalability and in terms of its ability to separate the nodes involved with interactions from those who do not interact with other groups and then identify important nodes according to the division presented the third column in Table 5.1.

Graph	**Nodes**	**Edges**	δ_G	**Communities**	Q	**Time(s)**
Protein interaction network	19928	82406	4.150×10^{-4}	56	0.6493	1021
DBLP co-authorship network	10000	65734	1.315×10^{-3}	843	0.8324	346
Twitter network	5389	46440	3.199×10^{-3}	12	0.5728	89
Facebook network	334	5394	9.7×10^{-2}	6	0.7728	36

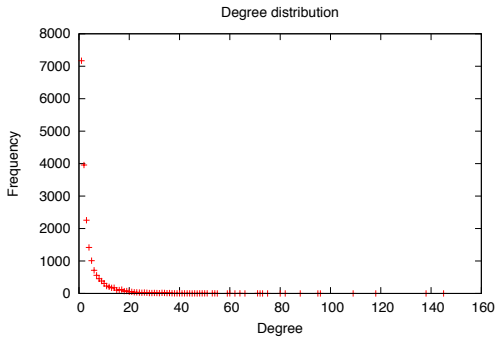
Table 5.2: Description of the datasets used for testing the layout algorithm

The graph density value δ_G presented in the table shows the proportion of edges and nodes; in general the social networks, according to [Fortunato, 2010] have a low density, that means it follows a power law distribution of the degree, or at least contains a community structure. These graphs have a modularity value that indicates that they have a community structure; this condition can be observed with the degree distribution of each one. Figure 5.6 presents the degree distribution for each graph. The degree distribution of each graph shows in general a power law distribution, indicating the scale-free nature of these networks.

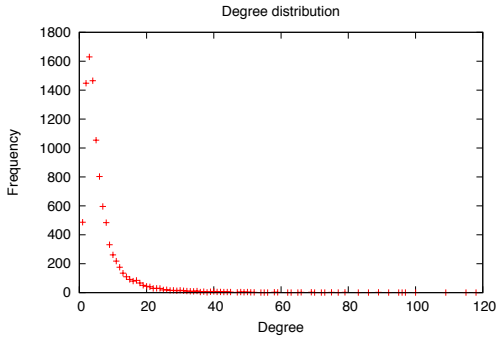
Thus, using these graphs and their partitions, obtained using the fast unfolding algorithm by [Blondel et al., 2008], it is possible to test the proposed layout algorithm. For each graph, first the distribution of roles will be presented, then the identification of the nodes with those roles will be performed over the layout.

The last column indicates the time used by the algorithm to complete the layout. All the execution were performed using an Intel Xeon Quad core 2.33GHz, 4Gb of RAM running under openSuSE 11.4 and the implementation was made in C++.

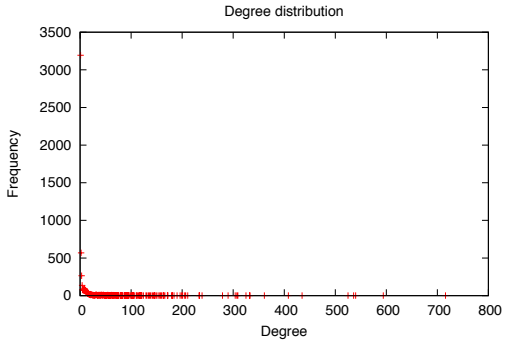
Table 5.3 shows the distribution of border and inner nodes of each one of the partitions used during experimentation. As presented in previous section, the percentage of border nodes of the network is expected to be low because of the scale-free structure of the networks.



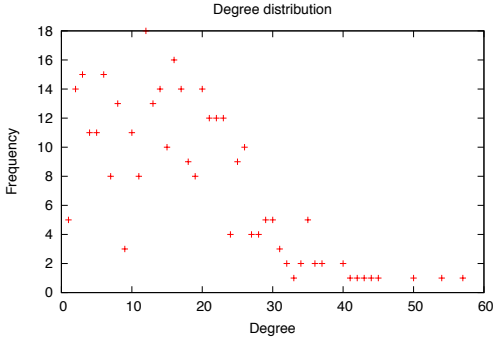
(a) Distribution of the degree for the protein interaction graph



(b) Distribution of the degree for the DBLP coauthorship graph



(c) Distribution of the degree for the Twitter graph



(d) Distribution of the degree for the Facebook graph

Figure 5.6: Degree distribution for the graph used during experimentations

5.4.1 EXPERIMENTS WITH THE PROTEIN INTERACTION NETWORK

The dataset *dip20090126*, is the result of the experimental identification of protein interactions [Sommer, 2010]. This set is interesting because, even if it is not a social network, it can be classified as complex networks presenting the similar properties to a social network.

With the partition obtained using the fast unfolding algorithm the relevance indices defined by equations 5.10 and 5.11 are calculated for each node; the resulting indices are presented in the Figure 5.7.

There are six roles present in the graph, four of non-hub type and two of hub type. The non-hub type are nodes which concentrate a low number of connections with their respective communities even if their interaction with other communities is not negligible.

Graph	Border nodes	Inner nodes	Ratio
Protein interaction network	7592	12336	38.09%
DBLP co-authorship network	4013	5987	40.13%
Twitter network	1305	4084	24.22%
Facebook network	85	249	25.45%

Table 5.3: Distribution of border/inner nodes of the partition

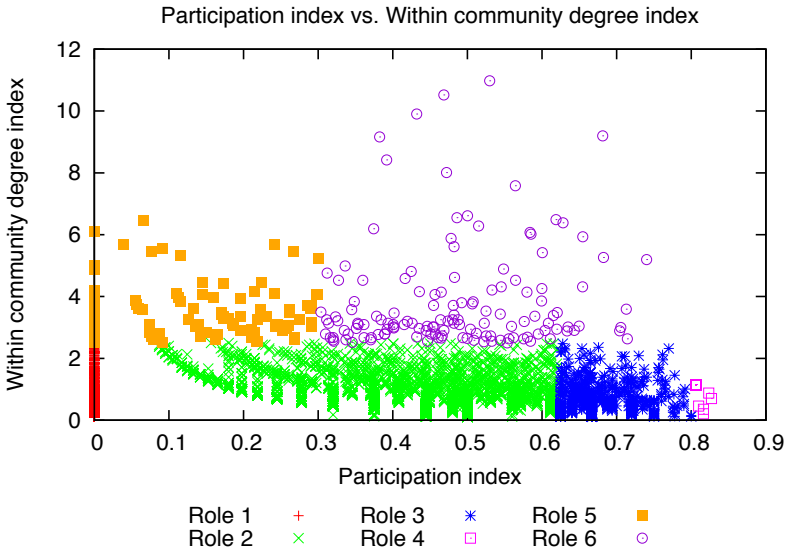


Figure 5.7: Plot of the participation index P vs the within community $z - score$ for the protein interaction dataset

The Role 1 hence correspond to the $\approx 62\%$ of the nodes; this role contains all the nodes which interactions remain within their same community: these are mostly inner nodes. Note that these nodes have a participation index of 0.

Roles 2 and 3 represent nodes with a low degree and higher participation index, i.e., these nodes connect several communities but still have a low degree to be considered as hubs. Role 4 is less common: according to [Guimera and Amaral, 2005] nodes with this role are not found in most real-world networks; this role is for nodes connected to almost all communities in the graph but with a low degree.

Roles 5 and 6 represent nodes with a higher number of connections with their communities than the average, thus they can be considered as hub nodes and their participation index is important.

Each node in the graph can be assigned to one of the 7 roles (but there are only 6

roles in this network), however in this work we will focus on hub roles, i.e., Roles 5, 6 and 7. These roles are interesting since they concentrate an important number of inner neighbors and a possibly high rate of participation; note that nodes from Role 5 can be also inner nodes.

Figures 5.8 and 5.9 shows two layouts of the same graphs of communities: in Figure 5.8 using Fruchterman & Reingold, and in Figure 5.9 using the algorithm proposed in this chapter. In both figures the bigger nodes correspond to nodes assigned to roles 5 and 6.

In the first case can be observed four clusters of nodes, two circular, with nodes from several communities, and two elongated with nodes, primarily from two communities. The way that the circular clusters are displayed prevents from noticing all the highlighted nodes.

That superposition of nodes is reduced using the proposed algorithm, as presented in Figure 5.9. In this case the nodes from roles 5 and 6 can be noticed easier than in the first case. As expected, those nodes in these roles, with a participation index greater than zero are placed inside the interaction zone, while those with $P = 0$ are located in their respective communities.

Another advantage of this layout is that the location of nodes is given by the similarity of their neighborhoods: if two nodes are drawn close to each other, then they have similar neighbors. Note that in Figure 5.9 there is a gap between some nodes and in the center of the interaction zone; this gap represents a similar structural division of the nodes as in the Fruchterman & Reingold algorithm layout.

5.4.2 EXPERIMENTS WITH THE CO-AUTHORSHIP NETWORK

This graph is a network of 10000 nodes and 65734 edges representing a co-authorship network. In this network it is expected to have a large number of interactions due to the way it has been created.

Once clustered, this network contains 843 communities with 4013 border nodes and 5987 inner nodes. Similar to the protein interaction network, this one contains nodes assigned to Role 4, that represent nodes with several connections to other communities, but with a low degree.

As presented in Figure 5.10, this network contains nodes from all roles. In this case there is one node assigned to Role 7. This role is for nodes with a high degree and have connections to several different communities.

Thus, in this case the idea is to analyze those nodes with hub roles: 5, 6 and 7, and to observe them in respect of their position in the graph. This analysis will help to, visually, identify each node and its connections to different communities.

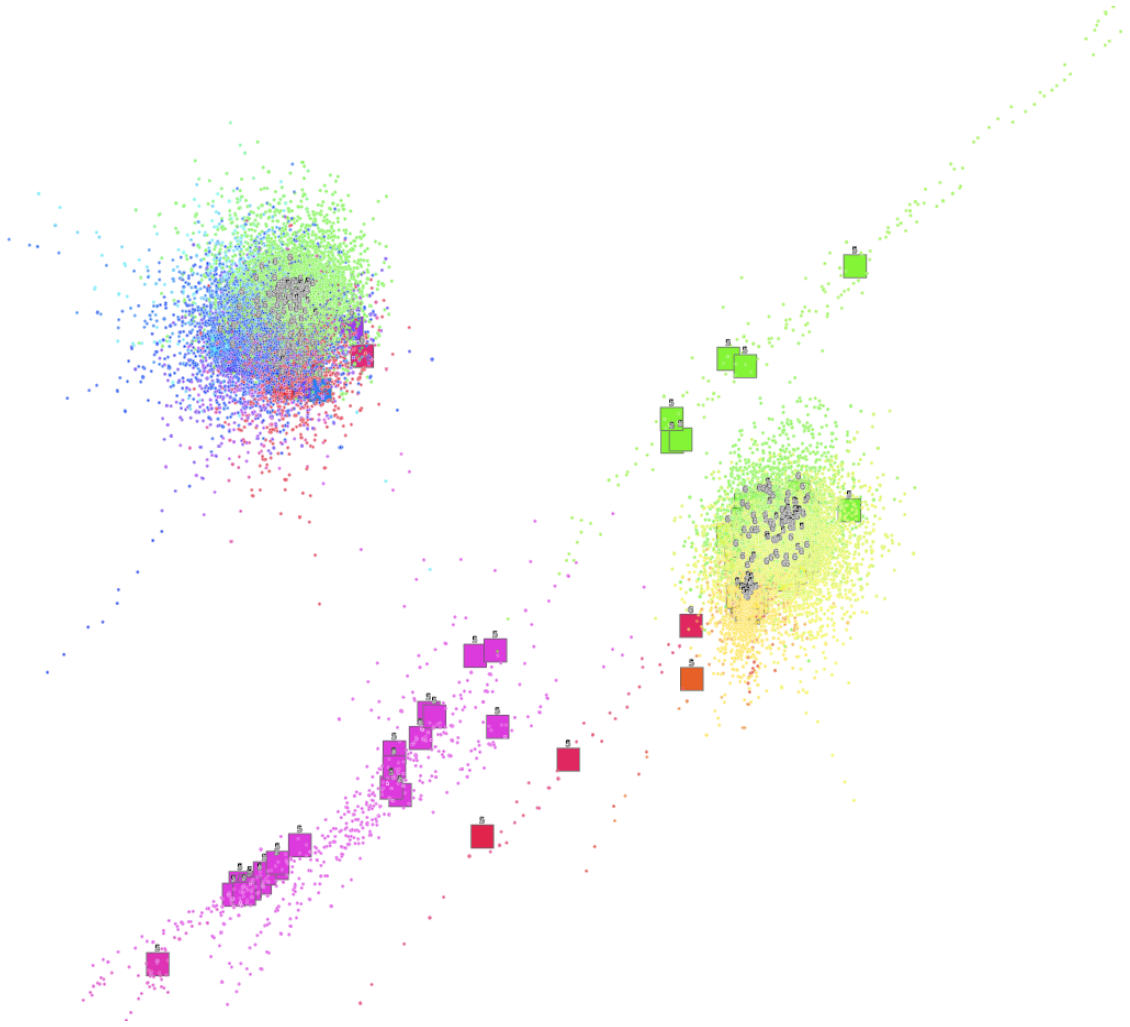


Figure 5.8: Layout of the graph using Fruchterman & Reingold for the *dip20090126* protein interaction network dataset. To reduce the number of drawn elements, the edges are not shown

Figures 5.11 and 5.12 presents two layouts for this graph. In 5.11, the layout produced by the Fruchterman & Reingold algorithm and in Figure 5.12 the result using the layout proposed in this chapter.

The layout produced by the FR algorithm makes the nodes to be concentrated near the center of the drawing, making difficult the identification of the interactions and roles.

Using the interaction centered algorithm the nodes are placed in such a way the different communities are identifiable and so the different roles present in the graph are.

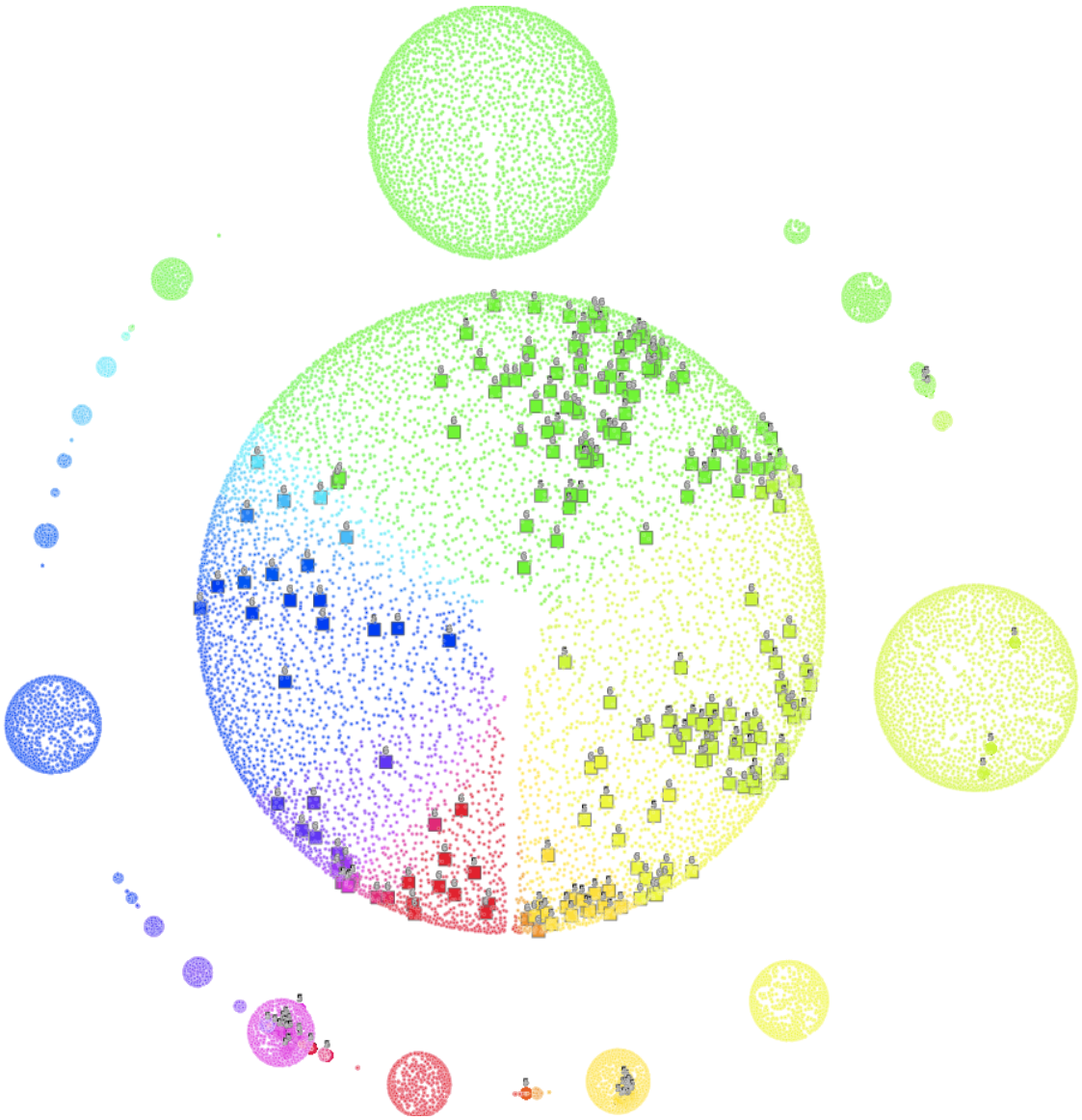


Figure 5.9: Layout of the graph using the our algorithm for the *dip20090126* protein interaction network dataset. To reduce the number of drawn elements, the edges are not shown

For example, in this network there is one node with the role 7; the size of this node has been increased to show its location.

In both drawings this node is located near the center, however in the Figure 5.12 the

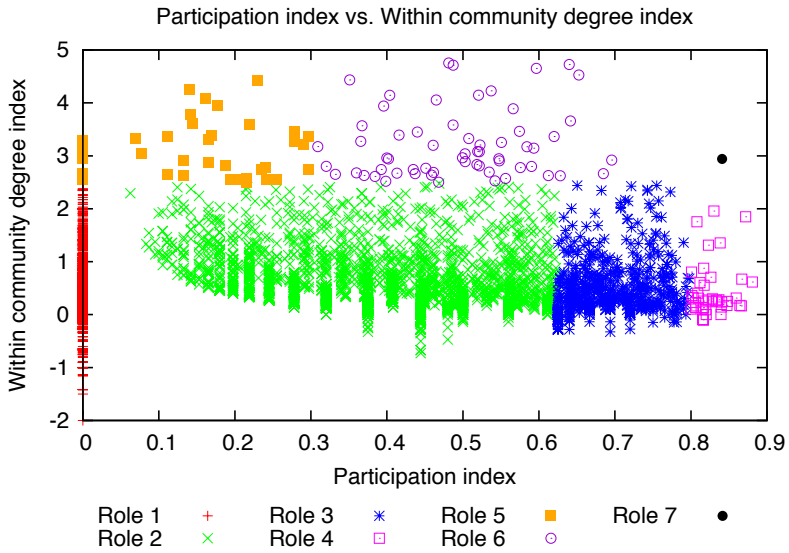


Figure 5.10: Plot of the participation index P vs the within community $z - score$ for the co-authorship network dataset

position of the node and its surrounding nodes are cleaner, allowing to identify neighbors and connections.

5.4.3 EXPERIMENTS WITH THE TWITTER NETWORK

This is a graph extracted from Twitter using a breadth first search crawler. This network represents the connections between followers and people being followed. One of the features of Twitter is the fact that there are several users largely followed by other people; these users may be celebrities, opinion leaders, politicians or even whole organizations such as governmental institutions or radio stations for example.

The distribution of roles for this graph is presented in Figure 5.13. In this case there are three non-hub roles and two hub roles. Note that nodes with roles 5 and 6 present a high within community degree index, and particularly one node with the Role 5 is far from other nodes in this role. This node hence has an important number of connections within its own community and additionally has connections to other communities.

Since the participation index of this node is greater than zero, it should be placed inside the interaction zone, maybe playing an important role as articulation point between its community and the others. To identify the nodes with important roles it is possible to use the layout of the graph allowing to observe the placement of those nodes.

In this graph one relevant node is that with Role 5 and a highest $z-$index.The size of

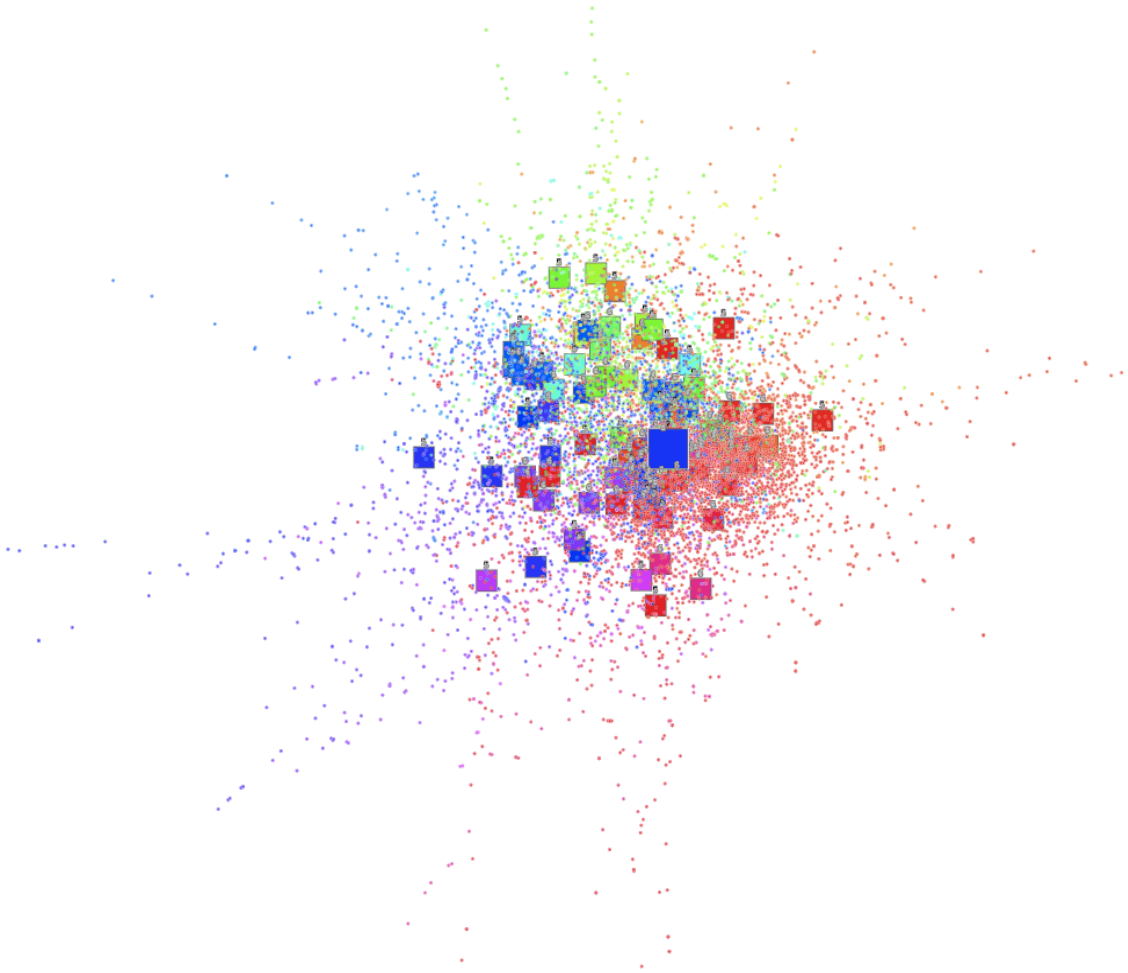


Figure 5.11: Layout of the graph using Fruchterman & Reingold for the DBLP dataset. To reduce the number of drawn elements, the edges are not shown. The bigger node in both sub-figures represents the node with role 7

this node has been increased in Figures 5.14 and 5.15; in both layouts this node is placed far from the center. This reflect the fact of the reduced participation index contrasting the high $z-$index.

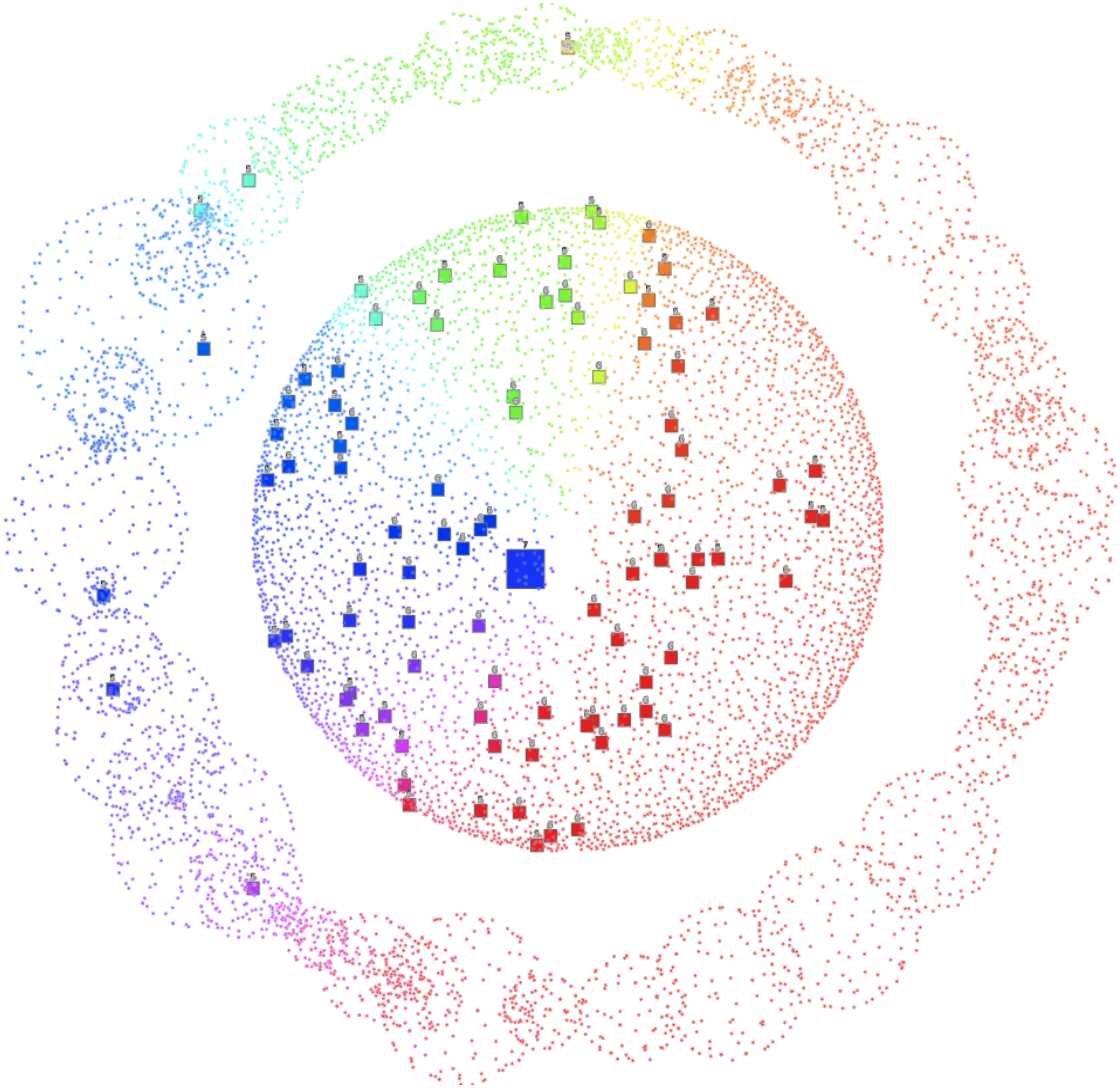


Figure 5.12: Layout of the graph using the presented algorithm for the DBLP dataset. To reduce the number of drawn elements, the edges are not shown. The bigger node in both sub-figures represents the node with role 7

5.4.4 Experiments with the Facebook network

This is a small network representing the connections between friends in Facebook; this is the same network used in the clustering experiments in Section 4.4.4. The clustering was performed using the fast unfolding algorithm.

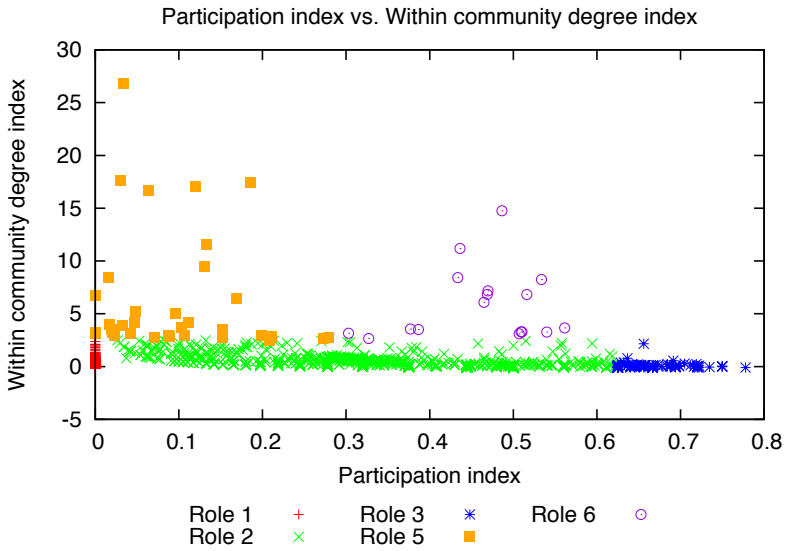


Figure 5.13: Plot of the participation index P vs the within community $z - score$ for the Twitter dataset

This graph contains 85 inner nodes and is divided into 6 communities. The distribution of roles of this network is presented in Figure 5.16. In this network the only hub role is the Role 5; there are 4 nodes (with a $P-$index greater than 0) in the interaction zone.

These four nodes with Role 5 and $P \neq 0$ have been highlighted in the drawings presented in Figures 5.17 and 5.18.

Figure 5.18 presents the layout of the graph using the proposed algorithm. Note that in this case the node in the interaction zone have been placed near the limit of that zone, leaving a higher blank space near the center of the drawing. This means that, despite the existence of nodes interacting with several communities, this number is not very high compared to their inner degree. Thus, the drawing reflects the fact that the only hub role existing in the graph is number 5.

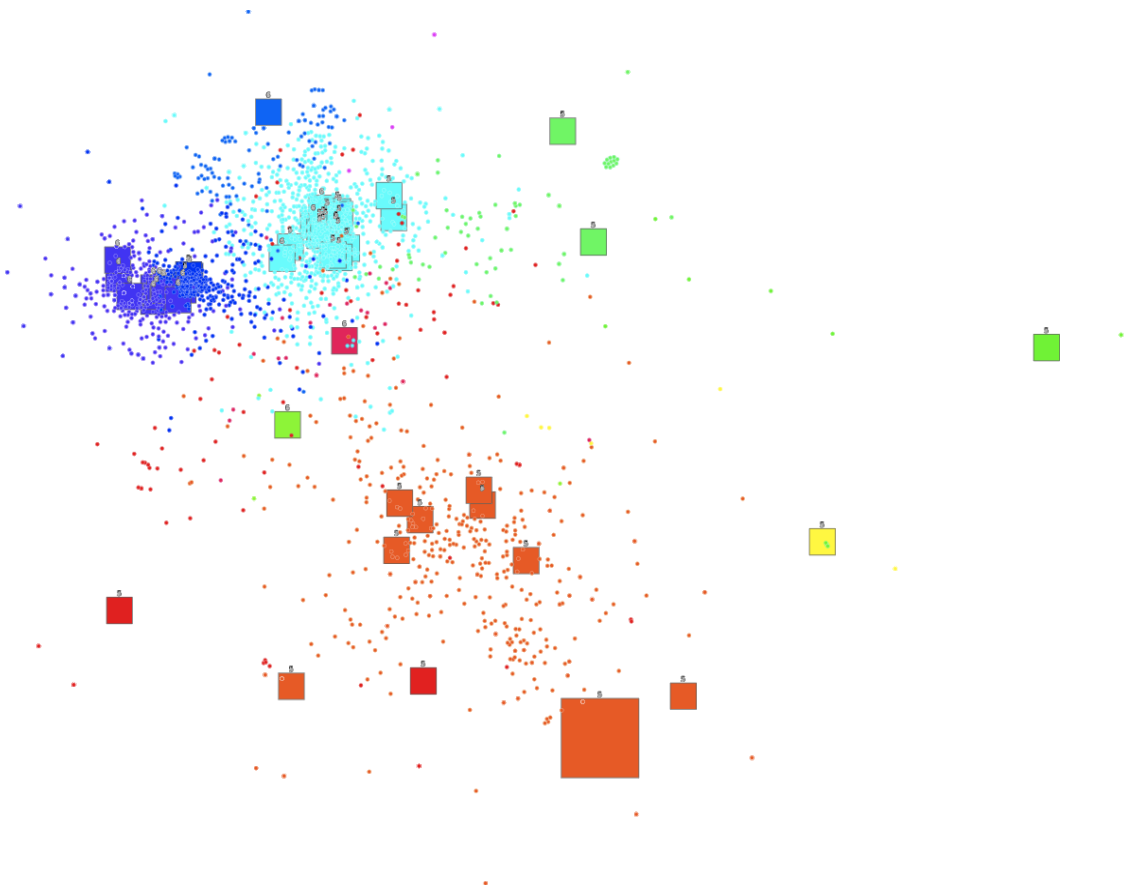


Figure 5.14: Layout of the graph using Fruchterman & Reingold for the Twitter dataset. To reduce the number of drawn elements, the edges are no shown. The bigger node represents the node with the highest z index in the partition

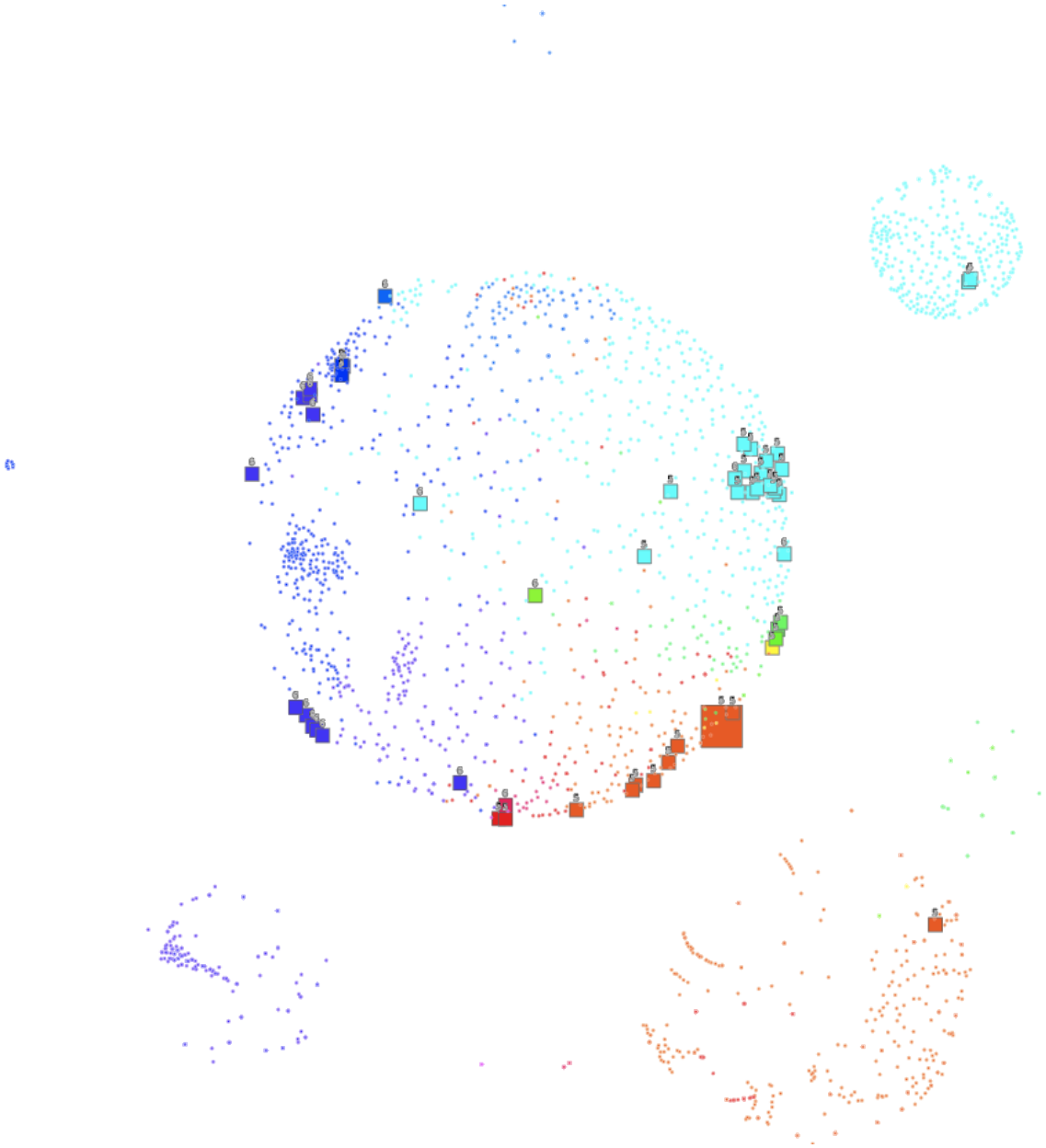


Figure 5.15: Layout of the graph using the presented algorithm for the Twitter dataset. To reduce the number of drawn elements, the edges are no shown. The bigger node represents the node with the highest z index in the partition

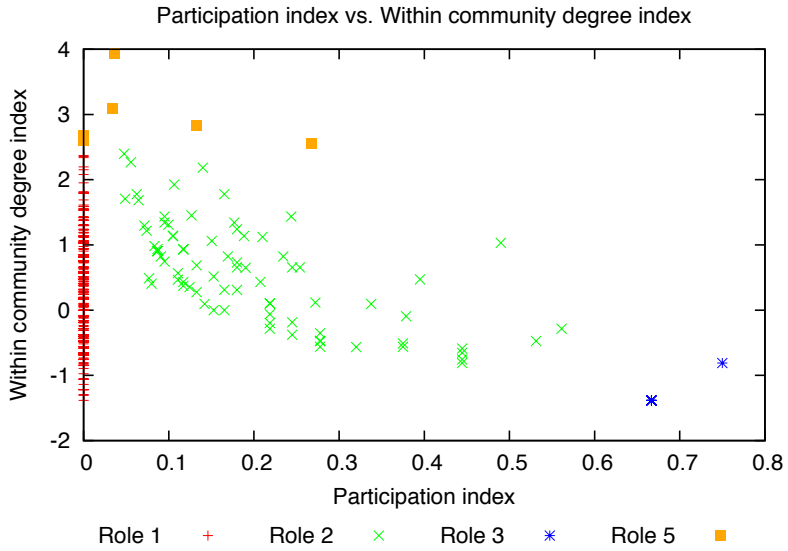


Figure 5.16: Plot of the participation index P vs the within community $z - score$ for the Facebook network dataset

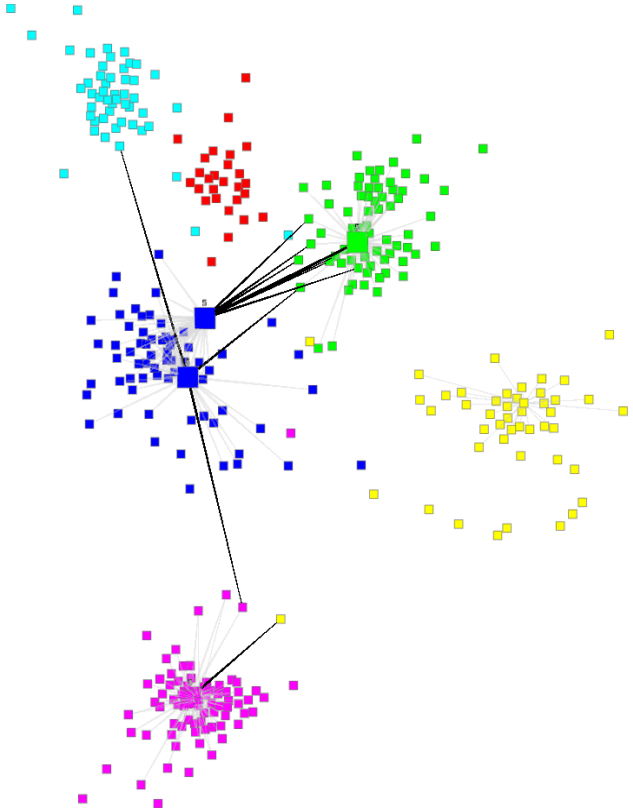


Figure 5.17: Layout of the graph using Fruchterman & Reingold for the Facebook personal network dataset. The bigger nodes represent the provincial hubs of the network (role 5)

Figure 5.18: Layout of the graph using the presented algorithm for the Facebook personal network dataset. The bigger nodes represent the provincial hubs of the network (role 5)

5.5 CONCLUSION

A novel approach for laying out clustered graphs was presented in this chapter. This algorithm allows separating the nodes involved in the community interactions from those node whose interactions remain within their respective communities. The nodes with connections to different communities are called border nodes, while the other nodes are called inner nodes.

The border nodes set contains nodes from all communities, whereas each community has a set of inner nodes. Each one of these sets are treated separately. For each one a similarity matrix is calculated and the position of each node is determined.

The similarity matrix is calculated using the Jaccard coefficient, that gives an idea of how similar two nodes are based on the projection of their respective neighborhood, i.e., the percentage of neighbors they have in common. Additionally to the structural similarity of the nodes, the similarity measure is designed to include the compositional similarity of the nodes. Thus, using Multidimensional Scaling – MDS, the position of the nodes from each subset of nodes is found in such a way that the distances between the node coordinates are near, or have some resemblance to the similarities defined for a particular subset of nodes.

At this point the coordinates are in a raw state: all the points are centered in the origin, making them to be a big blob of nodes having only the relative position of a node in respect of its similarity with the rest of them. The next step then, is to assign the proper coordinates to each subset of nodes; this step is a coordinates transformation to place the set of border nodes in the center of the drawing, confined within a circle or radius 0.5, conforming the interaction zone, and the inner nodes from each community to be placed within a ring-shaped region outside the interaction zone.

The most expensive operation in the algorithm is the MDS step. In general, MDS has a complexity of $O(2n^2)$, where n is the number of points to be placed. This algorithm divides the node set into $k+1$ subsets, making the complexity to be less than $O(2n^2)$. This effect is due to the reduction of the size of the matrices at each step. Despite this reduction of the complexity and the execution time, the algorithm is still complex and the execution time for graphs with 500000 or more nodes is considerable; this is a direction in which the approach can and should be improved, for example using GPU versions of the MDS algorithm, such as presented in [Ingram et al., 2009].

The final layout allows visualizing the nodes facilitating the interactions between communities. These interactions are represented by the connections of these nodes. However, within the interaction zone there might be a large number of nodes (with the graphs used during experimentation, 20%- 40% of the nodes are located within the interaction zone) it is necessary to filter them to identify those node playing an important

role in the interactions.

Thus, using a set of roles defined according to the connectivity pattern of each node, the function of all nodes in the graph is found. Within the interaction zone 6 roles may be identified: 3 non-hubs, of nodes with a low degree, and 3 hub roles, for nodes with a high degree. The definition of these roles hence help to identify how important is a node for a community in terms of the interaction.

In contrast with classic layout algorithms which place the nodes using different heuristics and with different purposes, the layout algorithm proposed in this chapter focuses on the interactions, allows to identify nodes with important roles and their relative location within the interaction zone.

In general, classic layout algorithms use only the affiliation variable to place the nodes. Recall this variable contains in an implicit way the structural variable of the social network, and it is used just in that implicit way. The layout algorithm presented in this chapter uses the affiliation and structural variables of a social network to make explicit their relation and to exploit the structure of each community in benefit of revealing the interactions between communities. However, more work and research needs to be done in order to interpret the knowledge extracted by the algorithm; for example, when using partitions generated using the community detection algorithm presented in Chapter 4 structural and composition variables of the graph are integrated to produce a partition, we would like to identify the impact of introducing different points of view over a partition.

Chapter 6
Conclusion and perspectives

6.1 Summary of contributions

This thesis is an initial attempt for constructing an integrated vision of the community detection algorithms to take into account both the structural and the composition variables existing within a social network. Since the increasing use of social networks and their integration to people's life: using social networking sites people can share contents, contact other people and in general interact, the amount of information associated to each actor in the networks is increasing as well.

This thesis was constructed over two main lines: first, a community detection part in which we proposed an integrated approach in order to use in an efficient way the available information of the social network. Second, we presented a layout algorithm for clustered graphs that is centered in the connections and interactions between the communities of a graph. This representation allows to focus on the nodes connecting the communities.

In the next section we detail the contributions in each line.

6.1.1 Community detection: an integrated approach

We started this thesis by recalling the variables composing a social network given by [Wasserman and Faust, 1994]; these variables are (1) a structural variable, describing the connections and ties of actors on the network; (2) a composition variable, describing each actor individually: this variable contains for example personal information, preferences, competences and in general, any information that can be used to describe a person or an organization. A third variable is used that describes the association of each actor to a group, for example to clubs, to different company branches, to universities. This variable is called . In these cases the affiliations are defined by some extrinsic designation process, however, this variable can be derived from the information given by the other

two variables.

6.1.1.1 THE NOTION OF POINT OF VIEW

The first contribution is the definition of the notion of point of view. The composition variable describes each actor individually using a set of features F^*. This set of features can be divided into different subsets which can be seen as perspectives, or *points of view* PoV_{F^*} of the composition variable. Dividing the composition variable into different points of view has two main advantages:

1. Reducing the number of features used during the analysis of the network.
2. Allowing to make an analysis per each selected point of view.

The first advantage allows selecting different features from the composition variable in such a way the points of view can group features that makes the analysis more interpretable and that reduces the number of dimensions of the features set. The second advantage allows influencing the community detection process to produce different partitions of the same graph.

6.1.1.2 INTEGRATING STRUCTURE AND COMPOSITION VARIABLES

Once we have defined the point of view PoV_{F^*}, we begin the definition of the community detection process. This process produces an affiliation variable that integrates the structure and the composition variables: this affiliation variable describes a partition of the social network with groups of well connected and similar nodes.

To generate this integrated affiliation variable we have proposed a two phases algorithm:

1. Having selected a point of view PoV_{F^*}, this point of view is used to generate an intermediate affiliate variable $\mathcal{A}_{PoV_{F^*}}$ that describes a partition of the social network according to the features of the point of view. Thus, each group contains similar nodes in terms of the chosen point of view.
2. Using the variable $\mathcal{A}_{PoV_{F^*}}$, the structural variable of the social network is changed to reflect the partition described by $\mathcal{A}_{PoV_{F^*}}$: for each edge $e \in G$, the weight of e is changed if both ends of it belong to the same group in $\mathcal{A}_{PoV_{F^*}}$. This change is made to influence the structural community detection algorithm. We use the Blondel's community detection algorithm [Blondel et al., 2008] that optimizes the modularity of the partition; since this algorithm can work with weighted graphs, we use our modified graph to privilege the creation of groups with similar nodes.

Thus, the process allows influencing the community detection process by including the information of the composition variable into the graph. The affiliation variable produced by this integrated approach produces a tradeoff between two quality measures: the modularity for measuring the partition from a structural point of view and the entropy used to measure the "disorder" of the partition induced by the composition variable.

To test our proposed community detection algorithm we used two social networks and three points of view for the first one and two points of view for the second social network.

The first social network is a network downloaded from Facebook. For this network we defined two points of view plus the null point of view; the first point of view describes the actors of the network through their academic competences, which are divided into 7 categories. The second point of view describes the actors of the network in terms of their sport preferences; this point of view contains 8 categories. We showed that in both cases the groups found by the algorithm are more homogeneous in terms of the features selected for each point of view. To contrast these results we presented the partition produced by the point of view null, which is equivalent to get a partition solely from the structure. In this case the groups are more heterogeneous, having a higher entropy than the other two.

The second social network is a representation of a co-authorship network from DBLP. Besides the point of view null, we defined another point of view representing the type of author and the preferred topic treated by that author. In this case we compared the results with the results obtained by other community detection algorithm that uses the structure and the composition variables, obtaining general better results than those reported in the work of [Zhou et al., 2009].

6.1.2 COMMUNITIES CONNECTIONS CENTERED LAYOUT ALGORITHM

The second contribution of this thesis is the development of a layout algorithm for clustered graphs that focuses on the connections between communities.

According to literature most layout algorithms for clustered graphs are focused only on showing the division of the groups, i.e., just the separation of the groups inside the partition. However, the connections between communities and the nodes participating on these connections are not explicitly taken into account.

Our proposed algorithm takes the groups of a graph partition and divide the nodes of each of these groups into two categories: (1) nodes with edges connecting different communities and (2) nodes with edges starting and finishing in the same community. Nodes in the first category are called border nodes, because they allow the community connecting to other communities. Border nodes are placed at the center of the drawing,

creating the interaction zone. Nodes in the second category are called inner nodes, because their connections remain within their own communities.

Once the nodes are assigned to one category, they are placed according to their structural similarity, defined from the common neighbors of each pair of nodes: the similarity of two nodes is proportional to the number of common neighbors they have. The 2 dimensional placement reflects the similarity of the nodes. To do this we used the Multidimensional Scaling algorithm, which maps those similarities (or dissimilarities) on a two dimensional space. This distribution of nodes helps identifying the nodes participating in the connections between communities, and the placement shows nodes with a similar neighborhood close to each other. Combining the communities configuration with the stacked bar diagrams helps analyzing the distribution of the groups derived from the composition variable within each group of the partition. The next step is the analysis of the impact of the point of view which if presented as part of the perspectives.

Additionally, using a set of roles defined for graph partitions we showed how it is easier to identify the nodes with a high participation index and those with a high number of internal connections. These roles have been deduced from the degree distribution of the partition. To test the layout algorithm we used four graphs, clustered using the Blondel's community detection algorithm [Blondel et al., 2008]. These graphs were selected because they have a community structure.

6.2 Discussion

6.2.1 Community detection algorithm

The community detection algorithm presented in this PhD thesis explores a new way to integrate two partitions of the same data, in this case nodes of a social network, to create a new partition that combines both informations. The basic idea is to find a tradeoff between those partitions, one grouping nodes according to their structural configuration and another according to their individual features.

The two-phase approach we presented here, changes the structural variable of the social network to reflect the configuration of the partition derived from the composition variable. This change biases the classic community detection algorithm used on the second phase to privilege groups with a higher weight, making more probable for two connected nodes to be in the same new group if they are in the same group in the initial partition.

This process requires the use of two clustering algorithms, one for each phase; despite this can be seen as a bottleneck, the clustering of the composition information can help analyze the network using the individual information, and so integrating two worlds.

Additionally, this process could be abstracted to define a framework to integrate the partitions generated at each step by using different clustering approaches for each phase.

6.2.2 LAYOUT ALGORITHM

The layout algorithm for clustered graphs was conceived to separate the nodes involved in connecting communities from those nodes that only have connections with other nodes within their own communities. This separation allows position each community in such a way the edges between communities can be concentrated, reducing the number of elements in the drawing plane.

Our proposed algorithm uses as a main component Multi-Dimensional Scaling, which can be expensive in terms of processing time. However, in our complexity analysis the way we divide the nodes set allows us to reduce the average complexity of the algorithm. Additionally, the design of the algorithm allows to parallelize it in two main ways: first, in a high level, by executing in parallel the layout of the inner nodes, and in a lower level, parallelizing the matrix operations used.

Another observation of this algorithm is the fact that it can be used with any graph partition. However, when applied to a social network partition that has been created using our community detection algorithm the notion of point of view is included into the graph, but the impact of that point of view has not been calculated. We would like to analyze the impact of a point of view comparing two layouts (from two partitions) of the same social network, however we think this is beyond the scope of this work and can be included into the perspectives opened by this PhD thesis.

6.3 FUTURE WORK AND PERSPECTIVES

As presented in Section 6.2, the methods and algorithms proposed in this PhD thesis open some perspectives and lead to some additional questions. This section presents these perspectives and questions, first for the community detection algorithm, then for the layout algorithm.

6.3.1 COMMUNITY DETECTION ALGORITHM

As presented in Chapter 4, the community detection algorithm looks for an equilibrium between the composition and structural variable for finding an affiliation variable that includes the other two. This could be seen as if the partition generated by the structural variable and the partition generated by the composition variable arrive to a midpoint in a partitions merging process.

Thus, knowing that the Adjusted Rand Index – ARI measures the similarity of two partitions, it could be used as a distance measure $d_{ARI}(\mathbf{C}_i, \mathbf{C}_j) = 1 - ARI(\mathbf{C}_i, \mathbf{C}_j)$. This distance is an indicator of how far are the partitions $\mathbf{C}_i$ and $\mathbf{C}_j$.

Inspired from the algorithm presented in [Cruz et al., 2011a], that reduces the entropy of the partition while increasing the modularity, and the algorithm presented in this PhD thesis, we outline an approach to find a partition that minimizes $d_{ARI}(\mathbf{C}_i, \mathbf{C}_j)$. The idea is to produce a partition $\mathbf{C}_1$ from the composition variable (as in the first clustering phase). Then, each node is assigned to one community, having one community per node. This produces a new partition $\mathbf{C}_2$ that represents the initial structural partition of the social network, and that is used to calculate the initial reference distance $d^0_{ARI}(\mathbf{C}_1, \mathbf{C}_2)$.

The last step using a Monte-Carlo approach, starts merging communities evaluating if the community merging:

1. Increases the modularity of the partition
2. Decreases the $d_{ARI}(\mathbf{C}_1, \mathbf{C}_2)$

in which case the merging can be done.

The final partition is expected to be on a midpoint between the partition generated from the composition variable and the partition generated from the structural variable.

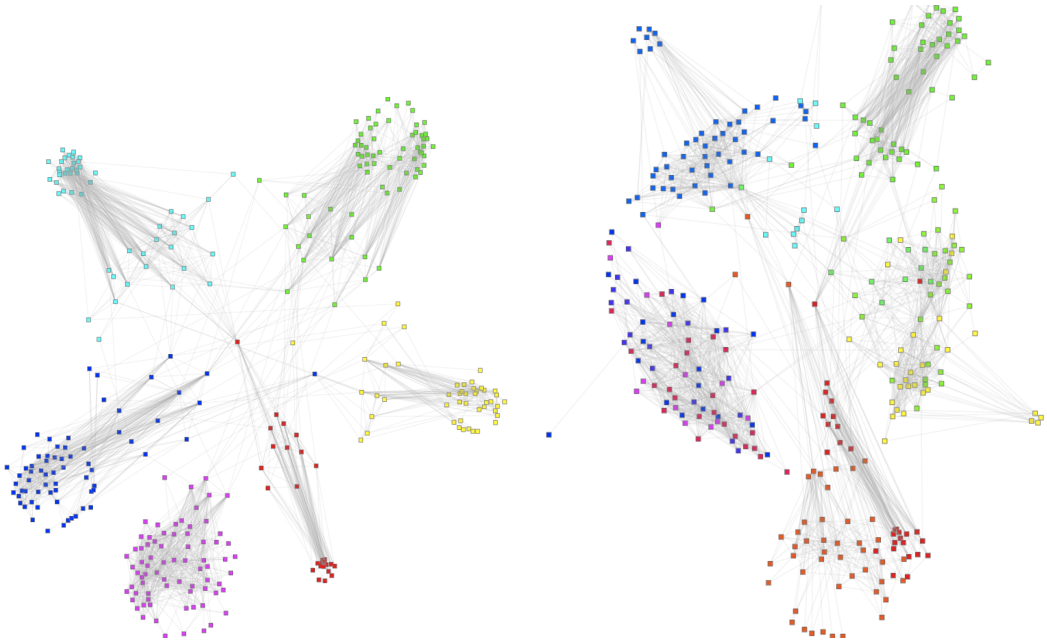
6.3.2 LAYOUT ALGORITHM

In Chapter 5 we presented a novel clustered graph layout algorithm oriented to highlight the connections between communities. This algorithm can be used with any clustered graph that has a community structure.

However, when the partition used by the algorithm includes the information of the point of view, it is difficult to measure the impact of this point of view on the layout. Figure 6.1 shows two layouts of the same social network. In Figure 6.1(a) the partition was generated using the null point of view, while Figure 6.1(b) was generated using the competences point of view (See Chapter 4.)

The impact of changing the point of view from PoV_{NULL} to PoV_{COMP} is observed on the division of some of the groups of the original partition. This division increases the number of border nodes, making them to be within the interaction zone. The problem is that the layout is completely different, making difficult to measure the impact of the new point of view.

This effect is derived from the effect of the reduction of the modularity when calculating the partition. The question is, **how to measure the impact of the point of view on the partition and how to present it on the layout?**



(a) Layout of the graph clustered using the PoV_{NULL}

(b) Layout of the graph clustered using the PoV_{COMP}

Figure 6.1: Result of the layout for the same social network clustered using different points of view

A possible answer to that question could be to create a distortion operation over the original layout to represent the community division/fusion while keeping as possible, the location of the nodes of the original partition. Exploring the impact of the point of view on the may be useful on social sciences to analyze different social networks. To measure the utility of the visual approach for social sciences researchers it is required to design a set of tests to be done by these researchers. This kind of tests will measure qualitatively the process from an user point of view.

Index

Bibliography

[Agrawal et al., 1998] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.*, 27(2):94–105.

[Aldrich and Herker, 1977] Aldrich, H. and Herker, D. (1977). Boundary spanning roles and organization structure. *The Academy of Management Review*, 2(2):pp. 217–230.

[Anderson, 2005] Anderson, J. W. (2005). *Hyperbolic Geometry*. Springer London Ltd., 2nd edition.

[Archambault et al., 2007a] Archambault, D., Munzner, T., and Auber, D. (2007a). Grouse: Feature-based, steerable graph hierarchy exploration. In *Grouse: Feature-Based, Steerable Graph Hierarchy Exploration*, pages 67–74, Sweden. Ken Museth and Torsten Möller and Anders Ynnerman.

[Archambault et al., 2007b] Archambault, D., Munzner, T., and Auber, D. (2007b). Topolayout: Multilevel graph layout by topological features. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):305–317.

[Archambault et al., 2008] Archambault, D., Munzner, T., and Auber, D. (2008). Grouseflocks: Steerable exploration of graph hierarchy space. *Visualization and Computer Graphics, IEEE Transactions on*, 14(4):900 –913.

[Auber, 2003] Auber, D. (2003). *Tulip : A huge graph visualisation framework*. P. Mutzel and M. Junger.

[Barnes, 1981] Barnes, E. R. (1981). An algorithm for partitioning the nodes of a graph. In *Proceedings of the 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, volume 20, pages 303 – 304.

[Bastian et al., 2009] Bastian, M., Heymann, S., and Jacomy, M. (2009). Gephi: An open source software for exploring and manipulating networks. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*.

[Batagelj et al., 2011] Batagelj, V., Brandenburg, F., Didimo, W., Liotta, G., Palladino, P., and Patrignani, M. (2011). Visual analysis of large graphs using (x,y)-clustering and hybrid visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 17(11):1587 –1598.

[Baur and Brandes, 2008] Baur, M. and Brandes, U. (2008). Multi-circular layout of micro/macro graphs. In *Proceedings of the 15th international conference on Graph drawing*, GD'07, pages 255–267, Berlin, Heidelberg. Springer-Verlag.

[Bellman, 1961] Bellman, R. (1961). *Adaptive control processes: a guided tour*. Rand Corporation Research studies. Princeton University Press.

[Berry et al., 2009] Berry, J. W., Hendrickson, B., LaViolette, R. A., and Phillips, C. A. (2009). arXiv:0903.1072v2.

[Bezdek, 1981] Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA.

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008 (12pp).

[Borg and Groenen, 1997] Borg, I. and Groenen, P. (1997). *Modern multidimensional scaling : theory and applications*. Springer series in statistics. Springer, New York, N.Y.

[Bourqui et al., 2007] Bourqui, R., Auber, D., and Mary, P. (2007). How to draw clustered-weighted graphs using a multilevel force-directed graph drawing algorithm. In *Information Visualization, 2007. IV '07. 11th International Conference*, pages 757 –764.

[Brandes, 2001] Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177.

[Brandes et al., 2008] Brandes, U., Gaetler, M., and Wagner, D. (2008). Engineering graph clustering: Models and experimental evaluation. *Journal of Experimental Algorithmics*, 12:1–26.

[Brockenauer and Cornelsen, 2001] Brockenauer, R. and Cornelsen, S. (2001). Drawing clusters and hierarchies. In Kaufmann, M. and Wagner, D., editors, *Drawing Graphs*, volume 2025 of *Lecture Notes in Computer Science*, pages 193–227. Springer Berlin / Heidelberg.

[Chang and Jin, 2002] Chang, J.-W. and Jin, D.-S. (2002). A new cell-based clustering method for large, high-dimensional data in data mining applications. In *Proceedings of the 2002 ACM symposium on Applied computing*, SAC '02, pages 503–507, New York, NY, USA. ACM.

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

[Chifu and Letia, 2010] Chifu, E. S. and Letia, I. A. (2010). *Self-organizing Maps in Web Mining and Semantic Web*. InTech.

[Chimiani et al., 2007] Chimiani, M., Gutwenger, C., Junger, M., Klau, G. W., Klein, K., and Mutzel, P. (2007). *Handbook of Graph Drawing and Visualization (Discrete Mathematics and Its Applications)*, chapter The Open Graph Drawing Framework (OGDF). Number 17. Chapman & Hall/CRC.

[Clauset et al., 2004] Clauset, A., Newman, M. E. J., and Moore, C. (2004). Finding community structure in very large networks. *Phys. Rev. E*, 70:066111.

[Cockburn et al., 2009] Cockburn, A., Karlson, A., and Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):2:1–2:31.

[Cross and Parker, 2004] Cross, R. and Parker, A. (2004). *The hidden power of social networks: Understanding how work really gets done in organizations*. Harvard Business School Press.

[Cruz et al., 2011a] Cruz, J. D., Bothorel, C., and Poulet, F. (2011a). Entropy based community detection in augmented social networks. In *Proceedings of the 2011 International Conference on Computational Aspects of Social Networks – CASON*, pages 163 – 168, Salamanca. LUSSI - Dépt. Logique des Usages, Sciences Sociales et de l'Information (Institut Mines-Télécom-Télécom Bretagne-UEB), IRISA - Institut de recherche en informatique et systèmes aléatoires (INRIA).

[Cruz et al., 2011b] Cruz, J. D., Bothorel, C., and Poulet, F. (2011b). Entropy based community detection in augmented social networks. In *Computational Aspects of Social Networks (CASoN), 2011 International Conference on*, pages 163 –168.

[Cruz et al., 2011c] Cruz, J. D., Bothorel, C., and Poulet, F. (2011c). Point of view based clustering of socio-semantic network. In Khenchaf, A. and Poncelet, P., editors, *EGC*, volume RNTI-E-20 of *Revue des Nouvelles Technologies de l'Information*, pages 309–310. Hermann-Éditions.

[de Leeuw, 1977] de Leeuw, J. (1977). Applications of convex analysis to multidimensional scaling. In Barra, J., Brodeau, F., Romier, G., and Cutsem, B. V., editors, *Recent Developments in Statistics*, pages 133–146. North Holland Publishing Company, Amsterdam.

[Di Battista et al., 1994] Di Battista, G., Eades, P., Tamassia, R., and Tollis, I. G. (1994). Algorithms for drawing graphs: an annotated bibliography. *Comput. Geom. Theory Appl.*, 4(5):235–282.

[di Giacomo et al., 2007] di Giacomo, E., Didimo, W., Grilli, L., and Liotta, G. (2007). Graph visualization techniques for web clustering engines. *Visualization and Computer Graphics, IEEE Transactions on*, 13(2):294 –304.

[Dice, 1945] Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

[Dongarra et al., 1990] Dongarra, J. J., Cruz, J. D., Hammerling, S., and Duff, I. S. (1990). Basic linear algebra subprograms. http://www.netlib.org/blas.

[Du et al., 2007] Du, N., Wu, B., Pei, X., Wang, B., and Xu, L. (2007). Community detection in large-scale social networks. In *WebKDD/SNA-KDD '07: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 16–25, New York, NY, USA. ACM.

[Eades and Feng, 1997] Eades, P. and Feng, Q.-W. (1997). Multilevel visualization of clustered graphs. In North, S., editor, *Graph Drawing*, volume 1190 of *Lecture Notes in Computer Science*, pages 101–112. Springer Berlin / Heidelberg.

[Eades, 1984] Eades, P. A. (1984). A heuristic for graph drawing. In *Congressus Numerantium*, volume 42, pages 149–160.

[Ellis and Dix, 2007] Ellis, G. and Dix, A. (2007). A taxonomy of clutter reduction for information visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1216–1223.

[Ellson et al., 2001] Ellson, J., Gansner, E. R., Koutsofios, E., North, S. C., and Woodhull, G. (2001). Graphviz - open source graph drawing tools. In *Graph Drawing*, pages 483–484.

[Fan et al., 2006] Fan, Y., Li, M., Zhang, P., Wu, J., and Di, Z. (2006). arXiv:physics/0609218v1.

[Flake et al., 2000] Flake, G. W., Lawrence, S., and Giles, C. L. (2000). Efficient identification of web communities. In *Proceeding of the In Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150 – 160.

[Foggia et al., 2009] Foggia, P., Percannella, G., Sansone, C., and Vento, M. (2009). Benchmarking graph-based clustering algorithms. *Image Vision Comput.*, 27(7):979–988.

[Ford and Fulkerson, 1956] Ford, L. R. and Fulkerson, D. R. (1956). Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399 – 404.

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

[Frank and Asuncion, 2010] Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

[Freeman, 1977] Freeman, L. C. (1977). A set of measures of centrality. *Sociometry*, 40(1):35 – 41.

[Fruchterman and Reingold, 1991] Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164.

[Gaetler, 2005] Gaetler, M. (2005). *Network Analysis: Methodological Foundations*, chapter Clustering, pages 178 – 215. Springer Berlin / Heidelberg.

[Gansner et al., 2005] Gansner, E. R., Koren, Y., and North, S. C. (2005). Topological fisheye views for visualizing large graphs. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):457–468.

[Garey and Johnson, 1983] Garey, M. R. and Johnson, D. S. (1983). Crossing number is np-complete. *Siam Journal On Algebraic And Discrete Methods*, 4(3):312–316.

[Girvan and Newman, 2002] Girvan, M. and Newman, E. J. (2002). Community structure in social and biological networks. *Proceeding of the National Academy of Sciences of the United States of America*, 99(12):7821 – 7826.

[Gómez et al., 2003] Gómez, J., Dasgupta, D., and Nasraoui, O. (2003). A new gravitational clustering algorithm. In *SDM*.

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

[Gordon, 1999] Gordon, A. D. (1999). *Classification*. Number 82 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, 2nd edition.

[Guimera and Amaral, 2005] Guimera, R. and Amaral, L. A. N. (2005). Cartography of complex networks: modules and universal roles. *J. Stat. Mech.-Theory Exp.*, page art. no. P02001.

[Günter and Bunke, 2003] Günter, S. and Bunke, H. (2003). Validation indices for graph clustering. *Pattern Recognition Letters*, 24(8):1107–1113.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer Berlin / Heidelberg, 5th edition.

[Heer et al., 2005] Heer, J., Card, S. K., and Landay, J. A. (2005). prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 421–430, New York, NY, USA. ACM.

[Henry et al., 2008] Henry, N., Bezerianos, A., and Fekete, J.-D. (2008). Improving the readability of clustered social networks using node duplication. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1317 –1324.

[Henry et al., 2007] Henry, N., Fekete, J.-D., and McGuffin, M. (2007). Nodetrix: a hybrid visualization of social networks. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1302 –1309.

[Herman et al., 2000] Herman, I., Marshall, M. S., and Melançon, G. (2000). Density functions for visual attributes and effective partitioning in graph visualization. In *Proceedings of the IEEE Symposium on Information Vizualization 2000*, INFOVIS '00, pages 49–, Washington, DC, USA. IEEE Computer Society.

[Hogan, 2011] Hogan, B. (2011). Namegenweb. Facebook Application.

[Holten, 2006] Holten, D. (2006). Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):741 –748.

[Huang and Eades, 1998] Huang, M. L. and Eades, P. (1998). A fully animated interactive system for clustering and navigating huge graphs. In *Proceedings of the 6th International Symposium on Graph Drawing*, GD '98, pages 374–383, London, UK. Springer-Verlag.

[Huang et al., 2005] Huang, X., Eades, P., and Lai, W. (2005). A framework of filtering, clustering and dynamic layout graphs for visualization. In *ACSC '05: Proceedings of the Twenty-eighth Australasian conference on Computer Science*, pages 87–96, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

[Hubert and Arabie, 1985] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2:193–218. 10.1007/BF01908075.

[Hubert and Schultz, 1976] Hubert, L. and Schultz, J. (1976). Quadratic assignment as a general data analysis strategy. *British Journal of Mathematical and Statistical Psychology*, 29(2):190–241.

[Hughes, 1968] Hughes, G. F. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63.

[Ingram et al., 2009] Ingram, S., Munzner, T., and Olano, M. (2009). Glimmer: Multilevel mds on the gpu. *IEEE Transactions on Visualization and Computer Graphics*, 15:249–261.

[Jain and Dubes, 1988] Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice Hall Advanced Reference Series. Prentice-Hall International.

[Kamada and Kawai, 1989] Kamada, T. and Kawai, S. (1989). An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15.

[Kantardzic, 2002] Kantardzic, M. (2002). *Data Mining: Concepts, Models, Methods, and Algorithms*. Wiley-IEEE Press, 1 edition.

[Kernighan and Lin, 1970] Kernighan, B. W. and Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(1):291 – 307.

[Kohonen, 1997] Kohonen, T. (1997). *Self-Organizing Maps*. Springer.

[Kumar and Garland, 2006] Kumar, G. and Garland, M. (2006). Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics*, 12:805–812.

[Kwak et al., 2009] Kwak, H., Choi, Y., Eom, Y.-H., Jeong, H., and Moon, S. (2009). Mining communities in networks: a solution for consistency and its evaluation. In *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 301–314, New York, NY, USA. ACM.

[Lamping et al., 1995] Lamping, J., Rao, R., and Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 401–408, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[Latapy et al., 2008] Latapy, M., Magnien, C., and Vecchio, N. D. (2008). Basic notions for the analysis of large two-mode networks. *Social Networks*, 30(1):31 – 48.

[Li et al., 2004] Li, T., Ma, S., and Ogihara, M. (2004). Entropy-based criterion in categorical clustering. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 68–, New York, NY, USA. ACM.

[Li and Takatsuka, 2004] Li, W. and Takatsuka, M. (2004). Adding filtering to geometric distortion to visualize a clustered graph on small screens. In *Proceedings of the 2004 Australasian symposium on Information Visualisation - Volume 35*, APVis '04, pages 71–79, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

[Lipczak and Milios, 2009] Lipczak, M. and Milios, E. (2009). Agglomerative genetic algorithm for clustering in social networks. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1243–1250, New York, NY, USA. ACM.

[Liu et al., 2000] Liu, B., Xia, Y., and Yu, P. S. (2000). Clustering through decision tree construction. In *Proceedings of the ninth international conference on Information and knowledge management*, CIKM '00, pages 20–29, New York, NY, USA. ACM.

[Liu, 2010] Liu, J. (2010). Fuzzy modularity and fuzzy community structure in networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, 77:547–557. 10.1140/epjb/e2010-00290-3.

[Luxburg, 2007] Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

[MacQueen, 1967] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observation. In *Proceedings of Berkely symposium on mathematical statistics and probability*, volume 1, pages 281 – 297.

[Mahajan et al., 2009] Mahajan, M., Nimbhorkar, P., and Varadarajan, K. (2009). The planar k-means problem is np-hard. In Das, S. and Uehara, R., editors, *WALCOM: Algorithms and Computation*, volume 5431 of *Lecture Notes in Computer Science*, pages 274–285. Springer Berlin / Heidelberg. 10.1007/978-3-642-00202-1_24.

[Mardia et al., 1979] Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate analysis / K.V. Mardia, J.T. Kent, J.M. Bibby*. Academic Press, London ; New York :.

[Maulik and Bandyopadhyay, 2002] Maulik, U. and Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(12):1650–1654.

[Melançon and Sallaberry, 2008] Melançon, G. and Sallaberry, A. (2008). Edge metrics for visual graph analytics: A comparative study. In *Information Visualisation, 2008. IV '08. 12th International Conference*, pages 610 –615.

[Mirkin, 2005] Mirkin, B. (2005). *Clustering for Data Mining: a Data Recovery Approach*. Number 3 in Computer Science and Data Analysis Series. Chapman & Hall/CRC.

[Mohar, 1999] Mohar, B. (1999). Drawing graphs in the hyperbolic plane. In Kratochvíyl, J., editor, *Graph Drawing*, volume 1731 of *Lecture Notes in Computer Science*, pages 127–136. Springer Berlin / Heidelberg. 10.1007/3-540-46648-7_13.

[Mun and Ha, 2005] Mun, S.-Y. and Ha, S.-W. (2005). An efficient visualization of hierarchical structures. In *Enterprise networking and Computing in Healthcare Industry, 2005. HEALTHCOM 2005. Proceedings of 7th International Workshop on*, pages 419 – 425.

[Munzner, 1998] Munzner, T. (1998). Drawing large graphs with h3viewer and site manager. In *Proceedings of the 6th International Symposium on Graph Drawing*, GD '98, pages 384–393, London, UK, UK. Springer-Verlag.

[Nagesh et al., 1999] Nagesh, H., Goil, S., and Choudhary, A. (1999). Mafia: Efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010, Center for parallel and distributed computing.

[Newman, 2001] Newman, M. E. (2001). Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality. *Physical Review. E, Statistical Nonliner and Soft Matter Physics*, 64:7.

[Newman, 2004a] Newman, M. E. J. (2004a). Detecting community structure in networks. *THE EUROPEAN PHYSICAL JOURNAL B - CONDENSED MATTER AND COMPLEX SYSTEMS*, 38(2):321 – 330.

[Newman, 2004b] Newman, M. E. J. (2004b). Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133.

[Newman and Girvan, 2004] Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review. E, Statistical Nonliner and Soft Matter Physics*, 69(2):026113.

[Noack, 2003] Noack, A. (2003). An energy model for visual graph clustering. In *Proceedings of the 11th International Symposium on Graph Drawing (GD 2003), LNCS 2912*, pages 425–436. Springer-Verlag.

[Pathak et al., 2008] Pathak, N., Delong, C., Banerjee, A., and Erickson, K. (2008). Social topic models for community extraction. In *The 2nd SNA-KDD Workshop '08 (SNA-KDD'08)*.

[Perer et al., 2011] Perer, A., Guy, I., Uziel, E., Ronen, I., and Jacovi, M. (2011). Visual social network analytics for relationship discovery in the enterprise. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 71 –79.

[Pizzuti, 2008a] Pizzuti, C. (2008a). Community detection in social networks with genetic algorithms. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 1137–1138, New York, NY, USA. ACM.

[Pizzuti, 2008b] Pizzuti, C. (2008b). Ga-net: A genetic algorithm for community detection in social networks. In Rudolph, G., Jansen, T., Lucas, S., Poloni, C., and Beume, N., editors, *Parallel Problem Solving from Nature - PPSN X*, volume 5199 of *Lecture Notes in Computer Science*, pages 1081–1090. Springer Berlin Heidelberg.

[Pizzuti, 2009] Pizzuti, C. (2009). Overlapped community detection in complex networks. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 859–866, New York, NY, USA. ACM.

[Procopiuc et al., 2002] Procopiuc, C. M., Jones, M., Agarwal, P. K., and Murali, T. M. (2002). A monte carlo algorithm for fast projective clustering. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, SIGMOD '02, pages 418–427, New York, NY, USA. ACM.

[Purchase et al., 2002] Purchase, H. C., Allder, J.-A., and Carrington, D. (2002). Graph layout aesthetics in uml diagrams: User preferences. *J. Graph Algorithms Appl*, 6:255–279.

[Purchase et al., 1997] Purchase, H. C., Cohen, R. F., and James, M. I. (1997). An experimental study of the basis for graph drawing algorithms. *J. Exp. Algorithmics*, 2.

[Rand, 1971] Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):pp. 846–850.

[Rattigan et al., 2007] Rattigan, M. J., Maier, M., and Jensen, D. (2007). Graph clustering with network structure indices. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 783–790, New York, NY, USA. ACM.

[Rijsbergen, 1979] Rijsbergen, C. J. V. (1979). *Information Retrieval*. Butterworth-Heinemann, 2nd edition.

[Roberts, 1986] Roberts, D. W. (1986). Ordination on the basis of fuzzy set theory. *PLANT ECOLOGY*, 66(3):123–131.

[Rousseeuw, 1987] Rousseeuw, P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65.

[Sachan et al., 2012] Sachan, M., Contractor, D., Faruquie, T. A., and Subramaniam, L. V. (2012). Using content and interactions for discovering communities in social networks. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 331–340, New York, NY, USA. ACM.

[Santamaría and Therón, 2008] Santamaría, R. and Therón, R. (2008). Overlapping clustered graphs: Co-authorship networks visualization. In Butz, A., Fisher, B., Krüger, A., Olivier, P., and Christie, M., editors, *Smart Graphics*, volume 5166 of *Lecture Notes in Computer Science*, pages 190–199. Springer Berlin / Heidelberg.

[Shannon et al., 2003] Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. (2003). Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504.

[Shen et al., 2006] Shen, Z., Ma, K.-L., and Eliassi-Rad, T. (2006). Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *Visualization and Computer Graphics, IEEE Transactions on*, 12(6):1427 –1439.

[Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 22(8):888–905.

[Shi et al., 2009] Shi, L., Cao, N., Liu, S., Qian, W., Tan, L., Wang, G., Sun, J., and Lin, C.-Y. (2009). Himap: Adaptive visualization of large-scale online social networks. In *Visualization Symposium, 2009. PacificVis '09. IEEE Pacific*, pages 41 –48.

[Sommer, 2010] Sommer, C. (2010). Graph datasets. Website.

[Tamassia, 1987] Tamassia, R. (1987). On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16:421–444.

[Thiel and Berthold, 2010] Thiel, K. and Berthold, M. (2010). Node similarities from spreading activation. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 1085 –1090.

[Tyler et al., 2003] Tyler, J. R., Wilkinson, D. M., and Huberman, B. A. (2003). Email as spectroscopy: automated discovery of community structure within organizations. In Huysman, M., Wenger, E., and Wulf, V., editors, *Communities and technologies*, chapter Email as spectroscopy: automated discovery of community structure within organizations, pages 81–96. Kluwer, B.V., Deventer, The Netherlands, The Netherlands.

[Wakita and Tsurumi, 2007] Wakita, K. and Tsurumi, T. (2007). Finding community structure in mega-scale social networks. *CoRR*, abs/cs/0702048.

[Wasserman and Faust, 1994] Wasserman, S. and Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Number 8 in Structural Analysis in the Social Science. Cambridge University Press.

[Wong et al., 2003] Wong, N., Carpendale, S., and Greenberg, S. (2003). Edgelens: an interactive method for managing edge congestion in graphs. In *Information Visualization, 2003. INFOVIS 2003. IEEE Symposium on*, pages 51 –58.

[Wong et al., 2006] Wong, P. C., Chin, G., Foote, H., Mackey, P., and Thomas, J. (2006). Have green - a visual analytics framework for large semantic graphs. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pages 67 –74.

[Xie and Beni, 1991] Xie, X. and Beni, G. (1991). A validity measure for fuzzy clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(8):841 –847.

[Yang et al., 2009] Yang, T., Jin, R., Chi, Y., and Zhu, S. (2009). Combining link and content for community detection: a discriminative approach. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 927–936, New York, NY, USA. ACM.

[Yen et al., 2009] Yen, L., Fouss, F., Decaestecker, C., Francq, P., and Saerens, M. (2009). Graph nodes clustering with the sigmoid commute-time kernel: A comparative study. *Data Knowl. Eng.*, 68(3):338–361.

[Zaidi et al., 2010] Zaidi, F., Archambault, D., and Melançon, G. (2010). Evaluating the quality of clustering algorithms using cluster path lengths. In Perner, P., editor, *Advances in Data Mining. Applications and Theoretical Aspects*, volume 6171 of *Lecture Notes in Computer Science*, pages 42–56. Springer Berlin / Heidelberg. 10.1007/978-3-642-14400-4_4.

[Zhao and Karypis, 2002] Zhao, Y. and Karypis, G. (2002). Comparison of agglomerative and partitional document clustering algorithms. Technical report, Department of Computer Science, University of Minnesota, Minneapolis, MN 55455.

[Zhou et al., 2009] Zhou, Y., Cheng, H., and Yu, J. X. (2009). Graph clustering based on structural/attribute similarities. *Proc. VLDB Endow.*, 2:718–729.

[Zhou et al., 2010] Zhou, Y., Cheng, H., and Yu, J. X. (2010). Clustering large attributed graphs: An efficient incremental approach. In *CDM '10 Proceedings of the 2010 IEEE International Conference on Data Mining*, pages 689–698.