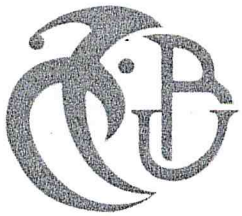
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAITRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE SAAD DAHLAB BLIDA -1-



Faculté des Sciences
Département d'Informatique

Mémoire de fin d'étude
Pour l'obtention d'un diplôme de Master en Informatique
Spécialité : Système Informatiques et Réseaux

Thème

Détection de communautés chevauchantes dans les réseaux sociaux à grande échelle

Réalisé par :

Djebli Mouàdh — Zouaoui Abdennour El Mahdi

Encadré par:

Mr. Merazka Mustapha

Organisme d'accueil : Centre de Recherche sur l'Information Scientifique et Technique

Jury :

Président :	Mr. Cherif-Zahar Sid-Ahmed.A	Maitre-assistant A
Examinateur :	Mr. Douga Yassine	Maitre-assistant B
Promoteur :	Mr. Ould-Khaoua Mohamed	Professeur à USDB

Soutenu le : 28/06/2018

Dédicaces

Je dédie ce modeste travail à :

Mes parents

Toute la famille

Et tous mes amis

Djebli Mouàdh

Dédicaces

Je dédie ce travail

À ma mère pour toute sa patience et sa persévérance durant cette période difficile.

À mon père pour son soutient moral et financier.

À mon cher frère Mounir.

À toute la famille.

À mes enseignants.

À tous mes amis

Zouaoui Abdennour El Mahdi

Remerciement

Nous remercions Dieu pour nous avoir donné santé, courage et patience afin de nous aider à réaliser ce travail.

Au terme de ce modeste travail nous tentons à remercier chaleureusement et respectivement tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste projet de fin d'étude, à savoir Notre encadreur Mr Merazka Mustapha et tout l'équipe de CERIST pour nous avoir accueillis.

Nous tentons aussi à remercier particulièrement notre promoteur Mr Ould-Khaoua Mohamed de nous avoir suivis et guidés tout au long de réalisation de notre travail.

On remercie vivement Mesdames et Messieurs les membres du jury d'avoir accepté d'évaluer notre modeste travail.

Résumé

La détection de communautés est l'un des sujets les plus populaires dans le domaine de l'analyse des réseaux sociaux, dont l'objectif est de comprendre en profondeur la structure des réseaux.

L'objectif de ce travail est de proposer une approche de détection de communautés chevauchantes qui sera stable, précise pour des réseaux sociaux à grande échelle. Ainsi la solution proposée fonctionne en deux phases. La première phase consiste à appliquer l'algorithme de Louvain pour générer une partition des communautés disjointes. Dans la deuxième phase nous proposons un ajustement pour vérifier si les nœuds situés à la frontière entre les communautés (nœuds frontaliers) peuvent être appartenir à plusieurs communautés.

L'approche développé est évaluée sur différents types de réseaux. La performance de cette approche est comparée avec d'autres algorithmes de détection de communautés chevauchantes.

Mots clés : communauté disjointe, communauté chevauchante, Louvain, nœuds frontaliers.

Abstract

Community Detection is one of the most popular topics in the field of social network analysis, whose objective is to understand in depth the structure of networks.

The purpose of this work is to propose an approach to detecting overlapping communities that will be stable, accurate for large-scale social networks. So the proposed solution works in two phases. The first phase is to apply the Louvain algorithm to generate a partition of disjoint communities. In the second phase we propose an adjustment to check if the nodes located at the border between the communities (border nodes) can be belonging to several communities.

The developed approach is evaluated on different types of networks. The performance of this approach is compared with other overlapping community detection algorithms.

Keywords : disjoint community, overlapping community, Louvain, border nodes.

ملخص

يعد اكتشاف المجتمع أحد الموضوعات الأكثر شيوعًا في مجال تحليل الشبكات الاجتماعية، هدفها هو فهم بنية الشبكات بشكل متعمق.

الهدف من هذا العمل هو اقتراح مقاربة اكتشاف المجتمعات المتداخلة بحيث ستكون هاته المقاربة مستقرة ودقيقة لشبكات التواصل الاجتماعي واسعة النطاق. لذا فإن الحل المقترح يعمل على مرحلتين. المرحلة الأولى هي تطبيق خوارزمية Louvain لتوليد تقسيم مجتمعات منفصلة. في المرحلة الثانية نقترح تعديلاً للتحقق مما إذا كانت العقد الواقعة على الحدود بين المجتمعات (نقاط الحدود) يمكن أن تنتمي إلى عدة مجتمعات.

يتم تقييم المقاربة التي قمنا بتطويرها على أنواع مختلفة من الشبكات. تتم مقارنة أداء المقاربة مع خوارزميات أخرى تكشف المجتمعات المتداخلة.

الكلمات المفتاحية: مجتمع منفصل، مجتمع متداخل، Louvain، نقاط حدود.

Table des matières

Liste des figures

Liste des tableaux

Liste des algorithmes

Table de notations

Symbole	Description
$G = (V,E)$	Graphe non orienté G formé d'un ensemble de nœuds V et d'un ensemble d'arêtes E
$G = (V,E,\omega)$	Graphe pondéré par la fonction de poids ω qui associe un nombre réel à chaque arête
n	Nombre de nœuds du graphe G
m	Nombre d'arêtes du graphe G
$d(i)$	Degré du nœud i , c'est-à-dire nombre d'arêtes incidentes à i ou somme des poids de ces arêtes pour un graphe pondéré
$P = \{C_1, \dots \dots, C_r\}$	Partition en r Communautés
C^{int}	Degré interne d'une communauté
C^{ext}	Degré externe d'une communauté
α	Degré de chevauchement
$C \cup \{i\}$	Communauté avec le nœud i à l'intérieur
$C \setminus \{i\}$	Communauté sans le nœud i à l'intérieur
U_{in}	Matrice de partition avec n ligne et k communauté
$\hat{U}$	Matrice (soft partition)
U_{out}	Matrice de partition finale
Q	Fonction de modularité
Q_{ol}	Extension de la fonction de modularité pour le cas de chevauchement
$Strength(C)$	Fonction de la force de communauté C
$Border(C)$, B_c	Liste des nœuds frontaliers de communauté C
$Normalize(\hat{U})$	Fonction de normalisation de la matrice finale $\hat{U}$

Tableau – Notations pour les graphes et les symboles.

Introduction générale

De nombreux systèmes réels sont représentés à l'aide de graphes. Des entités peuvent être modélisées par des sommets et leurs relations par des arêtes **[54]**. Il existe de nombreux domaines et champs d'application utilisant la structure de graphe (graphes sociologiques, graphes de collaborations **[86]**), réseaux biologiques (interactions protéine-protéine, réseaux de neurones, réseaux de gènes) **[87]**, réseaux sportifs, réseaux du web (graphes de pages web connectées **[22]**), réseaux de transports, etc. Ces graphes portent le nom de graphes de terrains ou de réseaux complexes.

L'étude et l'analyse de ce genre de graphes révèlent des informations sur les acteurs et expliquent la structure même du réseau. Parmi les nombreuses applications liées aux graphes, on trouve la détection de communautés qui vise à trouver au sein d'un graphe des groupes de nœuds fortement connectés entre eux et faiblement avec le reste du graphe, la détection de ces groupes (communautés) est importante car elle permet de mieux appréhender de très grands réseaux en identifiant des sous-parties dont les éléments sont semblables ou interagissent fortement entre eux. Fondamentalement, la communauté dans les réseaux sociaux peut avoir deux types, communautés disjointes et communautés chevauchantes. Dans les communautés disjointes, chaque nœud ne peut appartenir qu'à une seule communauté, mais dans le cas de communautés chevauchantes, ils peuvent appartenir à plusieurs communautés **[54]**.

La détection de communautés permet de trouver les individus d'une population les plus similaires en fonction de leurs relations. Par exemple, dans le réseau web, où les sommets sont des pages web et les arêtes des hyperliens, des groupes de pages ou de sites fortement connectés traitent souvent de thèmes apparentés et la détection de communautés permet l'amélioration des moteurs de recherche. En marketing, grâce à un graphe de co-achats où les sommets représentent les produits et les arêtes le fait que les produits ont été achetés ensemble, un algorithme de détection de communautés permettra de trouver les profils des produits les plus co-achetés pour les analyser, en vue de proposer un système de recommandation (utilisé par exemple par la société Amazon **[88]**). La détection de communautés présente trois challenges, le premier est que les tailles des communautés peuvent être différentes au sein d'un même graphe selon la nature des objets étudiés. Le second est que certains nœuds du graphe peuvent appartenir à plusieurs communautés. Par

exemple, dans un réseau de collaboration scientifique, un auteur peut publier dans différents domaines et champs scientifiques. Le troisième challenge concerne la taille des graphes. Dans un graphe à grande échelle où les communautés peuvent se chevaucher le problème devient un problème encore plus difficile, où les algorithmes de détection de communautés sont NP-hard, par exemple dans un graphe qui possède plusieurs millions de nœuds et plusieurs centaines de millions d'arêtes, appliquer un algorithme de détection de communautés sur une seule machine peut dans le meilleur des cas prendre plusieurs heures, au pire, une erreur mémoire peut survenir due à la quantité de données traitées. Dans ce cas, il est inutile d'utiliser des algorithmes exacts, qui ne peuvent être appliqués qu'à de très petits systèmes. De plus, même si un algorithme a une complexité polynomiale, il peut être encore trop lent pour s'attaquer à de grands systèmes. Dans tous ces cas, il est courant d'utiliser des Algorithmes d'approximation, i.e. des méthodes qui ne fournissent pas une solution exacte au problème à résoudre, mais seulement une solution approximative, avec l'avantage de complexité plus faible **[54]**.

Notre travail consiste à développer une heuristique pour identifier les clusters qui forment la structure de communautés chevauchantes dans des réseaux à grande échelle. Nous avons implémenté la méthode proposée, et nous avons procédé à son évaluation pour assurer une meilleure présentation du travail effectué et garantir la clarté du mémoire, outre cette introduction générale, ce manuscrit se compose de trois chapitres, une conclusion générale. Chacun met en évidence une contribution particulière du travail :

- Dans le premier chapitre de ce mémoire, nous présentons les concepts fondamentaux des graphes, la notion liés aux réseaux complexes et leurs caractéristiques, la définition de la structure de communauté dans un réseau, ses types, domaine d'application et ses intérêts.
- Dans le second chapitre, nous décrivons les principales méthodes de détection de communautés disjointes et chevauchantes.
- Dans le troisième chapitre, nous proposons une approche pour la détection de communautés chevauchantes. Nous présentons, en premier lieu, une vue globale de notre approche puis nous reprenons chaque étape de notre approche en l'expliquant en détail. Ensuite, nous présentons les résultats obtenus.

Et enfin nous présentons une synthèse sur l'approche présentée et les perspectives de nos travaux.

Chapitre I

Introduction aux réseaux sociaux complexes à grande échelle

1 Introduction

Dans ce chapitre, nous introduisons des concepts fondamentaux des graphes et présentons la notion de réseaux complexes avec les caractéristiques communes qui leur sont associées. Ensuite, nous établissons aussi les principales propriétés que nous allons utiliser ainsi de nombreuses notations qui illustrerons l'état de l'art, notre conception d'un algorithme de détection de communautés et nos expérimentations.

2 Préliminaires sur les graphes

Nous introduisons dans un premier temps les notations, hypothèses et concepts relatifs aux graphes.

2.1 Graphes

Un graphe non-orienté $G = (V, E)$ est composé d'un ensemble V de sommets (ou noeuds) et d'un ensemble **E** de paires (non ordonnées) de sommets nommées arêtes (ou liens). Nous noterons n le nombre de sommets ($n = |V|$) et m le nombre d'arêtes ($m = |E|$) **[web1]**. Les arêtes du graphe peuvent être pondérées grâce à une fonction de poids $\omega : E \rightarrow R^+$ permettant de modéliser plus finement les interactions entre sommets, nous obtenons ainsi un graphe pondéré $G = (V, E, \omega)$. Le poids d'une arête $\{i, j\}$ entre deux sommets i et j sera noté ω_{ij}. Par convention, un poids nul est attribué dans le cas où l'arête n'existe pas ($\omega_{ij} = 0 \; si \; \{i, j\} \notin E$). Dans le cas d'un graphe non pondéré les poids des arêtes de **E** sont fixés à 1, ainsi dans ce cas particulier $\forall_{i,j} \in V, \omega_{ij} \in \{0, 1\}$ **[web2]**.

2.2 Degré

Le degré d(v) d'un sommet $v \in V$ est le nombre d'arêtes incidentes au sommet v ; il s'agit du nombre de sommets voisins de v. Nous définissons aussi le poids $\omega(i)$ d'un sommet i comme la somme des poids de ses arêtes incidentes :

$$\omega(i) = \sum_{j \in V} \omega_{ij}$$

Notons que le poids d'un sommet coïncide avec la définition du degré d'un sommet dans le cas des graphes non-pondérés **[web3]**.

2.3 Voisinage

On dit que deux sommets d'un graphe non-orienté sont voisins ou adjacent s'ils sont reliés par une arête. Dans un graphe G non orienté le voisinage d'un sommet v ∈ V, souvent noté $N_G(v)$ peut désigner l'ensemble de ses sommets voisins ou bien un sous-graphe associé. Dans un graphe orienté, on emploie généralement le terme de prédécesseur ou de successeur **[web4]**.

2.4 Distance

La distance entre deux nœuds d'un graphe est la longueur d'un plus court chemin entre ces deux nœuds1. La longueur d'un chemin est sa longueur en nombre d'arêtes. Pour un graphe pondéré c'est la somme des poids des arêtes empruntées **[web5]**.

2.5 Diamètre

Le diamètre d'un graphe est la plus grande distance possible qui puisse exister entre deux de ses sommets **[web6]**.

2.6 Densité d'un graphe

La densité d'un graphe est définie comme $\frac{2m}{n(n-1)}$ soit le rapport entre le nombre d'arêtes et le nombre maximum d'arêtes possibles compte tenu du nombre de nœuds du graphe **[web7]**.

2.7 Matrice d'adjacence

La matrice d'adjacence A représentant les arêtes d'un graphe pondéré non-orienté est définie par :

$$A_{ij} = A_{ji} = \begin{cases} 0 & si\ \{i,j\} \notin E \\ \omega_{ij} & si\ \{i,j\} \in E \end{cases}$$

Dans le cas d'un graphe non-orienté, la matrice d'adjacence A est symétrique ($A^T = A$). Par ailleurs, dans le cas d'un graphe non pondéré $A_{ij} \in \{0,1\}$ (car nous fixons un poids ω_{ij} égal à 1 pour toutes les arêtes de E **[web8]**.

2.8 Graphe complet

Un graphe orienté élémentaire est dit complet s'il comporte un arc (v_i, v_j) et un arc (v_j, v_i) pour tout couple de sommets différents v_i , v_j ∈ V. De même, un graphe non-orienté simple est dit complet s'il comporte une arête $\{v_i, v_j\}$ pour toute paire de sommets différents v_i , v_j ∈ V. On note K_n un graphe complet d'ordre n **[web9]**.

2.9 Graphe connexe

Soit G un graphe. Le graphe G est dit connexe lorsqu'il existe une chaîne entre deux sommets quelconques de G **[web10]**.

2.10 Sous graphe

Soit G1 et G2 deux graphes. G2 est un sous-graphe de G1 si et seulement si :

- Les sommets de G2 sont des sommets de G1
- Deux sommets de G2 sont adjacents si et seulement si ces deux sommets sont adjacents pour le graphe G1.

Autrement dit :

G2 est un sous-graphe de G1 si et seulement si G2 est composé de certains sommets de G1 et de toutes les boucles et arêtes qui les relient dans G1 **[web11]**.

2.11 Clique

Une clique d'un graphe non orienté est, en théorie des graphes, un sous-ensemble des sommets de ce graphe dont le sous-graphe induit est complet, c'est-à-dire que deux sommets quelconques de la clique sont toujours adjacents **[web12]**.

2.12 Composante connexe

Dans un graphe non orienté, une composante connexe est un sous-graphe induit maximal connexe, c'est-à-dire un ensemble de points qui sont reliés deux à deux par un chemin. On peut ainsi regrouper les sommets d'un graphe selon leur appartenance à la même composante connexe. Dans le cas des graphes orientés on parle de composante fortement connexe pour un ensemble de sommets reliés les uns aux autres par des chemins du graphe **[web13]**.

2.13 Partions des sommets d'un graphe

Une partition d'un graphe est une partition de ses nœuds, ou plus rarement de ses arêtes. Si le nombre de groupe dans la partition est fixé à un entier k, on parle de k-partition. Pour $k=2$, on parle parfois de bissection.

Le partitionnement est le fait de calculer une partition. Le plus souvent le partitionnement de graphe consiste à créer une subdivision de l'ensemble des sommets de S en k sous-ensembles de tailles réduites de façon à minimiser un ou plusieurs critères **[web14]**.

3 Modélisation par les graphes

Il existe de nombreux domaines et champs d'application utilisant la structure de graphe (graphes sociologiques, graphes de collaborations **[86]**, réseaux biologiques (interactions protéine-protéine, réseaux de neurones, réseaux de gènes) **[87]**, réseaux sportifs, réseaux du web (graphes de pages web connectées) **[22]**, réseaux de transports, etc). Ces graphes portent le nom de graphes de terrains ou de réseaux complexes.

3.1 Les réseaux sociaux

L'un des premiers ayant travaillé sur les relations humaines à travers les graphes fut Moreno (1934) **[1]** qui publia "Who shall survive ?". Il étudie les affinités entre élèves de classes de divers degrés, représentés sous forme de sociogramme (un diagramme des liens sociaux qu'une personne possède) comme illustré dans la figure suivante, posant ainsi les bases de la sociométrie. L'auteur explique par la suite l'importance de l'utilisation des graphes comme outils d'analyse sociologique, Moreno (1951) **[2]**.

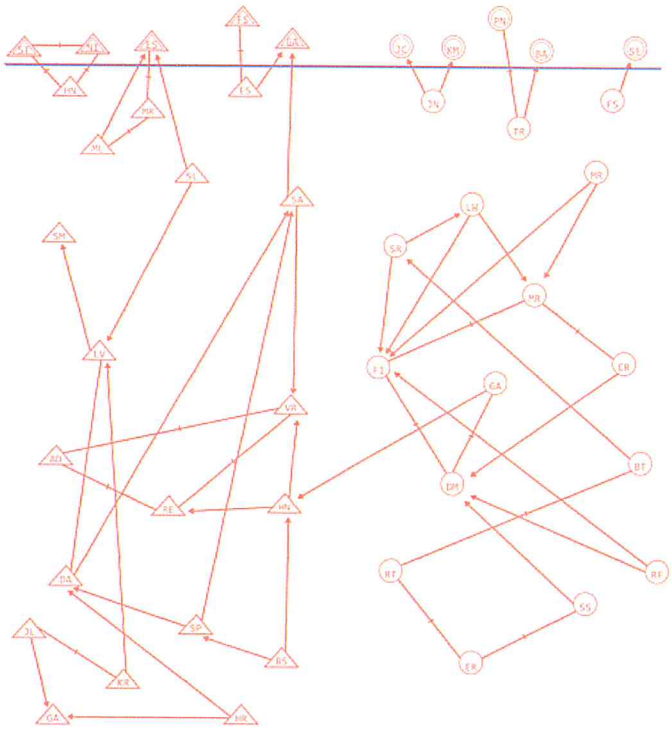


Figure 1.1 - Sociogramme de Moreno ,1934 « Who shall survive ? ».
Les filles (cercles) et les garçons (triangles). Les liens montrent deux meilleurs amis. Ligne bleue définit la frontière du groupe.

C'est à Barnes (1954) **[3]** que l'on doit le terme analyse des réseaux sociaux. Il porta son analyse sur un cas particulier de réseaux de terrains, les réseaux sociologiques.

L'analyse des réseaux sociaux est une approche sociologique c'est-à-dire l'étude des êtres humains dans leur milieu social, elle est fondée sur l'étude de la théorie des réseaux appliquée aux réseaux sociaux qui a pour objectif de rechercher des explications et des compréhensions typiquement sociales.

Les réseaux sociaux représentent des relations impliquant des entités sociales telles que les amitiés entre individus, la communication dans un groupe ou les transactions entre sociétés. Trouver des acteurs importants, découvrir des groupes ou des communautés cohésives, ou identifier des acteurs qui sont similaires d'une manière ou d'une autre, sont autant d'exemples d'analyses qui peuvent être faites pour les réseaux sociaux.

La théorie des réseaux sociaux conçoit les relations sociales en termes de nœuds et arêtes. Les

nœuds sont habituellement les acteurs sociaux dans le réseau, et les arêtes sont les interactions ou des relations entre ces nœuds.

Les réseaux sociaux sont un type de réseaux complexe, ils ont permis de valider à plus grande échelle des théories émises en sociologie comme par exemple la théorie des six degrés de séparation de Milgram ou la notion de réseau "petit monde" (Small world) **[18]**.

3.2 Information et technologie

Nous pouvons ici distinguer les réseaux d'informations, comme les réseaux de collaboration entre scientifiques **[20]**, les réseaux sémantiques **[21]** ou le World Wide Web **[22]**, et les réseaux technologiques, de transports **[23]** ou de routeurs **[24]** par exemple. Ils présentent généralement les caractéristiques des réseaux complexes dont celle de réseau sans échelles, vérifiée si la distribution des degrés de nœuds suit une loi particulière appelée loi de puissance **[12]**. Pour être précis, ce type de réseau est dénommé sans échelles. Le degré d'un élément du réseau, c'est-à-dire son nombre de liens avec les autres éléments, peut varier considérablement par rapport à la moyenne, de sorte qu'il y a de très nombreux éléments avec un degré faible et que le nombre d'éléments du réseau diminue fortement d'autant plus que le degré augmente.

3.3 Les réseaux biologiques

Les biologistes rencontrent des réseaux métaboliques, modélisant les processus de génération et de dégradation des matériaux et de l'énergie au sein d'organismes vivants **[25]**, des réseaux d'interaction entre protéines **[26]** ou encore des réseaux de régulation génétique **[27]**. Nous pouvons citer également les réseaux de neurones et les réseaux alimentaires. Ces réseaux exhibent des propriétés de réseaux complexes, en particulier la distribution sans échelles, d'après les études topologiques dont ils font l'objet **[28, 29]**.

4 Les réseaux complexes

Un graphe de terrain (ou réseau complexe) est un réseau constitué de données collectées correspondant à une vérité. Un graphe de terrain a des caractéristiques topologiques non triviales. Ces caractéristiques, qui n'apparaissent pas dans les graphes simples ou graphes aléatoires, mais émergent lors de la modélisation de systèmes réels. Ces réseaux mettent en évidence un ensemble d'individus, d'organisations ou d'objets reliés par des interactions sociales.

L'émergence du concept général de réseaux complexes s'appuie historiquement sur deux modèles de réseaux clairement identifiés comme ayant des propriétés remarquables par rapport aux graphes aléatoires. Des études menées notamment par Barabási et Bonabeau (2003) **[12]**, Newman (2003) **[13]** et Clauset et al (2009) **[14]** ont essayé de recenser les caractéristiques communes des réseaux complexes. Les caractéristiques portent sur l'effet « petit monde », un fort nombre de triangles entre groupes de nœuds fortement connectés et une distribution des degrés suivant une loi de puissance.

Donc pour caractériser les réseaux complexes, nous sommes intéressés par ces trois propriétés essentielles :

- Tout d'abord les réseaux petit-mondes (small world nerworks), le fait que chaque individu puisse être relié à n'importe quel autre par une courte chaîne de relations sociales. Ce concept reprend, après l'expérience du petit monde, conduite en 1967 par le psychosociologue Stanley Milgram, le concept de « six degrés de séparation » **[18]**. Milgram a fait une série d'expériences dans les réseaux petit-mondes pour estimer la distance moyenne entre les individus au sein d'un réseau social, le nombre moyen d'étapes était de 5,5 à 6 qui fournissent des preuves de six degrés de séparation donc celui-ci suggère que deux personnes, choisies au hasard parmi les citoyens américains, sont reliées en moyenne par une chaîne de six relations. Un réseau « petit monde », ou simplement un petit monde, est un modèle mathématique utilisé pour modéliser des réseaux réels, notamment les réseaux sociaux. Dans la majorité des cas, deux nœuds, i.e. deux personnes, peuvent être reliés par un très petit nombre d'amis intermédiaires comme illustre dans la figure suivante.

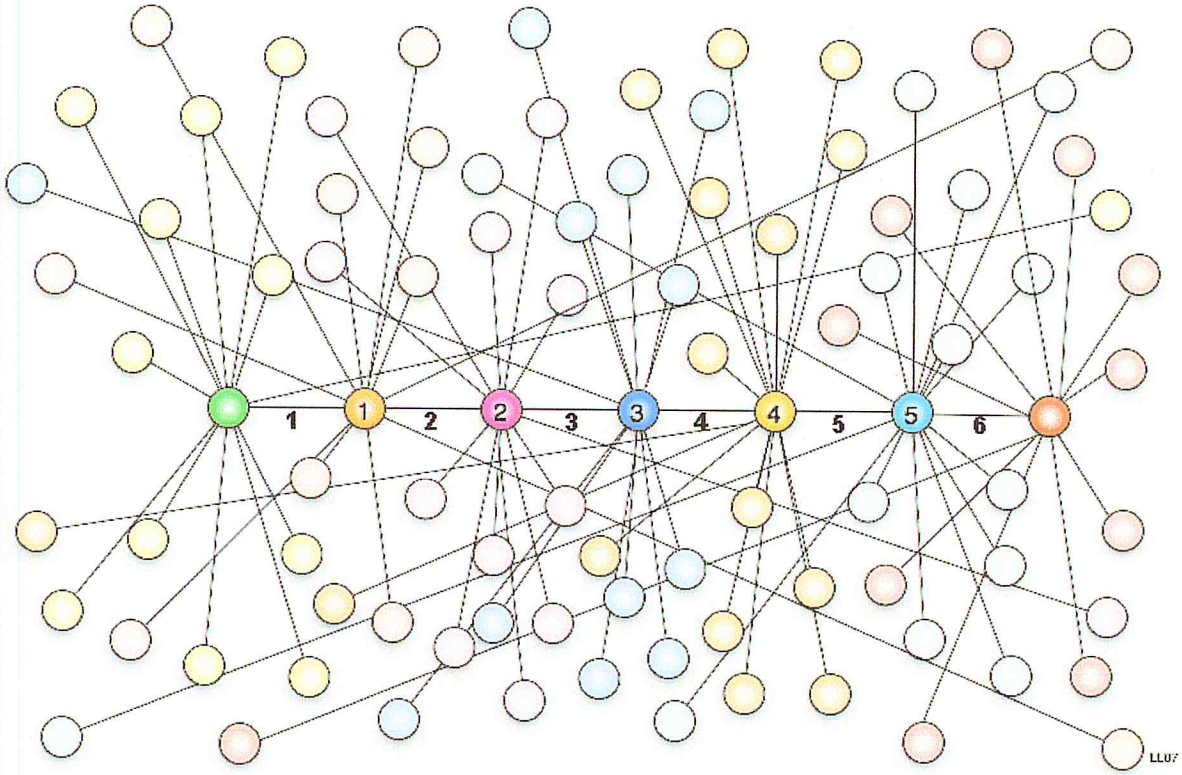


Figure 1.2 - Illustration de la propriété de petit monde.

- Ensuite, le nombre de triangles, cela traduit le fait que si les nœuds u et u' sont reliés, de même que u' et u'', alors il est très probable que u et u'' le soient aussi. Cette propriété est mesurée par le coefficient de clustering global défini par le rapport du nombre de triplets fermés (formant un triangle) et du nombre total de triplets **[15]**.

- Enfin, les réseaux sans échelles (scale-free networks) **[12]**, est un réseau dont les degrés suivent une loi de puissance (power law degree distribution). Plus explicitement, dans un tel réseau, la proportion de nœuds de degré k est proportionnelle à $k^{-\gamma}$ pour k grand, où γ est un paramètre (situé entre 2 et 3 pour la plupart des applications), Dans cette distribution, il y a un grand nombre de nœuds de degré faible et très peu de nœuds ayant un degré élevé. Beaucoup de réseaux, comme le réseau du web, les réseaux sociaux et les réseaux

biologiques semblent se comporter comme des réseaux sans échelles, la figure suivante montre la différence entre un réseau aléatoire et un réseau sans-échelle.

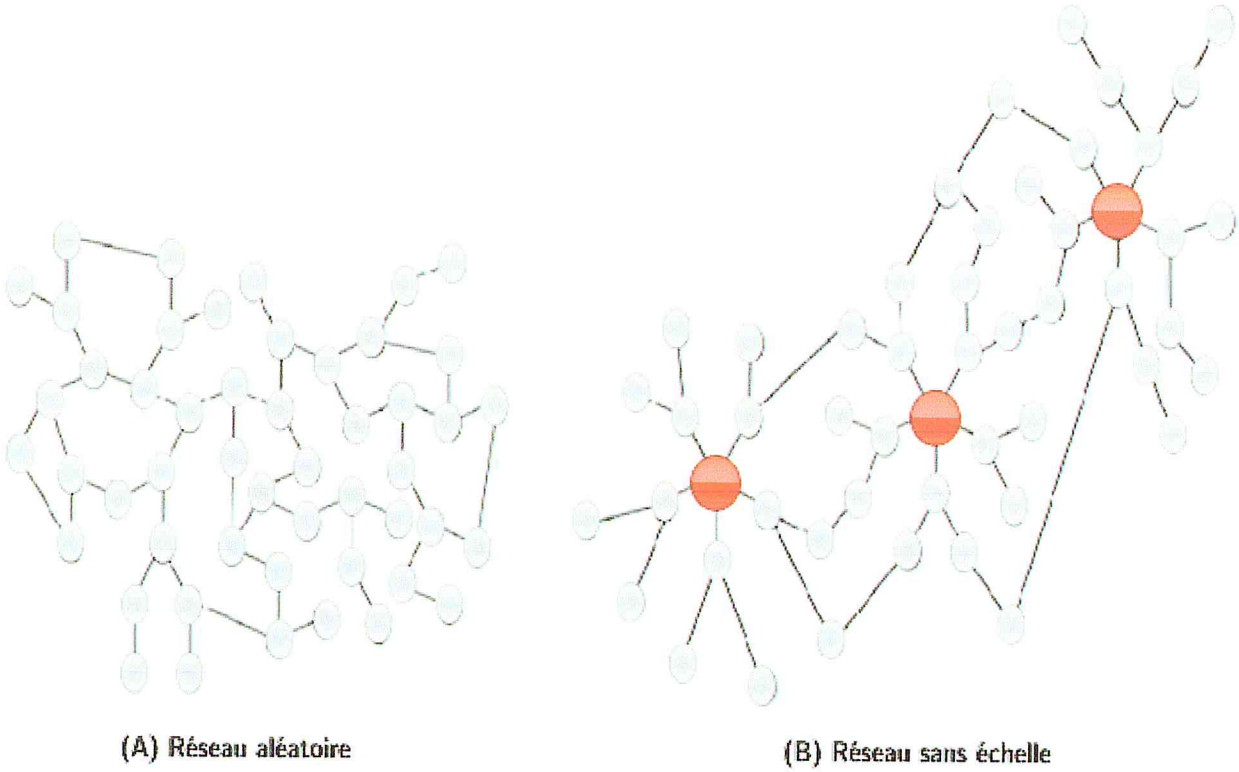


Figure 1.3 - Le réseau aléatoire et le réseau sans échelle.

(A) Le réseau aléatoire suit la distribution de Poisson **[16]**. La plupart des nœuds ont approximativement le même nombre de liens. (B) Réseau sans échelle suit la loi de puissance. La plupart des nœuds ont un ou deux liens, mais quelques nœuds hautement connectés, appelés nœuds centralisés (hubs), ils ont un grand nombre de liens. Les cercles gris indiquent les nœuds et les rouges les nœuds centralisés qui sont des nœuds hautement connectés.

Dans la figure suivante, nous verrons un exemple sur la distribution des degrés dans un réseau sans échelle selon (power law degree distribution) $k^{-\gamma}$, avec $\gamma = 2$ dans cet exemple.

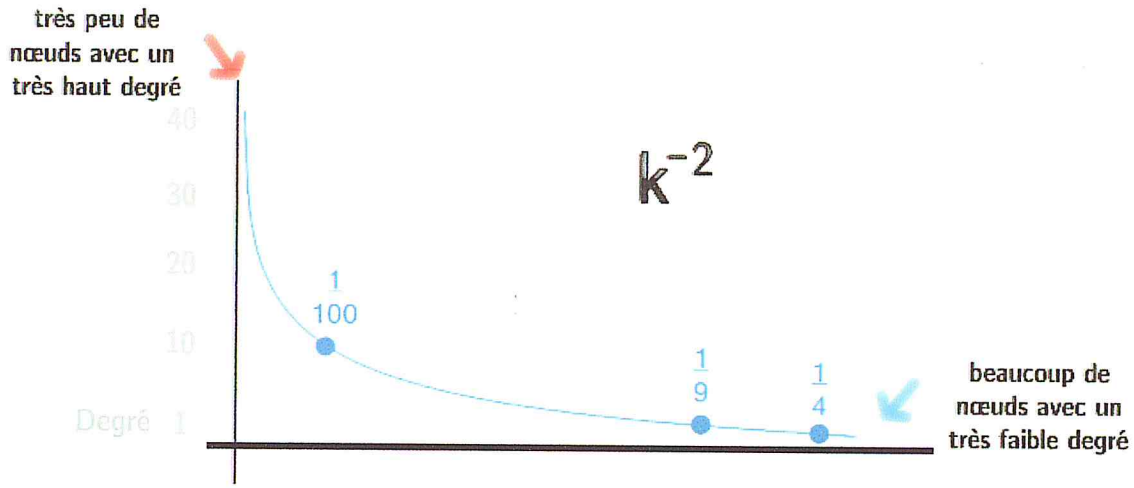


Figure 1.4 – La distribution des degrés dans un réseau sans échelle.

Dans ce réseau, le nombre de nœuds de degré k est proportionnel à k^{-2}, de sorte que le nombre de nœuds de degré 2 est le quart de tous les nœuds. Le nombre de nœuds de degré 3 est 1/9 de tous les nœuds et le nombre de nœuds de degré 10 est proportionnel à 1/100.

Le point à retenir est que cette longue queue (flèche rouge) signifie qu'il peut y avoir très peu de nœuds avec un très haut degré, mais il y aura aussi beaucoup de nœuds avec un très faible degré de connectivité (flèche bleu), ce qui nous donne un réseau hautement centralisé.

Ces propriétés, réunies dans un même graphe, engendrent d'autres propriétés dont une nous intéresse particulièrement, la plus importantes est l'existence de zones de densité très forte, à différentes échelles. C'est précisément la propriété de structure communautaire, qui fait l'objet de notre étude.

La théorie des graphes fournit un support de modélisation des réseaux complexes en généralisant leur structure quel que soit leur origine :

- Un élément constitutif du réseau (individu, ordinateur, protéines…) est représenté par un sommet ou nœud de graph.
- Une relation ou un lien entre deux éléments est représenté par une arête ou un arc du graphe.

Cette modélisation permet d'exprimer les propriétés distinctives des réseaux complexes, et d'y appliquer des algorithmes pour résoudre les problèmes que ceux-ci soulèvent.

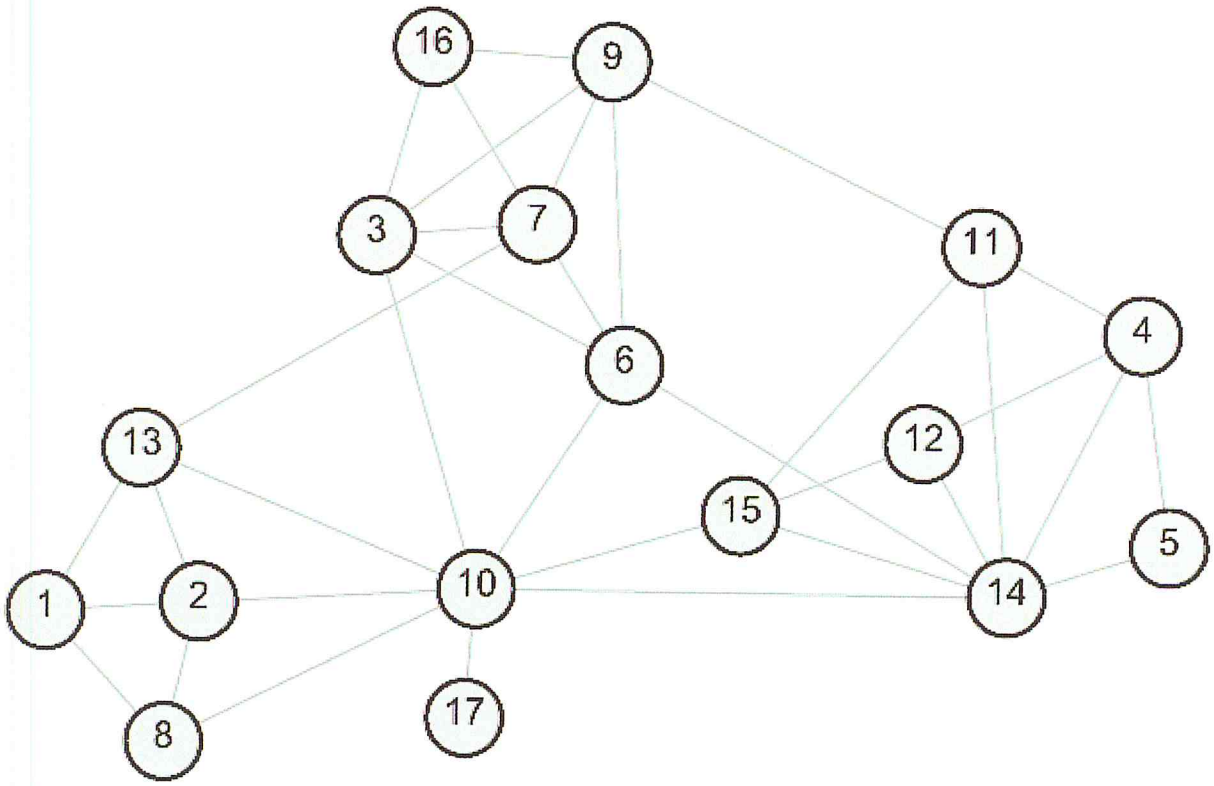


Figure 1.5 – Exemple d'un graphe.

Le graphe suivant contient 17 nœuds et 35 arêtes. Il est impossible d'estimer la distribution des degrés de nœuds avec un graphe si petit, mais la présence de nœuds centraux de fort degré est manifeste (n° 10 et 14).

5 Graphes aléatoires

Un graphe aléatoire est un graphe qui est généré par un processus aléatoire. Ils ont les propriétés précédentes : l'effet petit monde, distribution des degrés suivant une loi puissance, fort nombre de triangles et aussi la densité de ces types des graphes et petite c'est-à-dire que les degrés des sommets sont petits comparé à la taille du graphe. Un graphe aléatoire peut posséder ou pas les propriétés énoncées précédemment, ils sont des modèles pour étudier les grands graphes comme les graphes des réseaux sociaux, biologies, information et technologie etc... **[web15]**.

Les modèles de graphes utiles, par comparaison, à l'étude des réseaux complexes sont :

- Le modèle de graphe aléatoire de Erdös-Rényi (ER) **[17]** dont la distribution de degré suit une loi de Poisson **[16]**.
- Le modèle petit-monde présenté par Watts et Strogatz **[15]** qui, en jouant sur un paramètre de recâblage de nœuds, la distribution de degré suit une loi binomiale **[19]**.
  Le modèle de réseau sans échelles de Barabasi-Albert avec une distribution de degré de nœuds en loi de puissance, vérifie la propriété de longueur moyenne de chemin faible.

6 Les communautés et leurs propriétés

Les études portaient sur le fait de savoir si des personnes de même catégorie sociale allaient entrer en relation ou non. C'est en 1950 que les premières études portant sur les questions de groupes d'individus furent lancées par Homans (1950) **[7]**. Il exposa des exemples de groupes au sein de graphes sociaux. Ses travaux furent continués par Nadel (1957) **[8]** qui montra l'existence de structures sociales intrinsèques aux réseaux réels. Les premières études des structures de groupes au sein du graphe furent proposées par Glanzer et Glaser (1961) **[9]**. Les méthodes étaient basées sur des graphes de communication.

Il existe dans les réseaux sociaux à grande échelle des zones qu'ils sont plus densément connectées que d'autre, ces zones sont appelées « communautés » par analogie avec les réseaux sociaux, et correspondent à des groupes de sommets plus fortement connectés entre eux qu'avec les autres sommets. L'existence de structures de nœuds densément connectés entre eux et faiblement avec le reste du graphe fut exposée par Newman, qui employa le terme de communautés pour définir ces groupes de nœuds. Il montra que la présence de structures communautaires était une caractéristique des réseaux complexes **[54]**.

7 Structure de communautés

Dans l'étude des réseaux complexes, on dit qu'un réseau a une structure communautaire si les nœuds du réseau peuvent être facilement regroupés en ensembles de nœuds (se chevauchant potentiellement) de sorte que chaque ensemble de nœuds est densément connecté en interne. Dans le cas particulier de la recherche de communautés sans chevauchement, cela implique que le réseau se divise naturellement en groupes de nœuds avec des connexions denses en interne et des connexions plus éparses entre les groupes. Mais les communautés qui se chevauchent sont également autorisées. La définition la plus générale repose sur le principe que les paires de nœuds

sont plus susceptibles d'être connectées si elles sont toutes deux membres de la même communauté et moins susceptibles d'être connectées si elles ne partagent pas les communautés. La figure suivante montre la structure de communautés dans un réseau où les nœuds de ce réseau sont divisés en trois groupes, la plupart des connexions se situant dans des groupes et seulement quelques-uns entre groupes **[54]**.

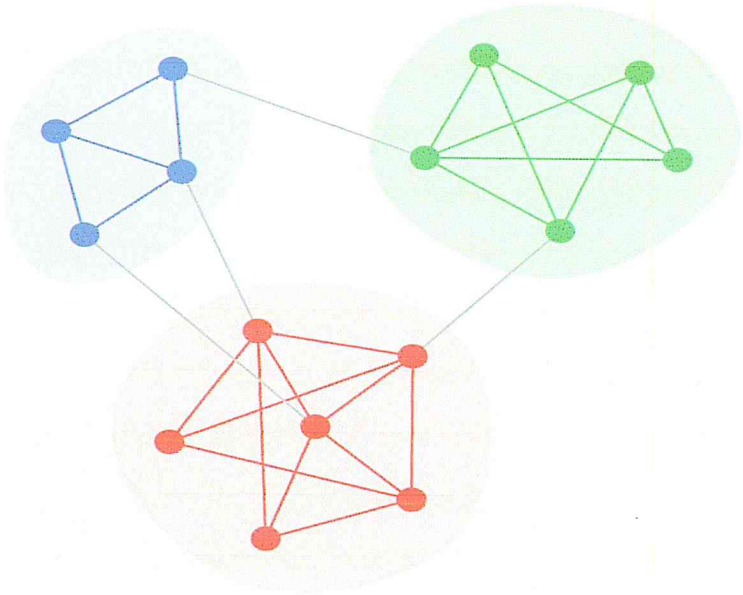


Figure 1.6 - Exemple de réseau montrant la structure de la communauté.

8 Nœud chevauchant ou non chevauchant

Les premiers travaux concernant la possibilité qu'un nœud d'un graphe social puisse appartenir à plusieurs groupes de nœuds furent introduits par Bonacich (1972) **[10]** qui utilisa la topologique des graphes pour en exposer l'existence. Breiger (1974) **[11]** proposa de dissocier les arêtes dans un groupe de nœuds très connectés de celles sortant de ces structures. L'idée fut d'expliquer les groupes de nœuds à travers les arêtes et de mettre une voix à la détection de telles structures.

Dans de nombreux modèles, Il est possible que certains nœuds appartiennent à plusieurs communautés. Par exemple, certains nœuds ont un nombre de liens identiques à plusieurs communautés. En considérant l'exemple ci-dessous qui montre deux communautés qui se chevauchent (indiqué par une ligne pointillée), on peut voir que le nœud en rouge est chevauché, il est dans les deux communauté (bleue et verte).

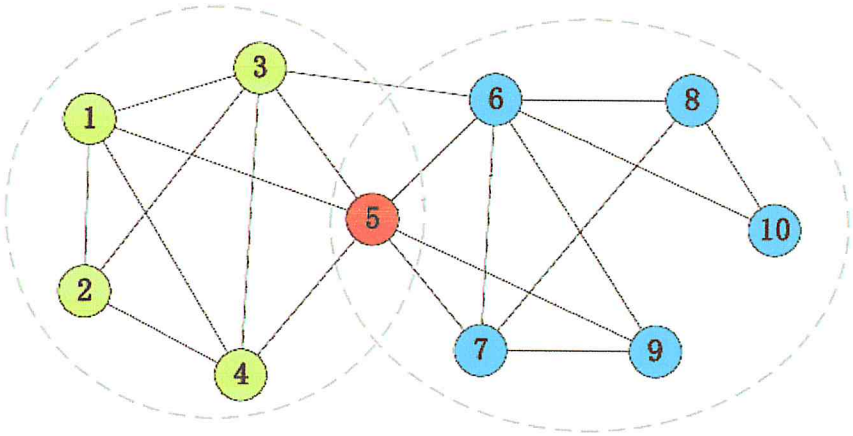


Figure 1.7 - Exemple de réseau montrant deux communautés qui se chevauchent.

On peut voir que le nœud en rouge est chevauché, c'est-à-dire qu'il est dans les deux communautés.

9 Intérêt de la détection de communautés chevauchantes

De nombreux réseaux du monde réel contiennent des communautés qui se chevauchent comme les réseaux protéine-protéine et les réseaux sociaux. La détection de communautés chevauchantes joue un rôle important dans l'étude de la structure cachée de ces réseaux et permet d'enrichir l'analyse, car elle est plus réaliste. Par exemple dans les réseaux sociaux, où un individu appartient habituellement à différents cercles à la fois, de celui des collègues de travail à la famille, aux associations sportives, etc… **[54].**

10 Détection de communautés

La détection de communautés est un domaine de recherche actif depuis ces vingt dernières années. De très nombreuses approches ont été mises en œuvre pour la détection de structures communautaires.

10.1 Définition formelle du problème

Dans l'hypothèse où un nœud appartient au plus à une communauté, le problème consiste à définir une partition de l'ensemble des nœuds, au sens mathématique, soit qui satisfait un critère de validation des communautés, soit qui maximise une fonction d'évaluation du partitionnement.

Considérons un réseau social représenté par un graphe $G\,(V,E)$. Le problème de détection de communautés dans sa forme générale consiste à trouver une partition $P = \{C_1, \dots \dots, C_r\}$ de l'ensemble des sommets V en r communautés avec $\bigcup_{k \in \{1 \dots r\}} = V$, $C_k \bigcap C_l = \emptyset$ $r \geq k \geq l$ et $C_r \neq \emptyset, \forall\, k \in \{1 \dots r\}$ de telle sorte que les sommets dans une communauté soient fortement connectés entre eux et faiblement avec le reste de graphe.

La majorité des algorithmes de détection de communauté trouvent des communautés disjointes, dans ce cas chaque nœud n'appartient qu'à une seule communauté.

La détection de communautés chevauchantes consiste à trouver des groupes de nœuds fortement connectés entre eux et faiblement avec le reste du graphe, avec des nœuds pouvant appartenir à plusieurs communautés. Plus formellement, il s'agit de trouver une couverture $C = \{C_1, \dots, C_k\}$, avec $C_k = \emptyset$, k n'est pas connue au préalable, avec $C_i \cap C_j = \emptyset$ ou $C_i \cap C_j \neq \emptyset$, et $\bigcup_{i=1}^{k} C_i = V$.

10.2 Intérêt de la détection de communautés et ses applications

La détection de communautés dans un réseau complexe recouvre deux réalités selon qu'un nœud du réseau peut ou non appartenir à plusieurs communautés. Ce problème se pose de différentes manières selon les disciplines. Les sociologues voient la communauté dans son acception générale de groupe d'individus partageant des valeurs, une culture, des comportements communs ou tout simplement des affinités. Dans ce contexte, il est fréquent que les communautés se chevauchent, c'est-à-dire qu'un individu se revendique de plusieurs groupes. Dans une autre application sociale, un hôpital par exemple, les communautés sont de différentes natures et permettent d'étudier la propagation des bactéries au sein de l'hôpital. Dans les réseaux d'interaction entre protéines, celles-ci interagissant fortement entre elles au sein d'un module (un groupe de protéines) ont des fonctions similaires voir identiques dans les cellules de l'organisme étudié et participent ensemble à un même processus biologique.

Les communautés permettent ainsi aux chercheurs de schématiser les réseaux qu'ils étudient en identifiant des sous-parties similaires. De la même manière, le découpage en communautés favorise la visualisation d'un graphe puisqu'une communauté peut être perçue comme un constituant du réseau, avec la possibilité de plonger à l'intérieur pour visualiser et étudier ses composants, des nœuds simples ou des sous-communautés si le réseau est hiérarchique. Ce découpage permet

également, comme pour le problème du partitionnement de graphe, d'obtenir un graphe grossier, une communauté étant vue comme un nœud, qui peut être traité plus facilement.

11 Conclusion

Au cours de ce premier chapitre, on a abordé l'ensemble des notions nécessaires relatives aux graphes, on a introduit également les concepts de la théorie des graphes et on a également présenté celui-ci en précisant notre problématique, ce qui nous permettra d'entamer sans problèmes le prochain chapitre état de l'art et ainsi s'approcher un peu plus du vif du sujet.

Chapitre II

État de l'art

1 Introduction

La communauté peut avoir deux types : communautés disjointes et communautés chevauchantes, donc la détection de communautés dans un réseau complexe recouvre deux réalités selon qu'un nœud du réseau peut ou non appartenir à plusieurs communautés. La détection de communautés est un domaine de recherche actif depuis ces vingt dernières années. De très nombreuses approches ont été mises en œuvre pour la détection de structures communautaires. Certaines méthodes considèrent le graphe dans son ensemble et effectuent une coupe pour trouver des communautés alors que d'autres privilégieront une approche nodale (c'est-à-dire, un partitionnement fondé sur les propriétés de nœuds voisins).

Ce chapitre vise à présenter les principales méthodes de détection de communautés disjointes et chevauchantes. Nous donnerons une formulation à la détection de communautés disjointes et chevauchantes, ainsi que les méthodes respectives pour résoudre ces problèmes. A la fin de chaque section un tableau récapitulatif sera présenté pour montrer la complexité de chaque méthode. A partir de l'état de l'art que nous aurons présenté, nous tirerons les conclusions nécessaires pour le développement de notre solution de détection de communautés chevauchantes.

2 Détection de communautés disjointes

Nous allons lister ici les principales approches qui ont été proposées à ce jour. Bien que la liste soit importante, elle est non exhaustive afin d'en limiter la longueur, nous n'avons retenu que les approches qui ont reçu le plus d'attention de la part de la communauté scientifique. Notre but est de donner une vue d'ensemble des méthodes proposées, et d'en illustrer la diversité.

2.1 Les approches classiques

La détection de communautés s'approche des thématiques classiques en informatique que sont : le partitionnement de graphe, le clustering hiérarchique et le clustering de partition.

2.1.1 Partitionnement de graphe:

Le but de partitionnement de graphe et de grouper les sommets en des clusters de taille prédéfinie (et le nombre de parties aussi prédéterminées) tel que le nombre d'arête entre les clusters est minimal. Nombreux algorithmes effectuent une bissection du graphe et le partitionnement en plus

de deux clusters est généralement obtenue par bissection itérative, on impose la contrainte que les clusters ont une taille égale. Donc la spécification de nombre de clusters et la taille est nécessaire. Cette approche ne convient pas totalement à la détection de communautés car elle a l'inconvénient de requérir une connaissance préalable du nombre de communautés recherchées ainsi que de leurs tailles.

Ce problème apparaît par exemple dans la conception de circuits imprimés et l'ordonnancement de tâches exécutées sur plusieurs processeurs. La plupart des variantes du problème de partitionnement de graphe sont NP_Difficile **[54]**, Il existe cependant plusieurs algorithmes qui peuvent faire du bon travail, même si leurs solutions ne sont pas forcément optimales.

Une des méthodes locales parmi les plus connues pour le résoudre est l'algorithme de Kernighan-Lin **[30]** qui, à partir de deux groupes de même taille, échange des nœuds entre eux pour optimiser une fonction de coût.

Citons enfin la bissection spectrale **[17]** qui cherche à établir une coupe, minimale en nombre d'arêtes, qui sépare un graphe en deux groupes égaux en taille. Cette coupe peut s'écrire en utilisant la matrice Laplacienne, définie comme la différence entre la matrice de degré et la matrice d'adjacence, ce qui transforme le problème en recherche de vecteurs propres d'une matrice.
Ces méthodes ne sont pas directement exploitables pour la détection de communautés, mais elles fournissent des principes s'appliquant à ce problème, par exemple le déplacement de nœud en optimisant une fonction avec l'algorithme de Kernighan-Lin ou la coupe minimale et la matrice Laplacienne avec la bissection spectrale.

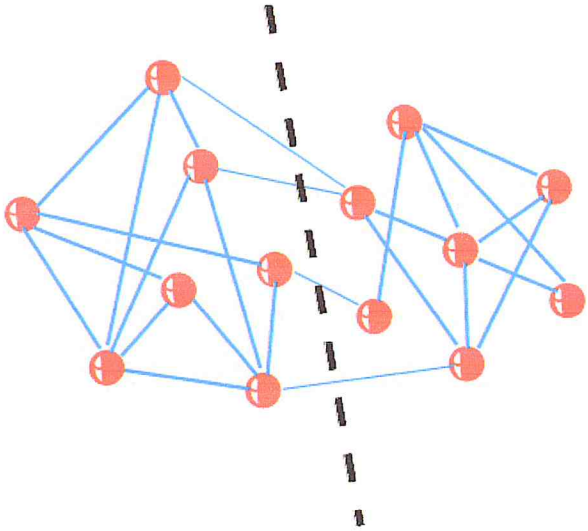


Figure 2.1 - Partitionnement de graphe.

La coupe montre la partition en deux clusters de taille égale.

2.1.2 Clustering hiérarchique

Dans ce modèle, les clusters sont formés de sous-groupe, l'ensemble constituant une vision hiérarchique d'un graphe que l'on peut représenter par un dendrogramme (**Figure 2.1**).

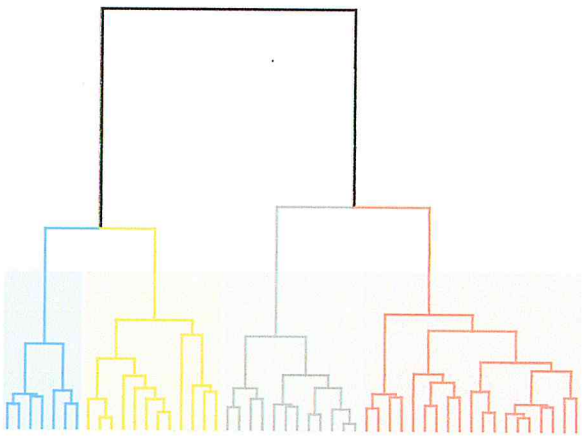


Figure 2.2 - Représentation d'un clustering hiérarchique sous la forme d'un dendrogramme.

Chaque fourche représente un groupe constitué de sous-groupes ou de nœuds du graphe. En général, on sait très peu de choses sur la structure de la communauté d'un graphe. Il est rare de

connaitre le nombre de clusters dans lequel le graphe est divisé, ou d'autre indication sur l'appartenance des nœuds.

Dans ces cas, les procédures de clustering telles que les méthodes de partitionnement peuvent difficilement être utiles, et on est forcé de faire des suppositions raisonnables sur le nombre et la taille des clusters, qui sont souvent injustifiés. D'un autre coté le graphe peut avoir une structure hiérarchique, c'est-à-dire : qu'il peut afficher plusieurs niveaux de clustering des nœuds, avec des petits clusters inclus dans de grands clusters qui sont à leur tour inclus dans les grands clusters, et ainsi de suite, les réseaux sociaux ont souvent une structure hiérarchique.

Dans tels cas, on peut utiliser des algorithmes de clustering hiérarchiques, c'est-à-dire des techniques de clustering qui révèlent la structure multiniveau du graphe. Le clustering hiérarchique est très courant dans l'analyse des réseaux sociaux.

Le point de départ de toute méthode de clustering hiérarchique est la définition d'une mesure de similarité entre les sommets. Après avoir choisir une mesure on calcule la similarité pour chaque paire de sommets, qu'ils soient connectés ou non. A la fin de processus, on se trouve avec une nouvelle matrice $\mathbf{X}$: $\mathbf{n} \times \mathbf{n}$, « la matrice de similarité ». Ensuite, un algorithme exploite cette matrice pour construire la hiérarchie selon une des deux techniques : agglomération et division.

La similarité de nœud peut être une distance dans un espace euclidien, Il existe plusieurs façons de définir la distance entre deux communautés. La plus simple (single linkage) considère que la distance entre deux communautés est la distance minimale entre deux sommets de celles-ci. A l'opposé, on peut considérer la distance maximale (complete linkage). De manière intermédiaire (average linkage) on peut considérer que la distance entre deux communautés est la moyenne des distances entre chaque paire de leurs sommets.

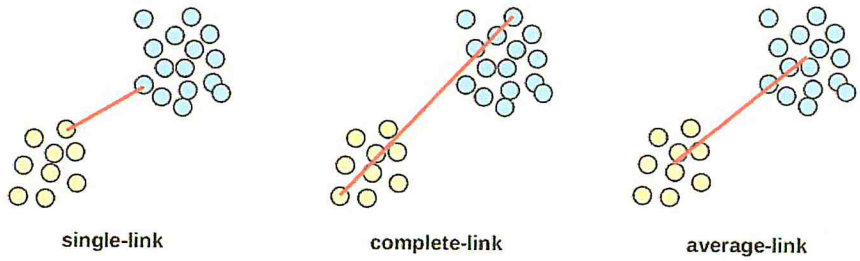


Figure 2.3 - Les trois types de « linkage » de clustering hiérarchique.

2.1.2.1 Définition de la similarité

- Chaque sommet est dans un cluster dont les sommets sont les plus simple à lui.
- Les mesures de similarité sont à la base des méthodes « approches classiques ».
- Certaines mesures populaires de similarité :
  - Distance Euclidienne (L_1_norm).
  - Distance de Manhattan (L_2_norm).
  - L_∞_norme.
  - Cosine similarity ……

Et comme qu'on a dit déjà que l'exploitation de la matrice de similarité pour construire la hiérarchie est réalisée selon les deux techniques d'agglomération et/ou division, ces techniques de classification visent à identifier des clusters avec une forte similarité, et peuvent être classées en deux classes d'algorithmes :

1. Algorithmes agglomératifs : dans lesquels les clusters sont itérativement fusionnés si leur similarité est suffisamment élevée.
2. Algorithmes par division : dans lesquels les clusters sont divisés itérativement en supprimant les arêtes reliant les sommets à faible similarité.

Les deux classes se réfèrent à des processus opposés :

- Les algorithmes agglomératifs : sont ascendants « bottom-up » comme on part de sommets en tant que des clusters séparés (singletons), et se termine avec le graphe comme un cluster unique.
- Les algorithmes par division : sont descendants « top-down » car ils suivent la direction opposée.

Les méthodes de clustering hiérarchique peuvent être adaptées à la détection de communautés à condition de trouver une mesure de similarité pertinente. Cependant, plusieurs difficultés ont été observées rendant délicate l'utilisation de ces méthodes dans tous les cas de réseaux complexes. Tout d'abord, des nœuds peuvent être mal placés dans un groupe, certains en dehors du groupe dans lequel ils jouent un rôle central, d'autres connectés à un seul voisin se retrouvent seuls dans une communauté. Ensuite, la structure hiérarchique n'est pas nécessairement pertinente si le graphe

ne possède pas cette structure intrinsèquement. Enfin, les méthodes fondées sur une distance ne permettent pas le traitement de très grands réseaux complexes du fait de leur complexité en temps qui peut atteindre O (n^2 log(n)).

2.1.3 Clustering de partition

Il indique une autre de classe de méthodes populaires pour trouver des clusters dans un ensemble de points de données, le nombre de clusters est déjà sélectionné disons **K**. Les points dont intégrés dans un espace métrique, de sorte que chaque sommet est un point et qu'une mesure de distance est définie entre des paires de points dans l'espace, la distance est une mesure de dissimilarité entre les sommets. Le but est de séparer les points en **K** clusters de manière à maximiser/minimiser une fonction de cout donnée basée sur des distances entre les points et/ou des points de centroides, c'est-à-dire des positions définies de manière appropriées dans l'espace **[54]**.

Certaines fonctions les plus utilisées sont énumérées ci-dessous :

- **Minimum K-clustering :** La fonction de cout ici est le diamètre d'un cluster, qui est la plus grande distance entre deux points d'un cluster, « les points sont classés de telle sorte que le plus grand **K** diamètre de cluster est le plus petit possible », l'idée est de garder les clusters très compacts.
- **K-clustering :** Identique au minimum K-clustering mais le diamètre est remplacé par la distance moyenne entre toutes les paires de points d'un cluster.
- **K-center :** Pour chaque cluster ***i*** on définit un point de référence x_i, le centroide, et on calcule le d_i maximum des distances de chaque point de cluster à partir de centroide.
  Les clusters et les centroides sont choisis de manière cohérente afin de minimiser la plus grande valeur de d_i.
- **K-median :**
  Identique à K-center, mais la distance maximale du centroide est remplacé par la distance moyenne.

La technique de partitionnement la plus populaire est K-means clustering **[66]**, ici la fonction totale de cout est la distance intra-cluster, ou la fonction d'erreur au carré.

$$\sum_{i=1}^{k} \sum_{X_j \in S_i} ||X_j - C_i||^2 \qquad (1)$$

S_i : sous-ensemble de points de i-éme cluster, et C_i son citroide.

Le problème des k-moyennes est NP_Difficile dans le cas général, peut être simplement résolu avec l'algorithme de Lloyd **[61]**. On commence à partir de d'une distribution initiale de centroides tels qu'ils sont aussi loin que possible de chacun d'eux. Dans la première itération, chaque sommet est assigné au centroide le plus proche. Ensuite les centres de masse de K clusters sont estimés et deviennent un nouvel ensemble de centroides, c'est-à-dire pour chaque centroide, déplacez le centroide à la moyenne des points assignés à ce centroide. Ce qui permet une nouvelle classification de sommets, et ainsi de suite.

Après un petit nombre d'itération, les positions des centroides sont stables et les clusters ne changent plus. La solution trouvée n'est pas optimale et dépend fortement d'un choix initial des centroides. Néanmoins, l'heuristique de Lloyd est restée populaire en raison de sa convergence rapide ce qui la rend appropriée pour l'analyse de grands ensembles des données.

Le résultat peut être amélioré en effectuant plus d'exécution à partir de différentes conditions initiales, et en choisissant la solution qui fournit la valeur minimale de la distance totale intra-cluster. Une autre technique populaire, similaire dans l'esprit au K-means clustering est Fuzzy K-means clustering **[67]**.

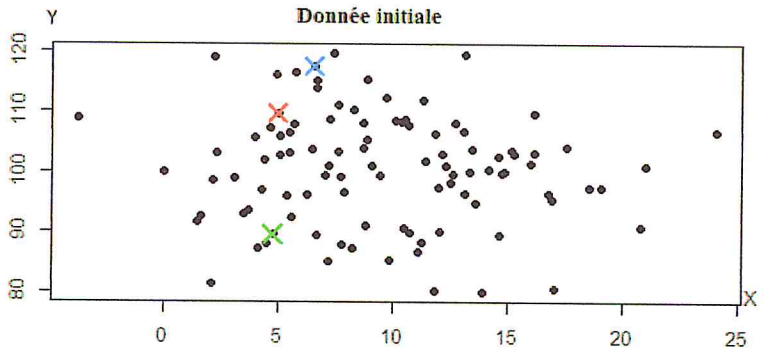


Figure 2.4 – Exemple avec 50 points de données avec trois centroïdes initiés aléatoirement.

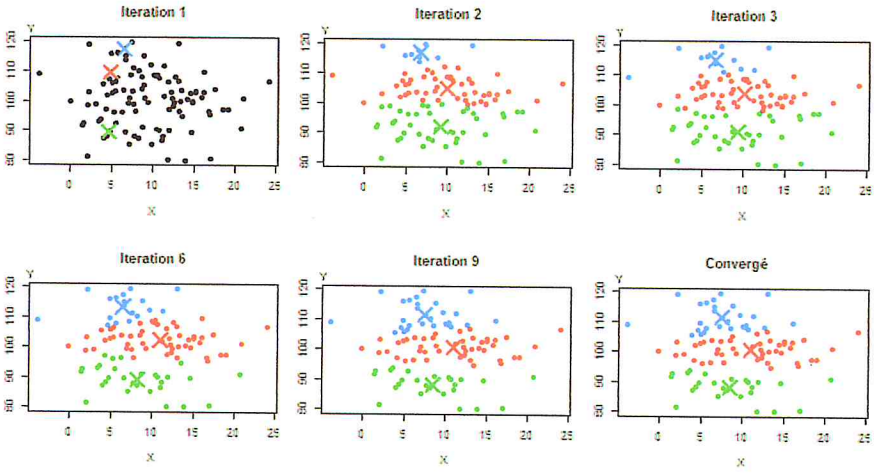


Figure 2.5 - Résultat obtenu après 9 itérations de K-means clustering.

L'itération 2 montre le nouvel emplacement des centres centroïdes. L'itération 3 a plus de points bleus lorsque les centroïdes se déplacent. En sautant à l'itération 6, nous voyons que le centroide rouge s'est déplacé plus vers la droite. L'itération 9 montre que la section verte est beaucoup plus petite que dans l'itération 2, le bleu a pris le dessus et le centroide rouge est plus mince que dans l'itération 6. Les résultats de la 9ème itération étaient les mêmes que ceux de la 8ème itération, elle a donc "convergé".

2.2 Les approches séparatives

Un moyen simple d'identifier les communautés dans un graphe consiste à détecter les arêtes qui relient les sommets des différents clusters et les enlever, afin que les clusters soient déconnectés les uns des autres, c'est la philosophie des algorithmes de divisions. Les arêtes sont retirées une à une, et à chaque étape les composantes connexes du graphe obtenu sont identifiées à des clusters. Le processus est répété jusqu'au retrait de toutes les arêtes. On obtient alors une structure hiérarchique de communautés (dendrogramme), comme pour les méthodes de clustering hiérarchique. Les méthodes existantes diffèrent par la façon de choisir les arêtes à retirer. La principale différence avec clustering hiérarchique est qu'on doit supprimer les arêtes entre les clusters au lieu des arêtes entre les paires de sommets à faible similarité (les approches séparatives s'inspirent du clustering hiérarchique mais sans utiliser la mesure de similarité entre nœuds), et il n'y a aucune garantie apriori que les arêtes inter-cluster connectent les sommets à faible similarité. Dans certain cas, les sommets (avec tous leurs arêtes adjacentes) ou les sous-graphes entiers

peuvent être supprimés, au lieu des arêtes simples. Il est habituel de représenter les parties résultantes au moyen de dendrogramme, dans les techniques de clustering hiérarchique.

2.2.1 Algorithme de Girvan et Newman

L'algorithme le plus populaire est celui proposé par Girvan et Newman. Cet algorithme est l'un des premiers algorithmes modernes pour la détection de communautés avec la création d'une mesure d'importance fondée sur les arêtes, la centralité d'intermédiarité. La centralité d'intermédiarité est une mesure de la centralité d'un sommet dans un graphe basé sur les chemins les plus courts. Elle est égale au nombre de fois que ce sommet est sur le chemin le plus court entre deux autres nœuds quelconques du graphe.

L'algorithme de Girvan-Newman étend cette définition au cas des arêtes, définir la (edge-betweenness) d'une arête en tant que nombre de chemins les plus courts entre toutes les paires de sommets qui courent le long du l'arête (**Figure 2.6**). C'est une extension aux arêtes, du concept populaire de l'intermédiarité de site, présenté par Freeman 1977 **[32]**, et qui exprime l'importance des arêtes dans des processus tels que la propagation de l'information, où l'information circule généralement sur des chemins les plus courts.

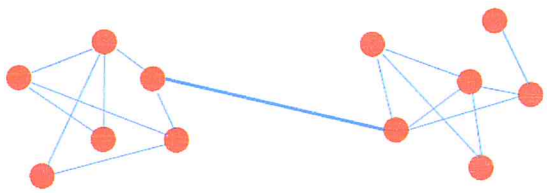


Figure 2.6 - La valeur de (Edge betweenness) la plus élevée pour les arêtes reliant les communautés.

Sur la figure précédente, l'arête au milieu a une distance entre les arêtes plus grande que toutes les autres arêtes, car tous les chemins les plus courts qui relient les sommets des deux communautés passent par cette arête.

Les étapes de l'algorithme pour la détection de la communauté sont résumées ci-dessous :

1. Calcule edge-betweenness pour tous les arêtes.

2. Enlèvement d'arête avec la plus forte edge-betweenness, c'est-à-dire qui ont le plus de chances d'être des ponts entre communautés, sont supprimées, en cas d'égalité avec d'autres arêtes l'un d'eux est choisi au hasard.
3. Recalcule edge-betweenness sur le graphe courant.
4. Itération du cycle de l'étape 2.

2.3 Les approches agglomérative

L'idée commune de toutes ces méthodes est d'utiliser une approche, s'apparentant à celle du clustering hiérarchique, elle consiste à fusionner des nœuds pour produire de super-nœuds, dans laquelle les sommets sont regroupés itérativement en communautés en partant d'une partition de n communautés composées d'un seul sommet. Les regroupements des communautés sont poursuivre jusqu'à obtenir une seule communauté regroupant tous les nœuds, et une structure hiérarchique de communauté (dendrogramme) et ainsi construite. Donc à la fin de processus, les super-nœuds (qui agglomèrent des groupes de nœuds connectés) représentent les communautés.

2.4 Les approches d'optimisation de la modularité

Newman introduit une notion de modularité **[60]**, il s'agit d'une fonction Q mesurant la qualité d'une partition du 'graphe en communauté', elle se base sur la proportion d'arêtes internes aux communautés et la proportion d'arêtes liés à chaque communauté, la modularité est définie comme :

$$\frac{1}{2m}\sum_{i,j}\left(A_{ij}-\frac{K_iK_j}{2m}\right)\delta(C_i,C_j) \qquad (2)$$

Où la somme passe sur toutes les paires de sommets, A est la matrice d'adjacence, m le nombre total d'arêtes du graphe, $K_i et\ k_j$ sont le degré des nœuds i et j respectivement, la fonction δ Donne 1 si les sommets i et j sont dans la même communauté $C_i = C_j$, 0 sinon. Par hypothèse, des valeurs élevées de modularité indiquent de bonnes partitions, des résultats empiriques montrent que les valeurs de modularité supérieures à 0,3 indiquent un bon partitionnement. Ainsi la partition correspondante à sa valeur maximale sur un graphe donné devrait être la meilleure. C'est la principale motivation pour la maximisation de modularité, et de loin c'est la classe des méthodes la plus populaire pour détecter les communautés dans les graphes.

L'optimisation de modularité est un problème NP_Complet **[33]**, il est donc probablement impossible de trouver la solution dans un temps polynomialement croissant avec la taille du graphe. Parmi les méthodes qui tente d'optimiser la modularité d'une partition de réseau, (Greedy techniques **[34]**, Simulated annealing **[35]**, External optimization **[36]**). Cependant il existe actuellement plusieurs algorithmes capables de trouver des approximations assez bonnes pour maximiser la modularité dans un délai raisonnable.

2.4.1 Algorithme de Louvain

L'algorithme de Louvain (Blondel et al., 2008) **[57]**, est un algorithme simple, efficace et facile à mettre en œuvre pour identifier les communautés dans les grands réseaux. L'algorithme a été utilisé avec succès pour des réseaux de différents types et pour des tailles allant jusqu'à 100 million de nœuds et des milliards de lien, l'analyse d'un réseau typique de 2 million de nœud prend 2 minutes sur un pc standard. L'algorithme dévoile les hiérarchies des communautés et permet de zoomer au sein des communautés pour découvrir des sous-communautés, sous-sous-communautés etc. C'est aujourd'hui l'une des méthodes les plus utilisées pour détecter les communautés dans les réseaux à grande échelle. L'algorithme de Louvain est une méthode d'optimisation gloutonne (Greedy technique), qui tente d'optimiser « maximisation », la modularité d'une partition du réseau. L'optimisation est effectuée en 2 étapes :

Tout d'abord, la méthode cherche des petits clusters en optimisant la modularité localement.

Deuxième, elle agrège les nœuds appartenant à la même communauté et construit un nouveau réseau dont les nœuds sont les communautés.

Ces étapes sont répétées de façon itérative jusqu'à ce qu'un maximum de modularité soit atteint et qu'une hiérarchie de communauté soit produite.

2.5 Les approches alternatives

Dans cette section, nous décrivons un algorithme qui ne rentre pas dans les catégories précédentes, une méthode simple et rapide basée sur la propagation de label **[37]**.

2.5.1 Propagation de label

La méthode de propagation de labels (Raghavanet al., 2007) **[38]** est basée sur la transmission d'un label d'un nœud à ses voisins. Un état d'équilibre est atteint lorsque chaque nœud a son label égal à celui de la majorité de ses voisins. Soit un graphe G= (V; E), à chaque étape, chaque nœud met à jour son label selon les labels de ses voisins, en utilisant un vote. Le label du nœud x prendra le label majoritaire de ses voisins. En notant C_x le label du nœud x, et par $N^l(x)$ l'ensemble du voisinage du nœud x avec le label l, l'affectation d'un label au nœud x est donnée par la formule suivante :

$$C_x = \arg\max_l |N^l(x)|. \qquad (3)$$

A la fin du processus, les nœuds ayant le même label représentent une communauté. Cette méthode peut être effectuée de manière synchrone ou asynchrone. La méthode asynchrone signifie que la mise à jour d'un label d'un nœud est connue par tous les autres nœuds du graphe immédiatement. Son label est utilisé pour la mise à jour des labels des autres nœuds. Ce qui n'est pas le cas du mode asynchrone, où la mise à jour des labels utilise les labels des nœuds à la précédente propagation.

Les étapes de l'algorithme de propagation de label :

1. Chaque nœud est initialisé avec un label unique.
2. De manière successive, chaque nœud *i* remplace son label par celui utilisé par le maximum de ses voisins (ou un choisi au hasard en cas d'égalités). Après un certain nombre d'itérations, le même label tend à être associé aux membres d'une même communauté.
3. Si chaque nœud ne change plus son label (si ce nœud ainsi le nombre maximal de ses voisins ont même label), alors arrêter l'algorithme.
4. Tous les nœuds ayant le même label forment une communauté.

Cet algorithme présente l'avantage d'avoir une complexité permettant de travailler sur de grands graphes. Cependant, l'algorithme de propagation de labels présente l'inconvénient d'être instable, ne donne que rarement le même résultat après plusieurs lancements. Ce problème peut s'expliquer par deux raisons. La première est le choix parfois aléatoire du label que doit prendre un nœud lorsqu'il y a plusieurs labels majoritaires dans son voisinage. La seconde porte sur les tailles des structures. En effet, la propagation de labels prendra moins de temps à couvrir de petits réseaux que de gros réseaux.

2.6 Tableau récapitulatif de complexité des méthodes disjointes

Nous avons vu quelques algorithmes de détection de communautés disjointes, un tableau récapitulatif, Tableau 1, permettant d'observer la complexité algorithmique de chaque algorithme disjoint.

Algorithme	Complexité	Description	Réf
Partitionnement de graphe		• Stables, déterministes. • La spécification de nombre de communautés et la taille est nécessaire.	[54]
Algorithme de Kernighan-Lin	O (n^2 log n)		[30]
Bissection spectrale	O (n^3)		[31]
Clustering hiérarchique	O (n^2 log n)	• Nécessite pas de connaissances préliminaires sur le nombre et la taille des communautés. • Les résultats de la méthode dépendent de la mesure de similarité spécifique adoptée.	[54]
Clustering de partition		• Complexité réduite. • Nombre de communautés doit être spécifié au début.	[54]
K-means clustering	O (nkdi)		[66]
Fuzzy K-means clustering.	O (nk^2di) d : la dimension, k : nombre de clusters, i : nombre d'itérations		[67]
Les approches séparatives			
Algorithme de Girvan et Newman	O (n^3)	• Bonne optimisation de la modularité. • l'algorithme est assez lent.	[34]
Les approches d'optimisation de la modularité		• Meilleure qualité de partitionnement.	[54]

Algorithme de Louvain	O (n log n)	• Des résultats très satisfaisant en terme de modularité avec une optimisation maximale. • Instable, non déterministe	[58]
Les approches alternatives			
Propagation de label	O ($n * m$)	• Instable, non déterministe. • Rapide, complexité réduite	[38]

Tableau 1 – Comparaison entre les principaux algorithmes de détection de communautés disjointes.

3 Détection de communautés chevauchantes

La section précédente nous a permis d'explorer les méthodes pour la détection de communautés disjointes. Nous allons lister ici les principales approches qui ont été proposées à ce jour. Bien que la liste soit importante, elle est non exhaustive afin d'en limiter la longueur, nous n'avons retenu que les approches qui ont reçu le plus d'attention de la part de la communauté scientifique. Notre but est de donner une vue d'ensemble des méthodes proposées, et d'en illustrer la diversité.

3.1 Percolation de cliques (Clique Percolation Method)

L'algorithme CPM **[59]** est l'un des algorithmes les plus classiques de détection de communautés chevauchantes. CPM est une solution qui a l'avantage d'être conceptuellement simple. Elle prend un paramètre, k, qui représente la taille des cliques qui est le seul paramètre d'entré de l'algorithme CPM.

Une clique est un sous-ensemble de sommets d'un graphe dont le sous-graphe induit est complet, c'est-à-dire que deux sommets quelconques de la clique sont toujours adjacents. Une K-clique est un sous-ensemble de k sommets tous adjacents (sous-graphe complet), et deux k-cliques sont adjacentes si elles partagent k-1 sommets comme illustré dans la figure suivante.

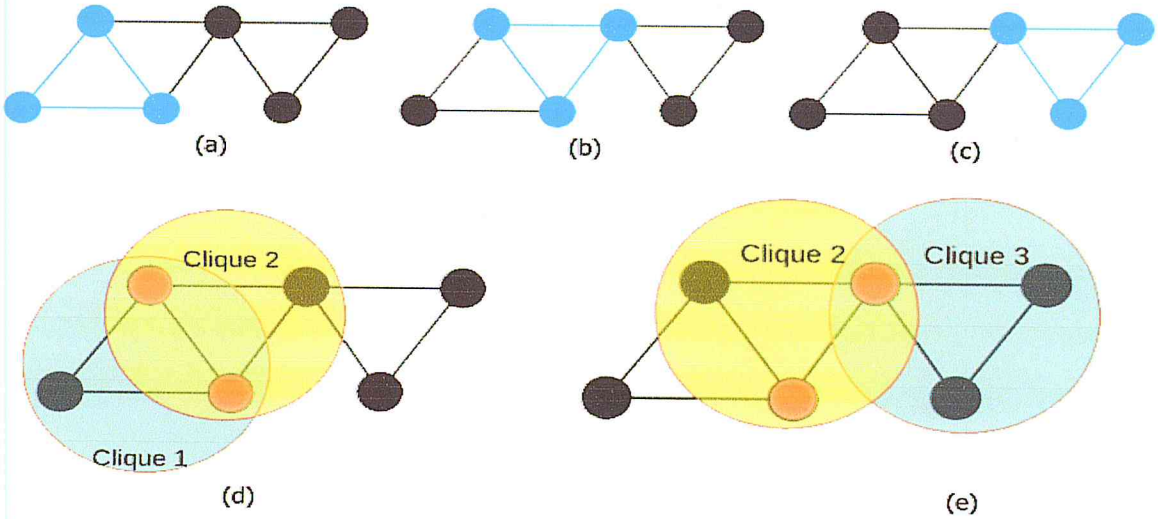


Figure – 2.7 : Exemple de graphe pour k=3 possédant trois 3-clique (en bleu).

Les trois sommets de ces sous-graphes (a), (b) et (c) sont tous adjacents deux-à-deux. Deux 3-cliques adjacents (d) et (e) partagent k-1=2 nœuds (en orange).

La technique de CPM est basée sur le concept que les liens internes de la communauté sont probable de former des cliques en raison de leur forte densité. D'un autre côté, il est peu probable que les liens entre les communautés forment des cliques.

Les auteurs observent qu'une communauté peut être définie comme une chaîne de k-cliques adjacentes. Cette méthode permet la détection de communautés chevauchantes où un sommet peut appartenir à plusieurs k-cliques. Par défaut, l'algorithme calcule les solutions pour toutes les valeurs de k, et laisse l'utilisateur choisir celle qui lui semble la plus pertinente, au vu des résultats obtenus. Pour la plupart des réseaux à grande échelle, une valeur de k=4 est la plus efficace et donne les résultats en termes de qualité de partitionnement les plus probants. Cela signifiera que tout nœud n'appartenant pas à au moins une clique de taille au moins 4 ne sera classifié dans aucune communauté. Si le graphe est peu dense, il est parfois utile de prendre une valeur de k=3.

L'algorithme fonctionne en deux étapes : premièrement, trouver dans le réseau toutes les cliques de taille exactement k. Deuxièmement, pour chaque clique trouvée, si elle a k-1 nœuds en commun (adjacent k-cliques) avec une autre clique de taille k, mettre les 2 cliques dans la même communauté. Une communauté est donc formellement définie comme un ensemble de cliques dans

lequel on peut passer de l'une choisie au hasard à n'importe quelle autre par une suite de déplacements, où un déplacement consiste à choisir une autre clique ayant k-1 nœuds communs avec cette clique.

Le principe de cette méthode paraît simple, elle fonctionne remarquablement bien dans les réseaux à grande échelle. Ceci est lié au fort taux de clustering au sein des communautés, et en particulier le grand nombre de fermetures transitives qu'elles contiennent dans la plupart des réseaux à grande échelle, et elle peut selon la paramétrisation de k être appliquée à des graphes de plusieurs centaines de milliers d'arêtes ainsi elle est capable de détecter un grand nombre de communautés chevauchantes, parfois avec de nombreux nœuds.

En revanche, les résultats en termes de qualité de partitionnement sont encourageants mais elle souffre de certaines faiblesses : dans des graphes très denses, elle peut avoir une complexité et une consommation mémoire prohibitives.

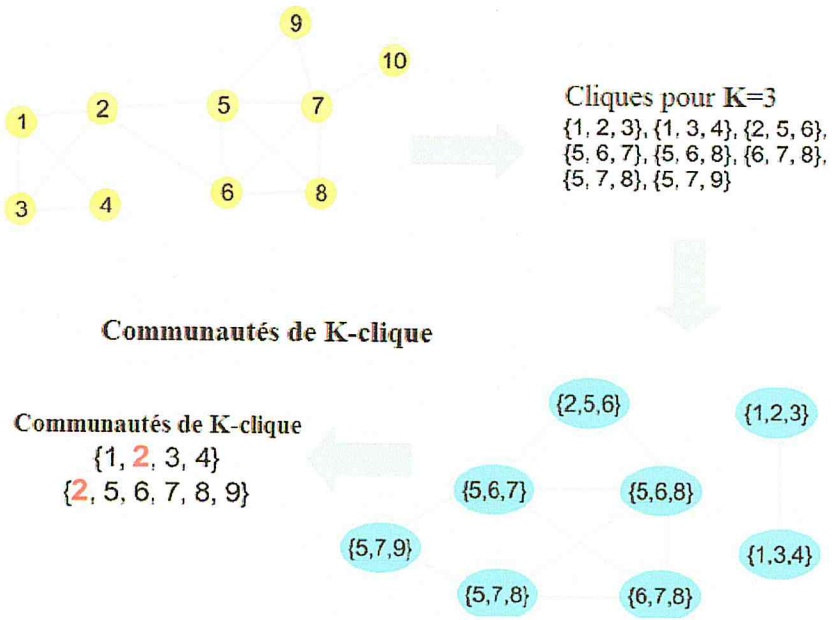


Figure 2.8 - Illustration de la détection de communautés chevauchantes par algorithme (CPM).

Les étapes sont **1)** Détection des k-cliques. **2)** Former le graphe de cliques. **3)** Fusionner les membres des composantes connexes dans le graphe de cliques pour obtenir les communautés k-cliques. Dans cet exemple, le nœud 2 est partagé par les deux communautés formées, résultant une structure chevauchante.

3.2 Graphe de liens et partitionnement de liens

Il a été récemment suggéré que la définition des clusters en tant qu'ensembles d'arêtes, plutôt que des nœuds, pourrait être une stratégie prometteuse pour analyser des graphes avec des communautés qui se chevauchent **[47]** et **[48]**. On doit se concentrer sur le graphe de lien L(G) **[49]**, c'est-à-dire le graphe dont les nœuds sont les liens du graphe original. La transformation consiste à faire correspondre, à chaque lien du graphe originel G, un nœud dans le graphe transformé L(G). Un lien existe entre deux nœuds de L(G) si et seulement si les liens correspondants dans G avaient une extrémité commune. On peut observer un exemple de transformation sur la figure suivante.

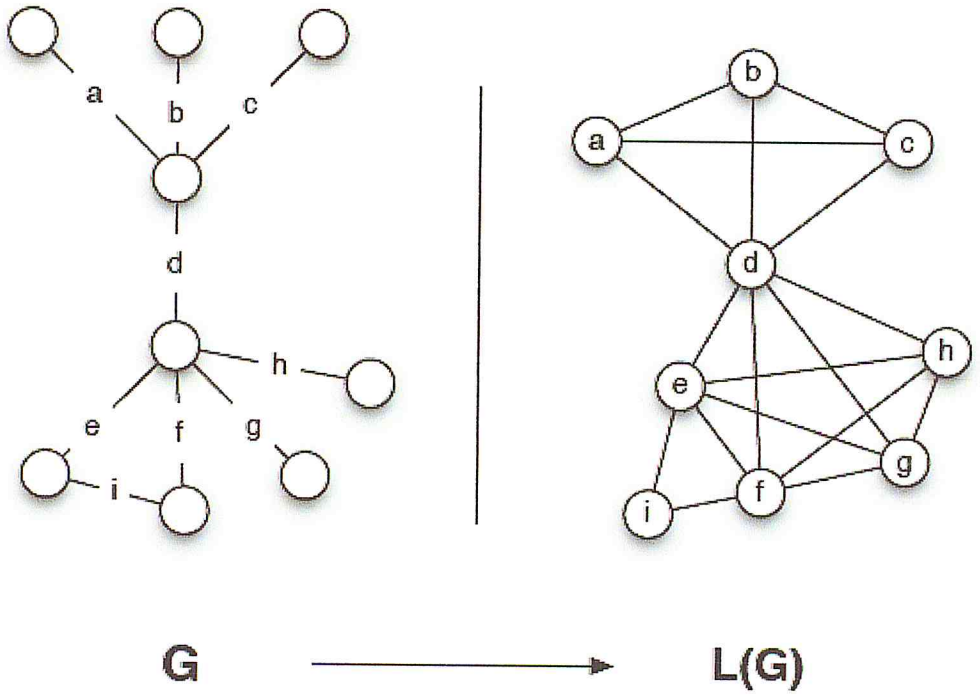


Figure 2.9 - Exemple de transformation d'un graphe en graphe de lien.

Une fois le graphe de lien obtenu, n'importe quelle méthode disjointe peut être utilisée pour détecter les communautés.

L'idée de partitionnement des liens au lieu de nœuds pour découvrir la structure de la communauté a été exploré aussi, partitionner un graphe de lien signifie grouper les arêtes du graphe original. Un nœud dans un graphe est appelé chevauché si ses liens sont mis dans plus d'un cluster.

Dans (Y.-Y. Ahn et al (2010)) **[50],** les liens sont partitionnés via une technique de clustering hiérarchique agglomératifs appelée clustering hiérarchique de lien, basant sur la similarité entre les liens pour construire un dendrogramme. Étant donné une paire de liens e_{ik}et e_{jk} connecté au nœud k, une mesure de similarité peut être calculée par Indice de Jaccard **[51]** défini comme :

$$S(e_{ik}, e_{jk}) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|} \qquad (4)$$

Où N_i est le voisinage du nœud i incluant i, une méthode de clustering hiérarchique qui est single-linkage clustering (clustering agglomératifs, à chaque étape combiner deux clusters qui contiennent la paire d'éléments les plus proches) est ensuite utilisé pour construire le dendrogramme où chaque feuille est un lien du graphe original et les branches représentent des communautés de liens. Couper ce dendrogramme à un certain seuil donne des communautés de liens à plusieurs niveaux en coupant ce dendrogramme à différents seuils. Chaque nœud hérite de toutes les adhésion « memberships » de ses liens et peut donc appartenir à plusieurs communautés qui se chevauchent.

Le dendrogramme fournit une hiérarchie riche de la structure, mais pour obtenir les communautés les plus pertinentes, il est nécessaire de déterminer le meilleur niveau auquel couper le dendrogramme. Pour ce faire, une fonction objective naturelle a été introduite, la densité de partition, **D**, basée sur la densité de liens au sein des communautés. Par exemple le seuil avec une densité de partition maximale.

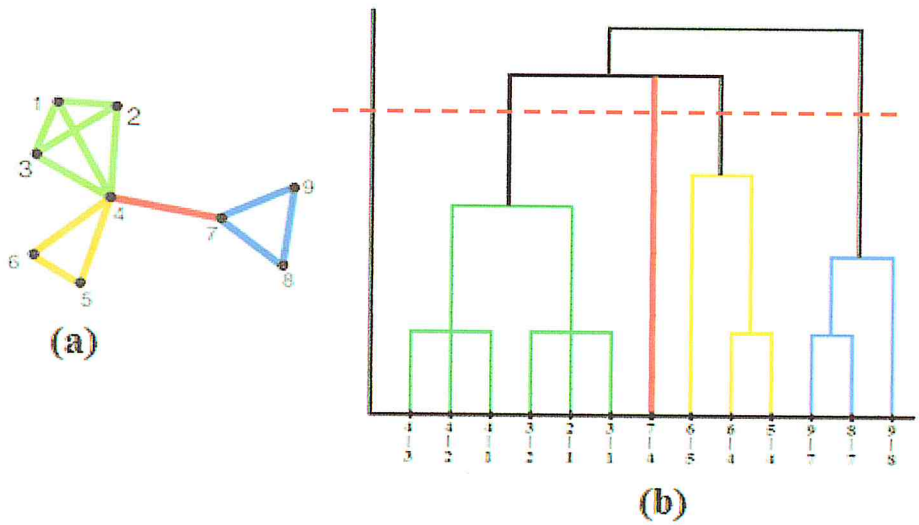


Figure 2.10 – Exemple d'un partitionnement de liens.

Pour graphe **(a)**, **(b)** est le dendrogramme obtenu après le calcul de la similarité entre les liens. Le découpage de dendrogramme (Ligne pointillée en rouge) dans un certain seuil de densité a résulté les communautés suivantes : {4-3, 4-2, 4-1, 3-2, 2-1, 3-1}, {7-4}, {6-5, 6-4, 5-4}, {9-7, 8-7, 9-8}, le nœud 4 est partagé dans les communautés en vert et en jaune donc 4 est un nœud chevauché.

La densité de partition dans un réseau avec M liens et N nœuds, la partition $P = \{P_1, \dots, P_c\}$ est une partition des liens en C sous-ensembles. Le nombre de liens dans le sous-ensemble P_c est $m_c = |P_c|$. Le nombre de nœuds induits, « tous les nœuds que ces liens touchent » est $n_c = \left|\bigcup_{e_{ij} \in P_c}\{i,j\}\right|$.

La densité de partition locale d'une communauté de lien c, est :

$$D_c = \frac{m_c-(n_c-1)}{\frac{n_c(n_c-1)}{2}-(n_c-1)} \qquad (5)$$

C'est le nombre de liens dans P_c normalisés par le nombres minimum et maximum des liens possibles entre ces nœuds.

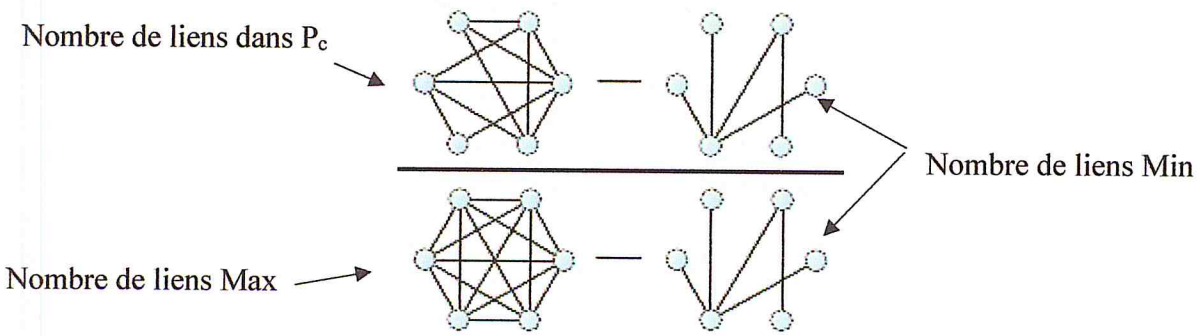


Figure 2.11 – Calcul de La densité de partition locale d'une communauté.

La densité de partition de l'ensemble du réseau est la moyenne de Dc :

$$\sum_{c=1}^{|P|} \frac{m_c}{M} D_c \qquad (6)$$

La figure suivante montre un exemple qui illustre comment la densité de partition est calculée.

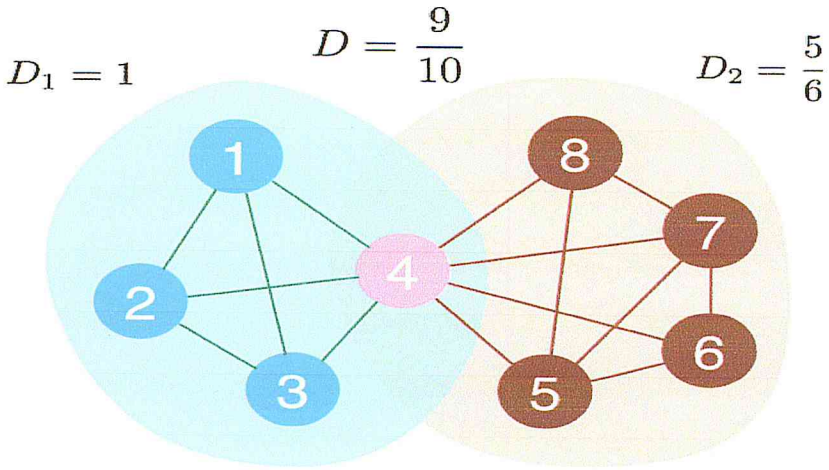


Figure 2.12 - Définition de la densité de partition.

La densité de partition locale des nœuds bleus D_1 est un car c'est une clique, alors que celle des nœuds rouges D_2 est inférieure à un. La densité de partition totale D de la structure de la communauté est la moyenne de deux densités de partition locales, $D = 0{,}9$.

Calculer D à chaque niveau du lien dendrogramme nous permet de choisir le meilleur niveau à couper. La méthode est capable de trouver des clusters significatifs dans les réseaux biologiques, ainsi que dans un réseau social de communications de téléphonie mobile. Elle peut également être étendue aux graphes pondérés.

Il existe d'autres méthodes de partitionnement des liens comme l'approche de (Evans et Lambiotte (2009, 2010)) **[52,53]** où le graphe original transformé en un graphe de liens pondéré, dont les nœuds sont les liens du graphe original. Ensuite, des algorithmes de détection de communauté disjoints peuvent être appliqués. La partition de nœud d'un graphe de liens conduit à une partition de lien du graphe original.

Bien que le partitionnement de lien pour la détection de chevauchement semble conceptuellement naturel et l'idée de grouper les liens est sûrement intéressante, il n'y a aucune garantie qu'il fournit une détection de meilleure qualité que la détection par nœud fait **[54].**

3.3 Extension local et optimisation

Dans les communautés chevauchantes, les communautés ne sont plus limitées les unes par les autres. Il devient alors possible de constituer chaque communauté indépendamment. Plusieurs solutions ont été proposées pour ce faire, basées sur un même principe en deux étapes :

1. Trouver des graines (seeds), c'est-à-dire des communautés initiales simples mais imparfaites.
2. Modifier ces graines, en les entendant et finalement en les contractant, jusqu'à obtenir les communautés chevauchantes recherchées.

Pour certains, les graines peuvent être des communautés trouvées par un algorithme de détection de communautés disjointes.

Dans Lancichinetti et al **[55]**, ont conçu une méthode basée sur l'extension locale et optimisation, l'hypothèse de base derrière leur algorithme est que les communautés sont essentiellement des structures locales, impliquant que les nœuds appartenant aux clusters eux-mêmes plus un voisinage étendu d'entre eux (extension local).

Cette méthode est basée sur l'optimisation d'une fonction de force (community strength), donc Ici, une communauté est un sous-graphe identifié par la maximisation d'une propriété de ses nœuds. La fonction pourrait en principe être arbitraire, dans leurs applications les auteurs ont choisi un simple essai basé sur le compromis entre le degré interne et le degré total du groupe.

La fonction de force (7) est définie donc comme le ratio entre le degré interne et la somme des degrés interne et externe des nœuds de la communauté. Ajouter ou retirer des nœuds de ces communautés peut augmenter la valeur de la fonction.

$$strength = \frac{C^{int}}{(C^{int}+C^{ext})^{\alpha}} \qquad (7)$$

Où C^{int} et C^{ext} sont le degré total interne et externe de la communauté, α est un paramètre positif à valeur réelle, appelé paramètre de résolution, qui permet de faire varier et contrôler la taille des communautés. Un paramètre α d'une valeur inférieure à 0,5 conduira à de très larges communautés (dans la plupart des cas, pour $\alpha < 0.5$ il n'y a qu'une seule communauté), tandis que des valeurs de α supérieures à 2 conduisent à des communautés de très petites tailles, un choix naturel est $\alpha = 1$.

Cette méthode développe une communauté à partir d'un nœud de graine aléatoire pour former une communauté naturelle et étendue, on continue d'ajouter et de retirer les nœuds voisins de la

communauté jusqu'à ce que la fonction force soit localement maximale, la fonction est recalculée après chaque ajout ou suppression d'un nœud, **figure 2.13** montre un exemple de communauté naturelle pour un nœud. Après avoir trouvé une communauté, sélectionner aléatoirement un autre nœud qui n'a pas encore été attribué à aucune communauté pour développer une nouvelle communauté. Toutes ces communautés forment ensemble la structure de la communauté qui se chevauche dans le réseau. Les étapes de cet algorithme sont les suivantes :

1. Choisir un nœud A aléatoirement.
2. Détecter la communauté naturelle du nœud A (optimisation de la force de communauté).
3. Choisir un nœud B aléatoirement qui n'est pas encore attribué dans une autre communauté.
4. Détecter la communauté naturelle du nœud B (optimisation de la force de communauté).
5. Répéter les étapes 3 et 4 jusqu'à ce que tous les nœuds soient attribués à au moins une communauté.

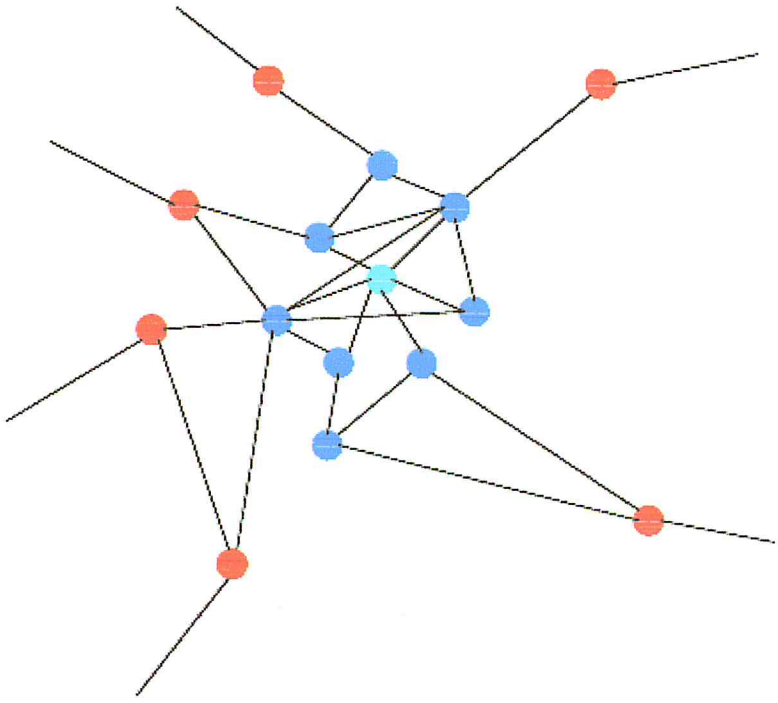


Figure 2.13 - Exemple schématique de communauté naturelle pour un nœud.

Selon la définition précédente, on commence par le nœud bleu ciel. Les nœuds bleus sont les autres membres du groupe et ont une valeur positive de la fonction force à l'intérieur du groupe, tandis que les nœuds rouges ont tous une valeur négative par rapport au groupe.

La méthode dépend de manière significative du paramètre de résolution α et la complexité de cas le plus défavorable est $O(n^2)$. Le problème est que certaines communautés risquant de ne pas être détectées en cas de chevauchement très important. Certaines communautés peuvent aussi au final être très semblables.

3.4 Méthodes à base de propagation de labels

Nous présentons deux principaux algorithmes pour la détection de communautés chevauchantes à base de propagations de labels.

3.4.1 COPRA

Dans l'algorithme de propagation de label, un label d'un nœud identifie une seule communauté à laquelle appartient ce nœud. Si les communautés se chevauchent, chaque nœud peut appartenir à plusieurs communautés. Par conséquent, pour trouver des communautés qui se chevauchent, nous devons clairement permettre à un label d'un nœud de contenir plus d'un seul identifiant de communauté **[44]**.

On donne pour chaque nœud x un ensemble de paires (c, b), où c, est un identifiant de communauté et b un coefficient d'appartenance, indiquant la force de l'appartenance de nœud x à la communauté c, de sorte que la somme de tous les coefficients d'appartenance pour x égale à 1.

Chaque étape de propagation mettrait label de nœud x à l'union des labels de ses voisins, additionne les coefficients d'appartenance des communautés sur tous les voisins et normaliser.

Plus précisément, en supposant une fonction $b_t(c, x)$ qui met en correspondance un sommet x et un identifiant de communauté c avec son coefficient d'appartenance à l'itération t,

$$b_t(c, x) = \frac{\sum_{y \in N(x)} b_{t-1}(c, x)}{|N(x)|} \qquad (8)$$

Où N (x) désigne l'ensemble des voisins de x.

Cette méthode utilise la mise à jour synchrone, parce qu'elle donne de meilleurs résultats que la mise à jour asynchrone **[56]**: donc, le label d'un nœud en itération t est toujours basée sur les labels de ses voisins en itération précédente t-1.

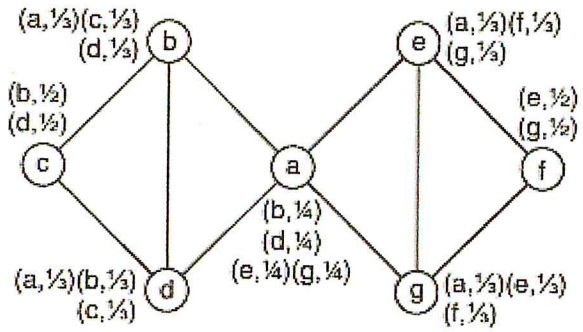


Figure 2.14 - Propagation des labels: première itération.

La figure montre le résultat de la première itération, cette méthode est toujours inadaptée comme algorithme de détection de communauté, car elle produit autant de communautés et converge vers une solution dans laquelle tous les sommets ont le même label : ici {(a,0.248), (b,0.188), (c,0.188), (d,0.188), (e,0.188), (f,0.188)} est le label de tous les nœuds.

Ce qui est requis est un moyen de conserver plus d'un identifiant de communauté dans chaque label sans les conserver tous. La méthode de copra utilise les coefficients d'appartenance à cet effet: lors de chaque étape de propagation, construire d'abord le label de nœud comme ci-dessus, puis supprimer les paires dont le coefficient d'appartenance est inférieur à un certain seuil. Ce seuil est exprimé comme une réciproque, $1/v$, où v est le paramètre de l'algorithme, utilisé pour contrôler le nombre maximum de communautés avec lesquelles un nœud peut être associé.

Il est possible que toutes les paires d'un label ont un coefficient d'appartenance inférieur au seuil $1/v$. Si c'est le cas, conserver seulement la paire qui a le plus grand coefficient d'appartenance, et supprimer tous les autres.

Si plus d'une paire a le même coefficient d'appartenance maximum, en dessous du seuil, alors conserver l'un deux par une sélection aléatoire. Cette sélection aléatoire rend l'algorithme non déterministe.

Après avoir supprimé les paires de label, nous le normalisons en multipliant le coefficient d'appartenance de chaque paire restante par une de sorte que la somme de tous les coefficients d'appartenance pour x égale à 1.

En utilisant cette méthode, avec v = 2, sur le réseau ci-dessus donne les résultats montrés dans la figure suivante.

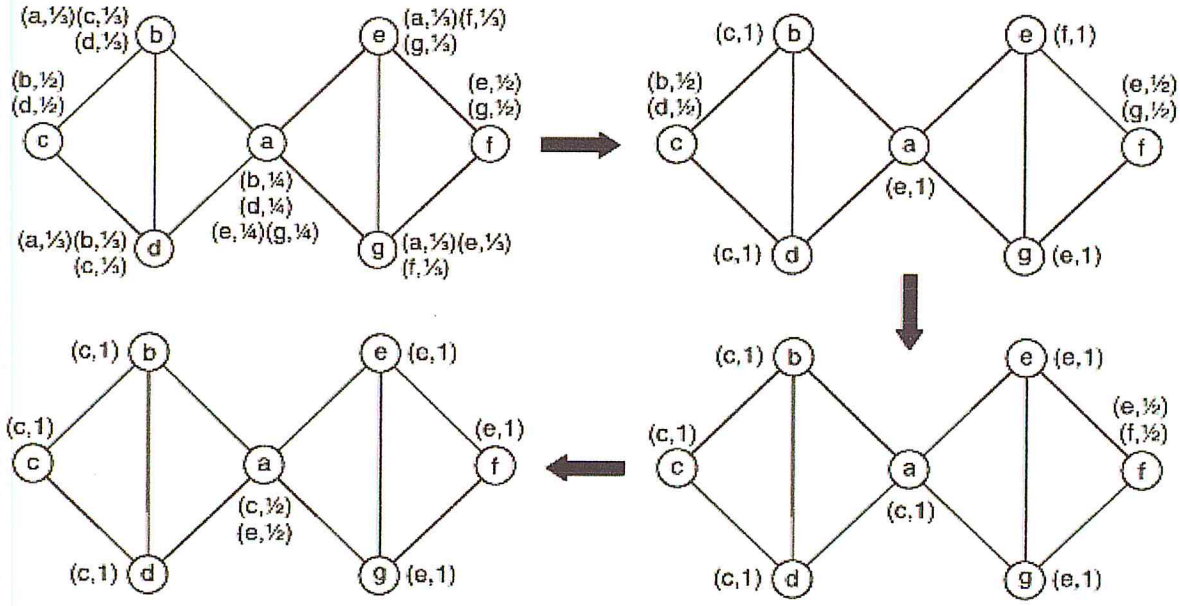


Figure 2.15 - Propagation de labels avec v = 2.

Dans la première itération, le nœud c est identifié par un label avec les identifiants de communauté b et d, chacun ayant le coefficient d'appartenance $^1/_2$. Parce que ce n'est pas moins que le seuil $^1/_2$, les deux sont conservés. De même, f est marqué avec e et g.

Les cinq autres nœuds ont tous au moins trois voisins, et leurs coefficients d'appartenance sont tous inférieurs au seuil ($^1/_3 < {}^1/_2$). Par exemple, b est identifié d'abord avec label $\{(a\,,{}^1/_3)\,,(c\,,{}^1/_3)\,,(d\,,{}^1/_3)\}$: nous choisissons aléatoirement c, supprimons a et d et normaliser en $(c\,,\ 1)$. Les labels pour a, d, e et g sont également choisies aléatoirement.

Avant l'itération finale, le nœud a a deux voisins identifiés par c et deux identifiés par e, et donc il conserve les deux identifiants de communauté: $\{(c\,,{}^1/_2)\,,(e\,,{}^1/_{23})\}$.

La solution finale contient donc deux communautés qui se chevauchent: $\{a, b, c, d\}$ et $\{a, e, f, g\}$.

L'algorithme de copra généralise l'algorithme de propagation de label LPA. Si v < 2 ils sont essentiellement les mêmes: label d'un nœud ne peut contenir qu'un identifiant de communauté, chaque étape de propagation conservant l'identifiant utilisé par le nombre maximum de voisins.

Cette méthode est également instable, ne donnant que très rarement la même partition d'un lancement à l'autre. L'instabilité nécessite de lancer plusieurs fois l'algorithme avant d'obtenir une solution correcte en termes de qualité. Cette méthode est sujette aux mauvaises propagations de labels, donnant des communautés géantes.

3.4.2 SLPA

SLPA pour " Speakerlistener Label Propagation Algorithm" (Xie et al. (2011) **[57]** est un algorithme fondé sur une mémoire de labels et sur la popularité des labels les plus influents au sein du graphe, c'est-à-dire, des labels que l'on retrouve le plus souvent pour l'identification des communautés résultantes du graphe. Le changement de label d'un nœud n'est plus fonction du voisinage actuel, mais dépend de la mémoire du nœud dont les labels voisins ont changé au cours du temps.

Initialement, on attribue à chaque nœud une mémoire de labels. Il s'agit d'un vecteur de taille k (si le nœud courant à k voisins) qui est originellement constitué des labels de ses nœuds voisins. Deux types de nœuds sont définis dans l'algorithme SLPA, les speakers et listeners.

Un nœud est alors sélectionné aléatoirement comme speaker et propage son label pendant que les autres observent la propagation, jusqu'à stabilisation. C'est alors qu'un autre nœud du graphe est sélectionné pour jouer le rôle du listener. Une fois le processus de propagation terminé, on peut observer la force d'appartenance qu'a un nœud vis-à-vis des diverses communautés via l'influence des nœuds dans les différentes mémoires. A la fin du processus, on regarde toutes les mémoires de tous les nœuds pour y voir les labels des communautés les plus fréquentes. Pour sélectionner les labels les plus fréquents, on utilise un seuil $r \in [0,1]$ (Post-processing threshold) qui permet de considérer les labels les plus influents (ceux les plus fréquents dans la mémoire) et qui a également une influence forte sur le pourcentage des nœuds chevauchants.

Il peut y à voir des labels ayant le même score d'influence, permettant ainsi le chevauchement. Les résultats en termes de qualité de partitionnement sont encourageants, la majorité des nœuds chevauchants dans les graphes utilisés dans leurs expérimentations sont détectés comme pour le

club de karaté, le réseau footballistique, le réseau de livres scientifiques, le réseau de collaboration scientifique et d'autres graphes sociologiques.

Les structures communautaires peuvent empêcher la propagation du label du speaker de continuer, et pas forcément les nœuds qui peuvent appartenir à plusieurs communautés.

3.5 Tableau récapitulatif de complexité des méthodes chevauchantes

Nous avons vu quelques algorithmes de détection de communautés chevauchantes, un tableau récapitulatif, Tableau 2, permettant d'observer la complexité algorithmique de chaque algorithme.

Algorithme	Complexité	Description	Réf
CPM	O (n^2)	• Solution simple, son algorithme prend un seul paramètre d'entrer K qui présente la taille des cliques. • Complexité et une consommation mémoire très élevé, ainsi le paramètre d'entrer K doit être défini au préalable.	[59]
Extension local et optimisation	O (n^2)	• Basée sur l'optimisation d'une fonction de force (community strength) • La qualité des communautés découvertes dépend de la qualité des graines. Certaines communautés risquant de ne pas être détectées en cas de chevauchement très important.	[55]
COPRA	O(kmn), k :paramètre entier	• Possibilité de fixer le nombre maximum de labels qu'un nœud peut retenir. • Copra est limité par taille des réseaux et produit un nombre des communautés de petite taille dans certains réseaux.	[44]
SLPA	O ($2n + n^2m$)	• Très rapide et peut traiter des réseaux très grands et denses	[57]

		• Problème dans la mémoire qui contient des labels. Comment désigner le bon nœud listener.	
Partitionnement de liens		• Solution simple et facile à implémentée. • Il n'y a aucune garantie qu'il fournit une détection de meilleure qualité que la détection basée sur les nœuds.	[54]

Tableau 2 – Comparaison entre les principaux algorithmes de détection de communautés chevauchantes.

3.6 Conclusion

Les études comparatives menées par les experts du domaine ont montré l'existence de nombreuses méthodes de détection de communautés. Cette section a permis de donner notre formulation du problème de détection de communautés disjointe et chevauchantes, d'analyser les méthodes existantes et de voir la mesure de qualité associé au partitionnement qui est la fonction de modularité sur laquelle s'appuie notre étude. L'état de l'art nous a permis de voir que l'algorithme de Louvain basé sur l'optimisation de la modularité permet de traiter de grands graphes, c'est donc l'option que nous privilégions. Dans la mesure où nous souhaitons créer une méthode de détection de communauté chevauchante et destinée à des graphes à grande échelle nous allons ajuster l'algorithme disjoint de Louvain afin de l'adapter pour détecter des communautés chevauchantes. Le prochain chapitre sera consacré à la définition de notre approche, nous détaillerons l'algorithme de Louvain ainsi son ajustement afin de répondre aux exigences de découverte et trouver à la fois les communautés qui se chevauchent.

Chapitre III

Développement d'une méthode pour la détection des communautés chevauchantes

1 Introduction

L'état de l'art concernant la détection de communautés disjointes et chevauchantes présenté dans le chapitre précèdent a permis de voir l'existence de quelque grandes classes d'algorithmes, à savoir les approches classiques et les approches séparatives, les approches alternatives. Les chercheurs ont été observé que les méthodes où le point de départ est atomique (par le nœud), permettaient de traiter de plus grands graphes **[54]**.

Dans ce chapitre, nous allons exposer notre approche fondée sur l'algorithme de Louvain **[58]** en introduisant une méthode efficace pour l'ajuster afin de répondre aux exigences de découverte et trouver à la fois les communautés qui se chevauchent.

2 Schéma et description générale de notre approche

Notre but est de permettre à chaque nœud du réseau d'appartenir à une ou plusieurs communautés. Notre approche de détection de communautés chevauchantes est composée de deux phases principales :

- Phase 1 : appliquer l'algorithme de Louvain pour générer des communautés disjointes.
- Phase 2 : ajuster les nœuds situés à la frontière entre des communautés disjointes de manière à les affecter dans un ensemble de bonnes communautés qui se chevauchent.

L'algorithme de Louvain est le plus rapide pour optimiser la modularité avec une complexité constatée sur des instances de graphes peu denses en $O(m)$, Bien que la complexité de calcul exacte de l'algorithme est difficile à déterminée, il semble fonctionner dans le temps $O(n\ log\ n)$ selon les tests, avec la plus grande partie de l'effort de calcul consacré à l'optimisation au premier niveau. L'optimisation de la modularité exacte est connue pour être NP_Difficile. Il autorise le traitement de très grands graphes et il fournit aussi un compromis très intéressant entre performance et vitesse d'optimisation pour une utilisation dans les algorithmes d'optimisation.

Il convient également de noter que la sortie de l'algorithme dépend de l'ordre dans lequel les nœuds sont considérés. L'algorithme est instable, ne produisant jamais le même résultat d'un lancement à l'autre. L'instabilité dans l'algorithme de Louvain est dû aux choix aléatoire des nœuds dans le processus, ce qui donne des résultats différents à chaque exécution de l'algorithme. L'ordre joue un rôle important sur la qualité des communautés détectées. C'est dans ce sens que notre première

contribution après la détection des communautés disjointes est d'assurer la stabilité de l'algorithme en évitant le choix aléatoire et exécuter l'algorithme dans un seul ordre. La seconde contribution sera d'ajuster l'algorithme pour le chevauchement en y incluant une fonction permettant de détecter des nœuds pouvant appartenir à plusieurs communautés.

Un problème très intraitable en matière de détection communautaire est que certains nœuds situés aux limites entre les communautés (Border nodes) sont souvent difficiles à classer dans une communauté ou une autre. En général, ces nœuds sont considérés comme les nœuds instables. La dixième phase est basée sur une définition locale de la force de la communauté (Community strength) avec un paramètre ajustable α, où un petit α correspond à une grande étendue des nœuds qui se chevauchent.

Avec la définition locale de la force de la communauté, les nœuds d'un réseau (utilisés principalement pour les nœuds situés à la frontière entre des communautés disjointes) peuvent être ajustés de manière à les affecter dans un ensemble de bonnes communautés qui se chevauchent, en s'appuyant simplement sur les informations locales impliquant le nombre de liens à l'intérieur et à l'extérieur du groupe de nœuds. Notre approche est composée de deux phases comme illustre la figure suivante.

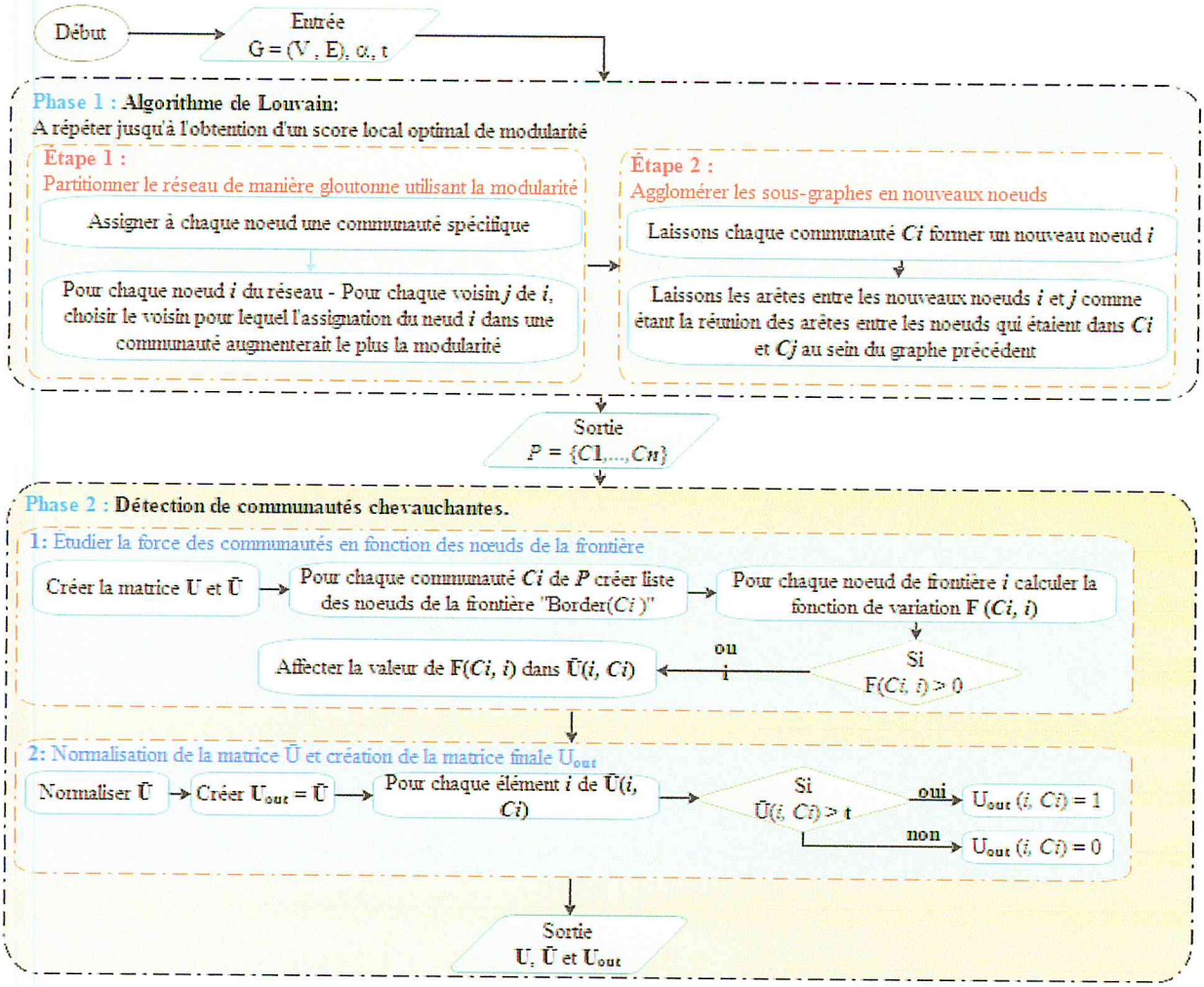


Figure 3.1 - Organigramme de notre approche de détection de communautés chevauchantes (DCC).

3 Détail de la phase 1 (Algorithme de Louvain)

Cette phase consiste à détecter une structure de communautés disjointe à l'aide de l'algorithme de Louvain.

Cet algorithme a d'abord été introduit pour trouver des partitions de haute modularité Newman-Girvan, ce qui est le critère le plus largement utilisé pour évaluer une partition d'un réseau complexe. L'algorithme utilise une optimisation gloutonne locale (Greedy optimization) pour trouver un maximum de modularité locale, qui est encore amélioré par un raffinement hiérarchique.

L'algorithme de Louvain est divisé en deux étapes répétées de manière itérative. Supposons que nous commencions avec un réseau de N nœuds. Premièrement, nous assignons une communauté différente à chaque nœud du réseau. Donc, dans cette partition initiale, il y a autant de communautés qu'il y a de nœuds.

Ensuite, pour chaque nœud i, nous considérons les voisins j de i et nous évaluons le gain de modularité (ΔQ) en retirant i de sa communauté et en le plaçant dans la communauté de j. Le nœud i est alors placé dans la communauté pour laquelle ce gain est maximum, Si aucun gain positif n'est possible, i reste dans sa communauté originale.

Ce processus est appliqué de manière répétée et séquentielle pour tous les nœuds jusqu'à ce qu'aucune amélioration ne puisse être réalisée, c'est-à-dire qu'aucun mouvement individuel ne peut améliorer la modularité. Cette première étape est alors terminée et un maximum local de la modularité est atteint.

Une partie de l'efficacité de l'algorithme résulte du fait que le gain de modularité (ΔQ) obtenu en déplaçant un nœud isolé i dans une communauté C, (ΔQ) peut facilement être calculé par:

$$\Delta Q = \left[\frac{\Sigma_{in} + K_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + K_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right] \qquad (9)$$

Où Σ_{in} est la somme des poids des liens dans C, Σ_{tot} est la somme des poids des liens incidents aux nœuds dans C, k_i est la somme des poids des liens incidents au nœud i, $k_{i,in}$, est la somme des poids des liens de nœud i aux nœuds de C et m est la somme des poids de tous les liens du réseau.

Une expression similaire est utilisée afin d'évaluer le changement de modularité lorsque i est retiré de sa communauté.

La deuxième étape de l'algorithme consiste à construire un nouveau réseau dont les nœuds sont maintenant les communautés trouvées lors de la première étape. Pour ce faire, les poids des liens entre les nouveaux nœuds sont donnés par la somme du poids des liens entre nœuds dans les deux communautés correspondantes.

Les liens entre les nœuds de la même communauté conduisent à des boucles pondérées pour cette communauté dans le nouveau réseau. Une fois cette deuxième étape terminée, il est alors possible de réappliquer la première étape de l'algorithme sur le réseau pondéré résultant et de l'itérer. La

figure suivante montre les étapes de l'algorithme de Louvain, Notons par "passage" la combinaison de ces deux étapes. Par construction, le nombre de communautés diminue à chaque passage, et par conséquent la plupart du temps de calcul est utilisé dans le premier passage. Les passages sont itérés jusqu'à ce qu'il n'y ait plus de changements et qu'un maximum de modularité soit atteint comme illustre la figure suivante.

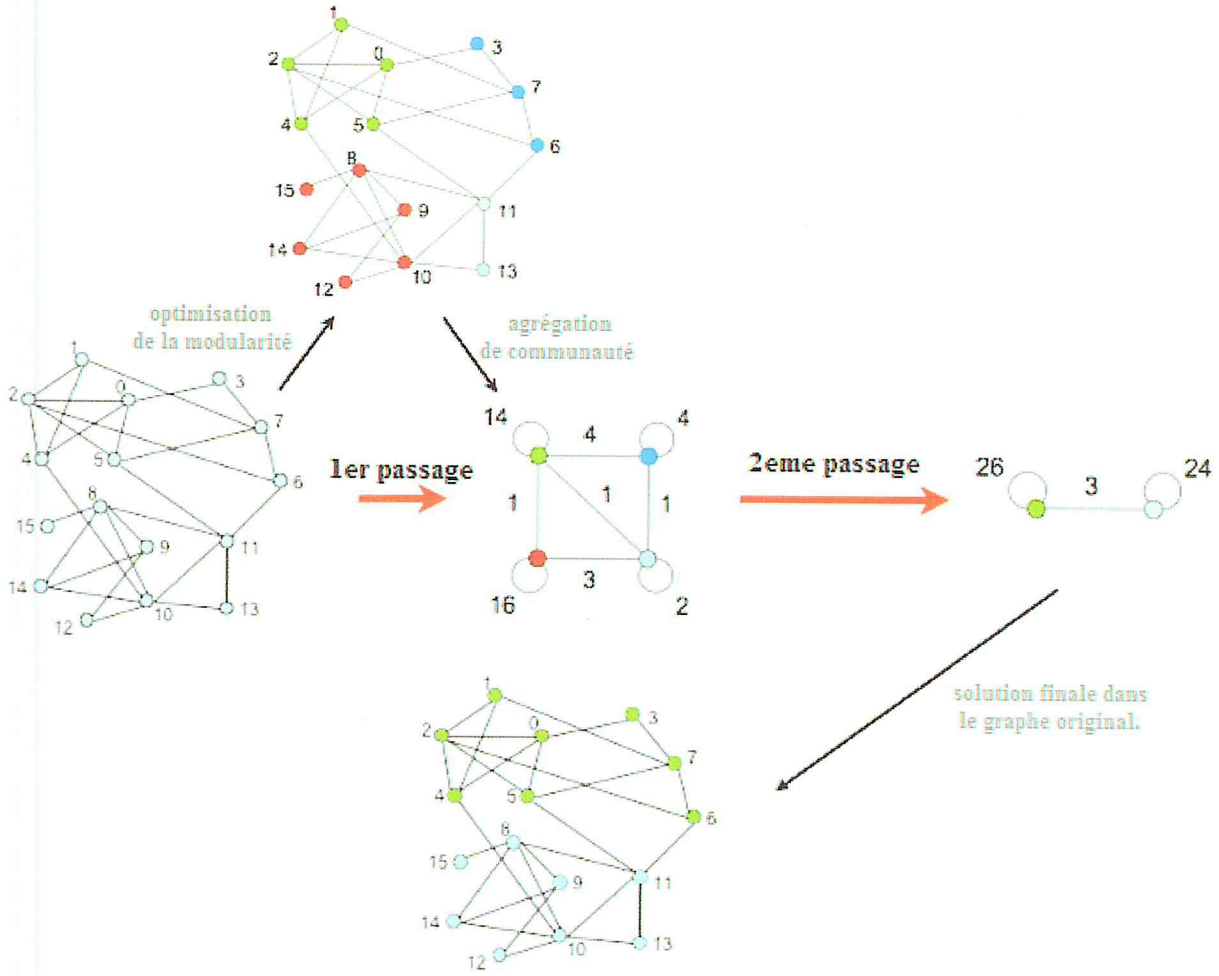


Figure 3.2 - Visualisation des étapes de l'algorithme de Louvain.

Chaque passage est constitué de deux étapes, une où la modularité est optimisée en ne permettant que des changements locaux de communautés, une où les communautés trouvées sont agrégées afin de construire un nouveau réseau de communautés. Les passages sont répétés itérativement jusqu'à ce qu'aucune augmentation de la modularité ne soit possible.

L'algorithme de Louvain dans la première étape utilise l'heuristique VM **[68]** (vertex mover). À partir d'un graphe d'origine à partitionner $G^0 = G$, l'application de VM au niveau 0 donne une partition P^0. Le graphe de niveau 1, noté G^1 est engendré à partir de P^0selon la procédure d'agrégation de communauté dans la deuxième étape. Au niveau 1, l'heuristique VM est appliquée à G^1 et cette procédure récursive se poursuit jusqu'au dernier niveau l où VM ne déplace aucun noeud, c'est-à-dire lorsque $|P^l| = |G^l|$, un autre exemple dans la figure suivante explique passage d'un niveau à l'autre.

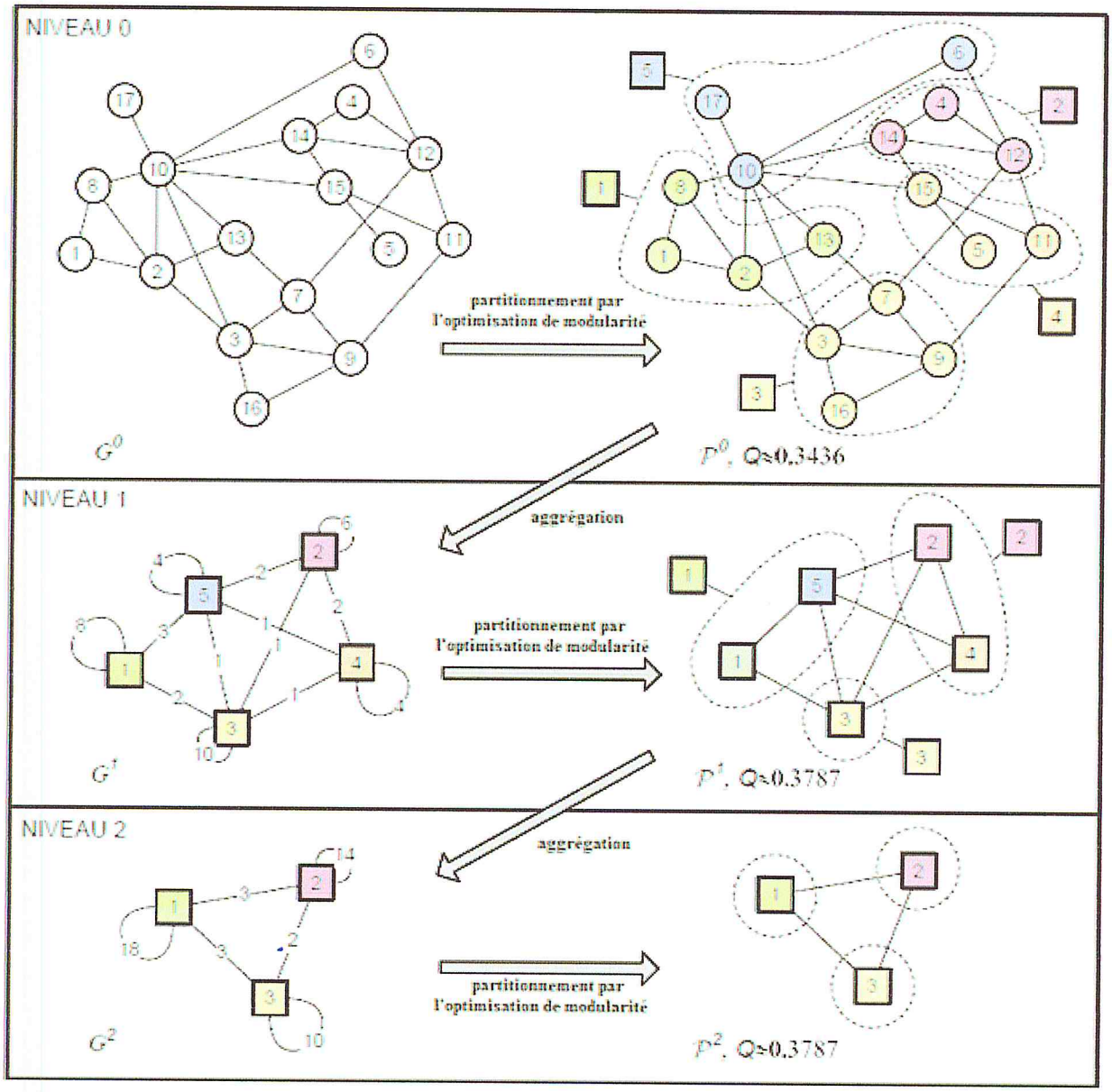


Figure 3.3 - Illustration de l'algorithme de Louvain.

Au niveau 0, le partitionnement par l'heuristique VM du graphe G^0donne un découpage en cinq communautés (partition P^0). Dans le graphe du niveau suivant après l'agrégation, il y a cinq nœuds (un par communauté du niveau précédent) et les arêtes sont pondérées de façon à ce que la modularité soit la même, si l'on prend des communautés singletons (un nœud dans chacune) dans le nouveau graphe. Le processus se poursuit avec ce graphe jusqu'au niveau 2 où la modularité Q de la partition singleton de départ ne peut pas être améliorée.

L'algorithme de Louvain présente dans les étapes 1 et 2 plusieurs procédures qui doivent être précisément définis et sont fortement liés à la fonction de modularité de Girvan-Newman, ces procédures sont répétées de façon itérative jusqu'à ce qu'un maximum de modularité soit atteint et qu'une hiérarchie de communautés soit produite.

Algorithme 1 L'algorithme de Louvain

Entrée : Un graphe G = (V, E)

1: **A répéter jusqu'à l'obtention d'un score local optimal**
2: **Étape 1:** partitionner le réseau de manière gloutonne utilisant la modularité
3: **1)** Assigner à chaque nœud une communauté spécifique
4: **2)** Pour chaque nœud i du réseau
— Pour chaque voisin j de i, choisir le voisin pour lequel l'assignation du nœud i dans une communauté augmenterait le plus la modularité
— Répéter le processus jusqu'à ce qu'il n'y ait plus de changement
5: **Étape 2 :** Agglomérer les sous-graphes en nouveaux nœuds
6: **1)** Laissons chaque communauté C_i former un nouveau nœud i
7: **2)** Laissons les arêtes entre les nouveaux nœuds i et j comme étant la réunion des arêtes
entre les nœuds qui étaient dans C_i et C_j au sein du graphe précédent

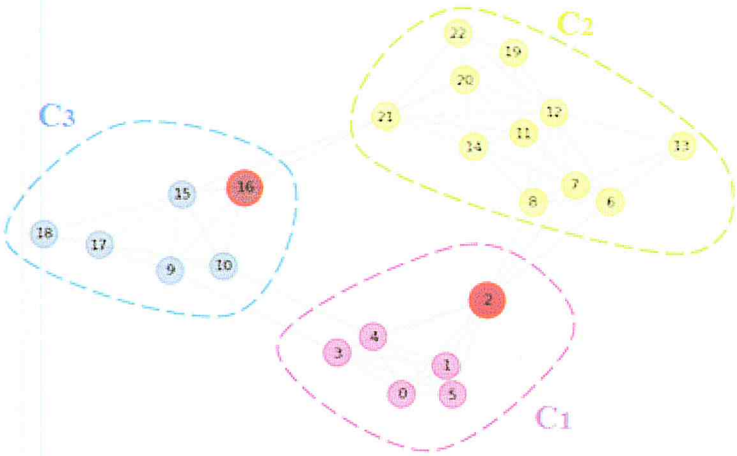
Algorithme 1 – Algorithme de Louvain.

4 Détail de la phase 2

On nommera notre approche : DCC (détection de communautés chevauchantes).

Après avoir détecté les communautés disjointes dans la précédente phase. Cette phase a pour but la détection des communautés chevauchantes en se basant sur le résultat de la phase précédente.

En ce qui concerne les communautés qui se chevauchent et les nœuds instables, on peut voir **figure 3.4** (le graphe de gauche), qui montre un exemple, où les nœuds 2 et 16 apparaissent comme des nœuds pouvant appartenir à deux communautés.



	U_{is}	$\bar{U}$	U_{out}
0	[1 0 0]	[1 0 0]	[1 0 0]
1	[1 0 0]	[1 0 0]	[1 0 0]
2	[1 0 0]	[0.9815 0.185 0]	[1 1 0]
3	[1 0 0]	[1 0 0]	[1 0 0]
4	[1 0 0]	[1 0 0]	[1 0 0]
5	[1 0 0]	[1 0 0]	[1 0 0]
6	[0 1 0]	[0 1 0]	[0 1 0]
7	[0 1 0]	[0 1 0]	[0 1 0]
8	[0 1 0]	[0 1 0]	[0 1 0]
9	[0 0 1]	[0 0 1]	[0 0 1]
10	[0 0 1]	[0 0 1]	[0 0 1]
11	[0 1 0]	[0 1 0]	[0 1 0]
12	[0 1 0]	[0 1 0]	[0 1 0]
13	[0 1 0]	[0 1 0]	[0 1 0]
14	[0 1 0]	[0 1 0]	[0 1 0]
15	[0 0 1]	[0 0 1]	[0 0 1]
16	[0 0 1]	[0 0018 0.9982]	[0 1 1]
17	[0 0 1]	[0 0 1]	[0 0 1]
18	[0 0 1]	[0 0 1]	[0 0 1]
19	[0 1 0]	[0 1 0]	[0 1 0]
20	[0 1 0]	[0 1 0]	[0 1 0]
21	[0 1 0]	[0 1 0]	[0 1 0]
22	[0 1 0]	[0 1 0]	[0 1 0]

Figure 3.4 - Un exemple de communautés chevauchantes identifiées par notre approche (DCC).

Dans la partie gauche, les communautés sont trouvées en utilisant l'algorithme de Louvain, Les nœuds chevauchants détectés sont indiqués par la couleur rouge. Dans la partie droite nous montrons (hard partition, soft partition et final clusters) de chaque nœud.

La deuxième phase est basée sur une définition locale de la force de la communauté (community strength) définie comme le ratio entre les liens internes et la somme des degrés des nœuds de la communauté :

$$strength(C) = \frac{c^{int}}{(c^{int}+c^{ext})^{\alpha}} \qquad (11)$$

Où C^{int} et C^{ext} sont le degré total interne et externe de la communauté, α est un paramètre appelé paramètre de résolution qui permet de faire varier la taille des communautés trouvées, Une valeur de α inférieure à 0,5 conduira à un nombre important de nœuds chevauchants, tandis que des valeurs de α supérieures à 2 conduisent à un petit nombre de nœuds chevauchants. La valeur de α est relative à la valeur de modularité, de sorte que la modularité augmente si α augmente aussi, et donc le nombre de nœuds chevauchés diminue.

Avec la définition locale de la force de communauté, les nœuds d'un réseau (utilisés principalement par les nœuds situés à la frontière entre communautés disjointes) peuvent être ajustés de manière à les affecter dans un ensemble de communautés qui se chevauchent, en se basant simplement sur les informations locales. Le nombre de liens à l'intérieur et à l'extérieur du groupe de nœuds. D'un autre côté, nous aimerions encore quantifier l'influence d'un nœud i donné par rapport à la communauté C. Pour ce faire, nous allons évaluer la variation de la force d'une communauté C avec et sans nœud i **[41, 62]**. Cette variation est définie par l'équation (12) :

$$F(C,i) = strength(C \cup \{i\}) - strength(C \backslash \{i\}) \qquad (12)$$

Où les symboles $C \cup \{i\}$ et $C \backslash \{i\}$ désignent les sous-graphes obtenus à partir de la communauté C avec le noeud i à l'intérieur et à l'extérieur respectivement. La méthode peut également analyser le problème des nœuds instables dans la détection des communautés. Enfin, la performance de notre méthode est testée sur un ensemble de réseaux générés par ordinateur et les réseaux du monde réel.

4.1 Préliminaires

Formellement, étant donné un réseau $G = (V, E)$ avec $N = |V|$ nœuds et $m = |E|$ arêtes, l'objectif de l'algorithme de Louvain pour la détection de communauté est de trouver une partition P=$\{C_1, C_2, \ldots\ldots, C_c\}$ de nœuds ($Ci \cap Cj = \emptyset$, $i \neq j$ et $UiCi = V$).

Une représentation pratique de la partition correspondante est la matrice de partition $U = \{U(i, k)\}$ **[64]**. La matrice de partition U a N lignes et c colonnes où c est le nombre de communautés, et $U(i,k) = 1$ si et seulement si le nœud i appartient au k ième sous-ensemble de la partition; sinon c'est zéro. Il s'ensuit clairement que $\sum_{k=1}^{c} U(i,k) = 1$ pour chaque i=1,..., N. La taille de la communauté k peut alors être calculée comme $\sum_{i=1}^{N} U(i,k)$ et pour toute partition significative, on peut supposer que $0 < \sum_{i=1}^{N} U(i,k) < N$

Nous utilisons le symbole $\boldsymbol{U_{in}}$ pour désigner la matrice de partition (hard partition) obtenue avec l'algorithme de Louvain.

La matrice de partition (Soft partition) $\widehat{U} = \{\widehat{U}(i,k)\}$: on a obtenu cette matrice afin de permettre à l'élément $\widehat{U}(i,k)$ d'atteindre toute valeur réelle de l'intervalle [0, 1], et les contraintes correspondantes imposées à la matrice de partition restent les mêmes :

$$\widehat{U}(i,k) \in [0,1], pour\ 1 \leq k \leq c, 1 \leq i \leq N$$

$$\sum_{k=1}^{c} \widehat{U}(i,k) = 1, pour\ 1 \leq i \leq N \qquad \textbf{(1b)}$$

$$0 < \sum_{i=1}^{N} U(i,k) < N\ pour\ 1 \leq k \leq c.$$

Nous introduisons un seuil t pour convertir la matrice de partition $\widehat{U}$ en une matrice de partition finale $\boldsymbol{U_{out}}$. $U_{\text{out}}(i,k) = 1$ si et seulement si $\widehat{U}(i,k) > t$. La Matrice finale U_{out} est juste le résultat final que nous recherchons.

4.2 Modularité

L'algorithme de Louvain utilisé pour la détection des communautés disjointes s'articule autour de la maximisation de la modularité Notre méthode est de trouver des communautés qui se chevauchent. Pour ce faire, la définition de modularité traditionnelle doit être ajustée aux partitions qui se chevauchent. Shen et al. **[63]** ont introduit une extension de la modularité adaptée pour trouver à la fois les communautés chevauchantes et la structure hiérarchique dans les réseaux complexes. La modularité de Shen est définie par l'équation:

$$Q_{ol} = \frac{1}{2m} \sum_{k=1}^{c} \sum_{i,j \in C_k} \frac{1}{O_i O_j} \left(A_{ij} - \frac{K_i K_j}{2m}\right) \qquad (13)$$

Où $Q_i$et Q_j et sont les nombres de communautés auxquelles appartiennent respectivement les nœuds i et j. Dans le cas de communautés disjointes, c'est-à-dire que chaque nœud appartient à une seule communauté, la modularité $\boldsymbol{Q_{ol}}$ deviendra la modularité de Newman (2).

4.3 L'algorithme de la phase 2

Initialement, nous appliquons l'algorithme de Louvain pour obtenir une partition de départ $P = \{C_1, C_2, \ldots\ldots, C_c\}$ de communautés disjointes, ce qui est indiqué par la matrice de partition

U_{in} . Ensuite, nous mettons à jour U_{in} en ajoutant ou en supprimant un nœud à la fois, à condition que la force de la communauté s'améliore. En général, les seuls nœuds capables d'améliorer la force de la communauté sont les nœuds frontaliers de la communauté (Border Nodes). Pour une communauté donnée C_1 les nœuds frontaliers (Border Nodes) de cette communauté sont les nœuds adjacents aux nœuds de la communauté voisine C_2, en incluant aussi les nœuds de C_2 adjacent aux nœuds bordure de C_1.

Ainsi, plutôt que d'explorer tous les nœuds du réseau dans le processus de mise à jour, nous pouvons sauter tous les nœuds à l'exception de ceux qui sont les nœuds de bordure de cette communauté, ce qui pourrait améliorer significativement l'efficacité de l'algorithme.

Formellement, les nœuds frontaliers de la communauté C sont définis comme :

$$Border\ (C) = \{\{u\},\{v\}\,|\{u,v\}\ \in E, u\ \in C\ , v \notin C \qquad (14)$$

Où u et v sont deux nœuds adjacents, avec u appartenant à la communauté C et v appartenant à la communauté voisine . Si $F(C_k, i) > 0$, c'est-à-dire l'influence du nœud i par rapport à la communauté C_k est positive, quand le nœud i est à l'intérieur de la communauté C_k le retirer de la communauté ou bien, ajoutez-le à la communauté. La variation $F(C_k, i)$ peut être désigné par l'élément $\widehat{U}(i,k)$ de la matrice $\widehat{U}$.

En termes **Eq(1b)**, nous nous attendons à ce que le degré d'appartenance total pour chaque nœud soit égal à 1, et distribué parmi les communautés. Pour y parvenir, nous normalisons simplement la matrice actuelle $\widehat{U}$ pour avoir des valeurs entre 0 et 1

$$Normalize(\widehat{U}) = \{\widehat{U}(i,k)|\widehat{U}(i,k) = \frac{\widehat{U}(i,k)}{\sum_q \widehat{U}(i,k)}\ , q = 1\ldots.c\} \qquad (15)$$

La matrice $\widehat{U}$ peut être considérée comme une matrice de partition (Soft-partition) avec des communautés qui se chevauchent. Enfin, un seuil t est introduit (dans de nombreux cas, on peut simplement définir le seuil t = 0) et puis nous pouvons convertir la matrice de partition (Soft-partition) en la matrice de partition finale $\boldsymbol{U_{out}}$. L'appartenance de chaque communauté est alors $C_k = \{\, i|\ \widehat{U}(i,k) > t\ , i \in V\ \}$, ce que représente le résultat final.

Algorithme 2 L'algorithme de notre méthode

Entrée : Un graphe G = (V, E), α, t

1. P = Louvain (G); // $P = \{C_1, C_2, \ldots . C_c\}$
2. $U_{in} = P$, // construction de la matrice (Hard partition)
3. $\widehat{U} = U_{in}$;
4. **pour tout** $C_k \in P$ // $k = 1 \ldots . c$, **faire**
5. $B_k = border(C_k)$;
6. **pour tout** $i \in B_k$
7. **Si** $i \in C_k$
8. $C'_k \leftarrow C_k \backslash \{i\}$;
9. **Sinon** $C'_k \leftarrow C_k \cup \{i\}$;
10. **Fin si**
11. **Si** $F(C_k, i) > 0$
12. $\widehat{U}(i, k) = F(C_k, i)$;
13. $C_k \leftarrow C'_k$;
14. **Fin si**
15. **Fin pour**
16. **Fin pour**
17. $\widehat{U} \leftarrow$ normalize $(\widehat{U})$;
18. $U_{out} = (\widehat{U} > t)$;
19. **Retourner** U_{in}, $\widehat{U}$, U_{out} ;

Algorithme 2 – Algorithme de la phase 2 de l'approche (DCC).

Comme mentionné précédemment, un paramètre optimal α est très utile pour la méthode. Ici, nous pouvons choisir une meilleure valeur de α en maximisant la modularité de la partition chevauchante. Une illustration de cette procédure sur le réseau de la **figure 3.4** est montrée par la **figure 3.5**. La modularité maximale peut être obtenu pour $\alpha > 0.7$.

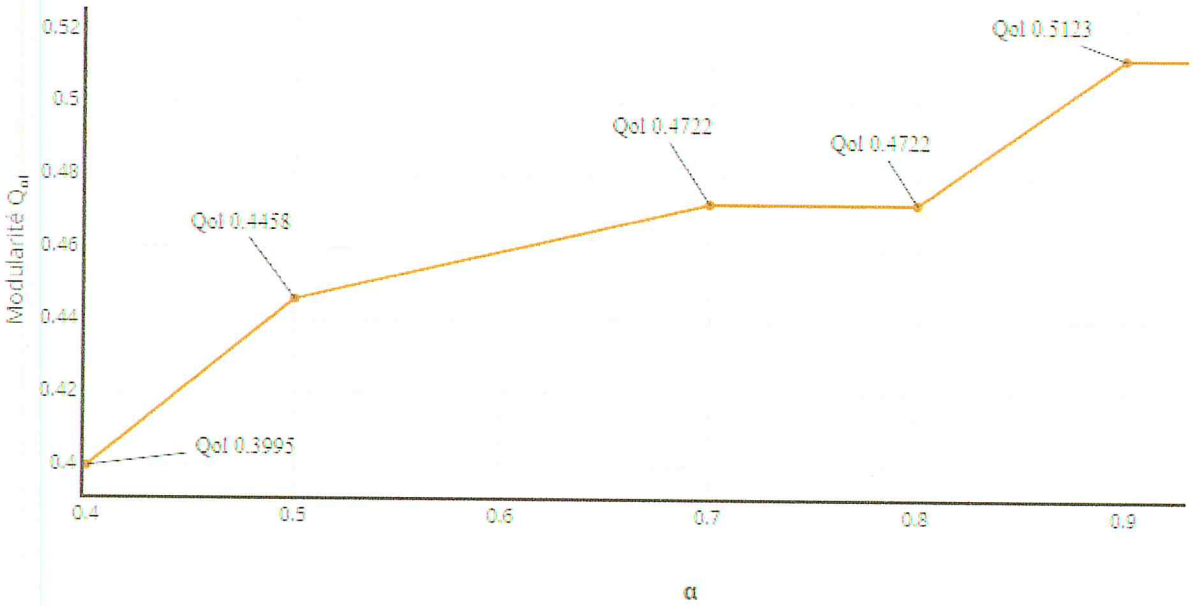


Figure 3.5 - Modularité par rapport au paramètre α pour l'exemple du réseau (voir Figure 3.4).

Ensuite, nous analysons la complexité temporelle de l'algorithme de la deuxième phase. Étant donné un réseau $G = (V, E)$ avec $N = |V|$ nœuds et $m = |E|$ arrêtes, supposons que la matrice de partition U_{in} et le paramètre α sont également donnés. Soit c le nombre de communautés, et m_k^{in} et m_k^{out} les nombres de liens internes et externes pour la communauté C_k, respectivement.

Avec la définition de la communauté (c'est-à-dire une communauté est un groupe de nœuds dans lequel le nombre de liens internes au sein du groupe est plus grand que celui des liens externes), alors :

$$\sum_{k=1}^{c} m_k^{in} > \sum_{k=1}^{c} m_k^{out}, \qquad \sum_{k=1}^{c} m_k^{in} + \frac{1}{2}\sum_{k=1}^{c} m_k^{out} = m \qquad (16)$$

Soit $|B_k|$ le nombre de nœuds frontaliers (Border nodes) de la communauté C_k, alors :

$$\sum_{k=1}^{c} |B_k| = \sum_{k=1}^{c} m_k^{out} \qquad (17)$$

Avec Équations. **(16)** et **(17)**, le nombre total de nœuds frontaliers est :

$$\sum_{k=1}^{c} |B_k| < m \qquad (18)$$

Au cours de l'algorithme de la deuxième phase, les opérations principales sont concentrées sur les nœuds frontaliers de chaque communauté du réseau. Dans ce cas, la complexité totale de l'algorithme peut être estimée à $O(m)$. Sur des réseaux peu denses l'algorithme prend une durée d'exécution presque linéaire, la complexité de calcul exacte de l'algorithme est difficile à déterminée, puisque nœuds frontaliers dépendent du nombre de liens dans le réseau initial.

La complexité globale estimée de notre approche est égale à la somme de complexité des deux phases. La complexité est donc :

$$\boldsymbol{Complexité\ globale} = \boldsymbol{O(n\ log\ n)} + \boldsymbol{O(m)} = \boldsymbol{O(m + n\ log\ n)}.$$

5 Résultats expérimentaux

Dans cette section, nous validons notre approche en l'appliquant à des réseaux générés par ordinateur (LFR benchmark) **[69]** et à des réseaux réels dans des domaines sociaux qui sont un ensemble de réseaux couramment rencontrés dans la détection de communautés.

5.1 Réseaux du monde réel

Nous présenterons quelques exemples frappants des réseaux réels possédants une structure de communauté. De cette façon, nous verrons à quoi ressemblent les communautés et pourquoi elles sont importantes. Les réseaux sociaux sont des exemples paradigmatiques de graphes avec les communautés. Les gens ont naturellement tendance à former des groupes, dans leur environnement de travail, leur famille, leurs amis. Nous validons notre méthode en l'appliquant à ces réseaux réels. Le premier est le célèbre réseau de club de karaté de Zachary (Zachary's karate club) **[70]**, le deuxième un réseau de la ligue américaine de football collégial (US college football league) **[71]**, le troisième réseau est un réseau social animal des dauphins (bottlenose dolphin) **[72]**, le dernier réseau est le réseau de musiciens de jazz (Jazz Musicians Network) **[73]**.

5.1.1 Réseau de club de karaté

Premièrement, nous avons étudié un célèbre réseau social d'un club de karaté analysé par Zachary, qui est largement utilisé comme test de référence pour la détection de communautés dans des

réseaux complexes. Le réseau se compose de 34 membres d'un club de karaté en tant que nœuds et de 78 liens représentant l'amitié entre les membres du club qui a été observée sur une période de trois ans. Un désaccord entre l'administrateur du club et l'instructeur du club a mené à la division du club en deux groupes distincts, soutenant l'instructeur et l'administrateur du club, la figure suivante montre comment les membres sont groupés.

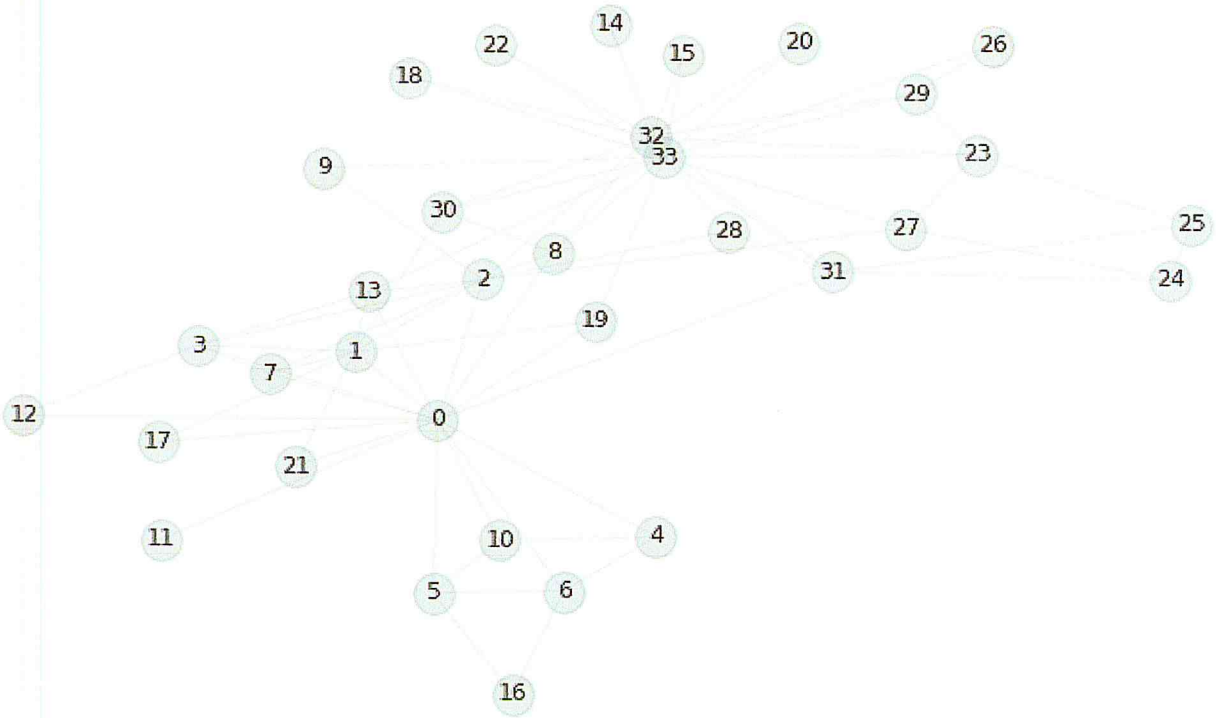


Figure 3.6 - Structure du réseau de club de karaté.

En effet, en regardant la figure, on peut distinguer deux agrégations, un autour le nœud 33 qui est le président, l'autre autour du nœud 0 qui est l'instructeur.

Dans notre test, la méthode de Louvain est appliquée pour générer une partition des communautés disjointes. La question qui nous intéresse est celle de savoir si nous pouvons détecter les communautés chevauchantes du réseau avec notre approche, et identifier simultanément les nœuds chevauchants correspondants avec précision. La Figure 3.7 montre le résultat obtenu.

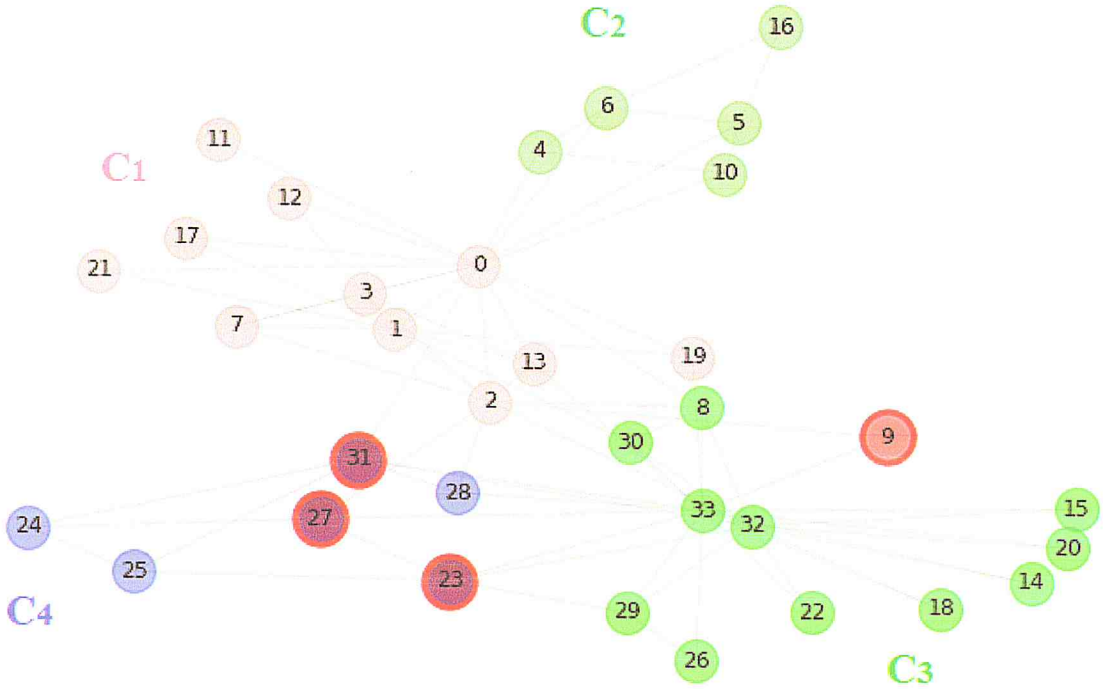


Figure 3.7 - Structure de communautés chevauchantes dans le réseau de club de karaté avec DCC.

Les nœuds chevauchés correspondants sont notés par la couleur rouge. L'approche proposée a détecté quatre communautés avec quatre nœuds chevauchants, avec $\alpha = 1$, la modularité correspondante $Q_{ol} = 0.3882$. Les nœuds chevauchants sont 9, 23,27 et 31. Le résultat dans la matrice finale U_{out} pour ces nœuds est le suivant, 9: [1 0 1 0], 23: [0 0 1 1], 27: [0 0 1 1], 31 [0 0 1 1].

Dans le diagramme suivant nous pouvons voir la variation de la modularité par rapport au paramètre α.

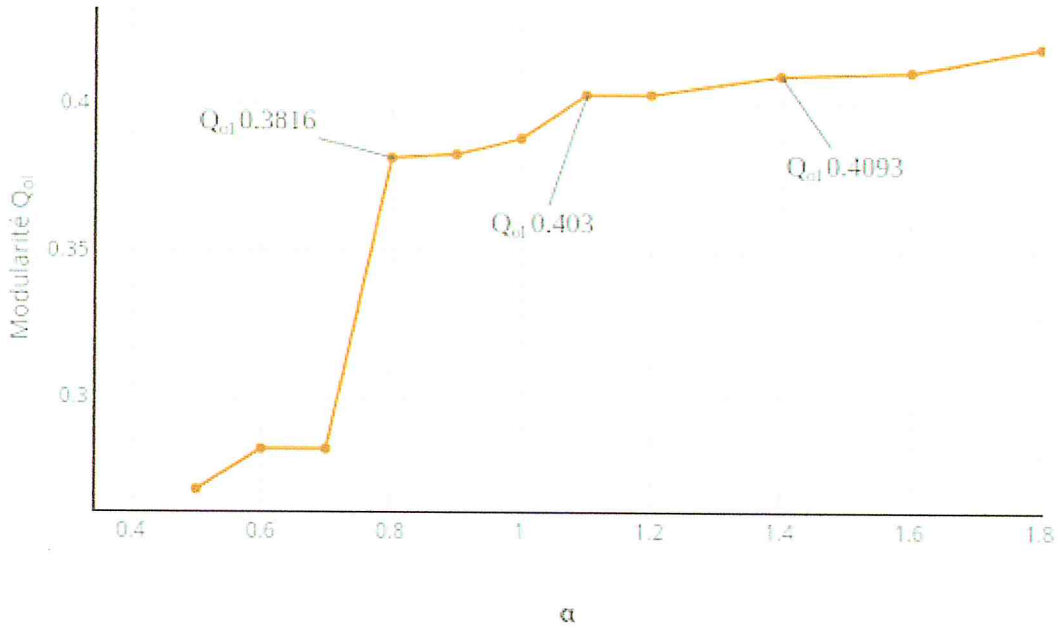


Figure 3.8 - La modularité par rapport au paramètre α pour le réseau de club de karaté. La modularité maximale peut être obtenu pour $\alpha > 0.8$.

5.1.2 Réseau de la ligue américaine de football collégial :

Le deuxième réseau que nous avons étudié est un réseau de football collégial, les nœuds du réseau représentent les 115 équipes alors que les liens représentent 613 parties jouées pendant la saison de football en l'an 2000. Les équipes sont divisées en conférences de 8 à 12 équipes formant des communautés réelles. Les équipes de la même conférence jouent plus souvent que les équipes qui ne participent pas à la même conférence. La figure suivante montre les vraies communautés de ce réseau.

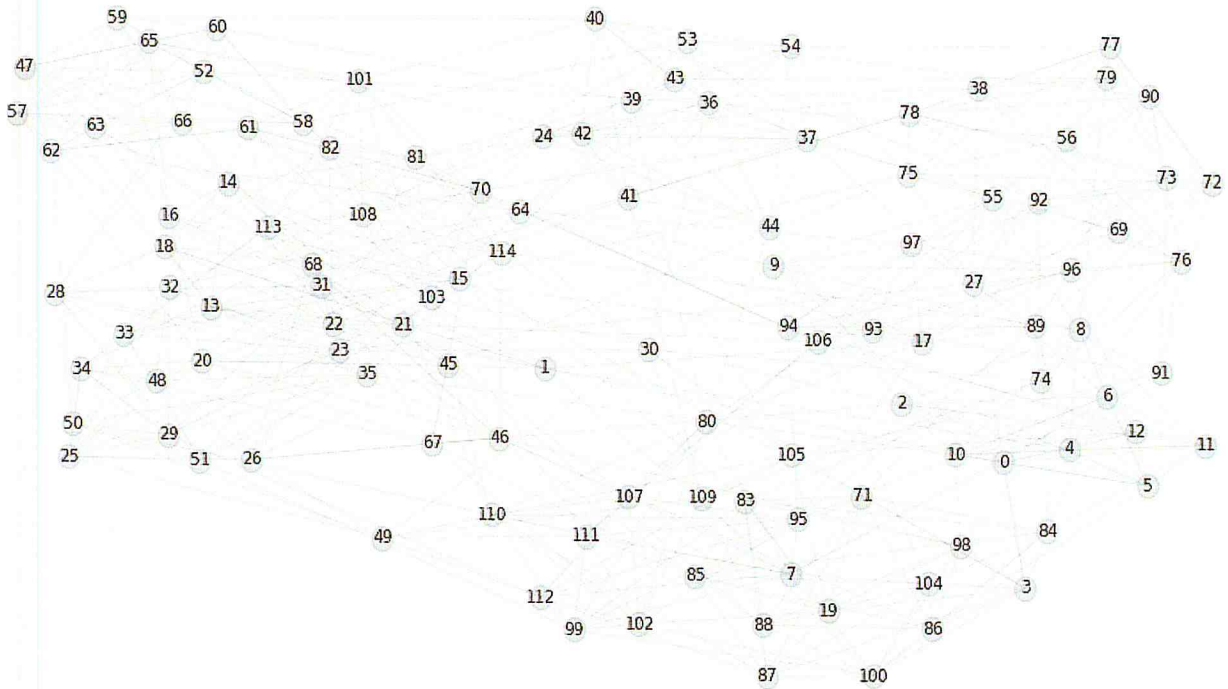


Figure 3.9 - Structure du réseau de football collégial.

La figure suivante montre le résultat final pour ce réseau obtenu avec notre approche DCC. Ce résultat est dans le cas d'une partition à huit communautés obtenue par la méthode Louvain de ce réseau, avec $\alpha = 1.2$, la modularité correspondante $Q_{ol} = 0.5612$. Comme nous pouvons le voir, la méthode peut identifier huit communautés. Plus important encore, dans le cas de $\alpha = 1.2$, l'algorithme peut identifier 9 nœuds comme des nœuds chevauchés qui appartiennent à au moins deux communautés en même temps.

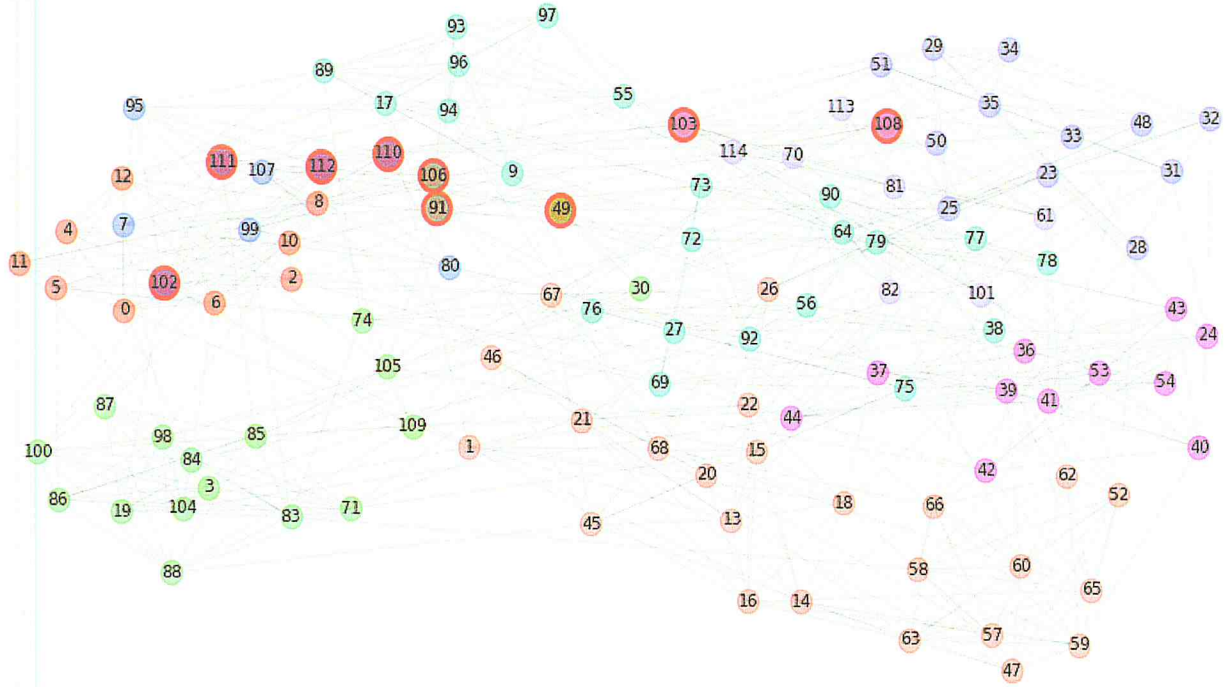


Figure 3.10 - Structure de communautés chevauchantes dans le réseau de football collégial avec DCC.

Les nœuds chevauchés correspondants sont notés par la couleur rouge. Dans le diagramme suivant nous pouvons voir le changement de la modularité par rapport à α.

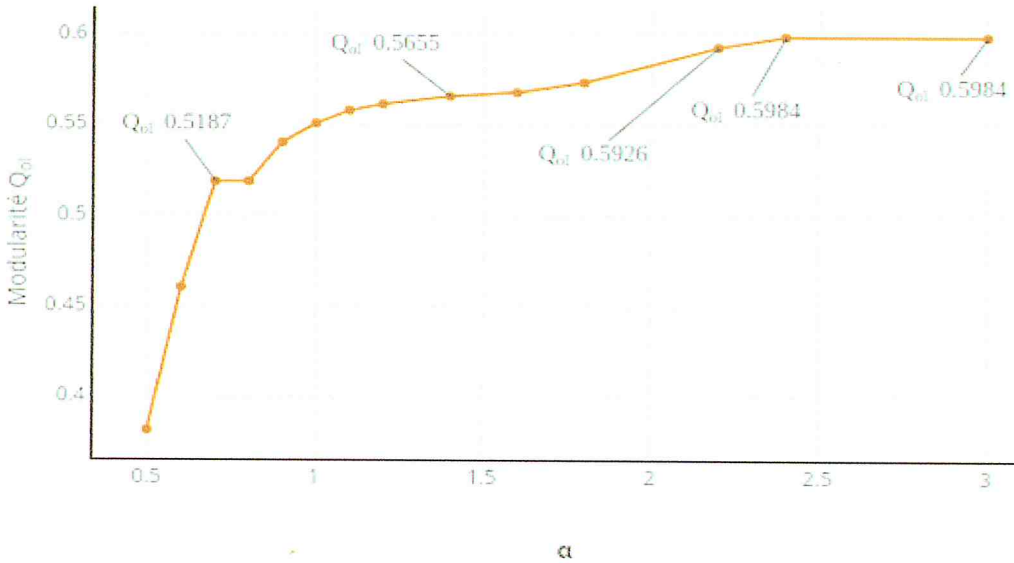


Figure 3.11 - La modularité par rapport au paramètre α pour le réseau de football collégial. La modularité maximale peut être obtenu pour $\alpha > 1$.

5.1.3 Réseau des dauphins :

Le troisième exemple que nous discutons est le réseau social des dauphins, représentant les interactions sociales des grands dauphins vivant à (Doubtful Sound), en Nouvelle-Zélande. Le réseau a été étudié par le biologiste David Lusseau **[74]**, qui a divisé les dauphins en deux groupes en fonction de leur âge. Ce réseau a été construit à partir des observations d'une communauté de 62 grands dauphins sur une période de sept ans, de 1994 à 2001. La figure suivante montre les deux groupes de ce réseau.

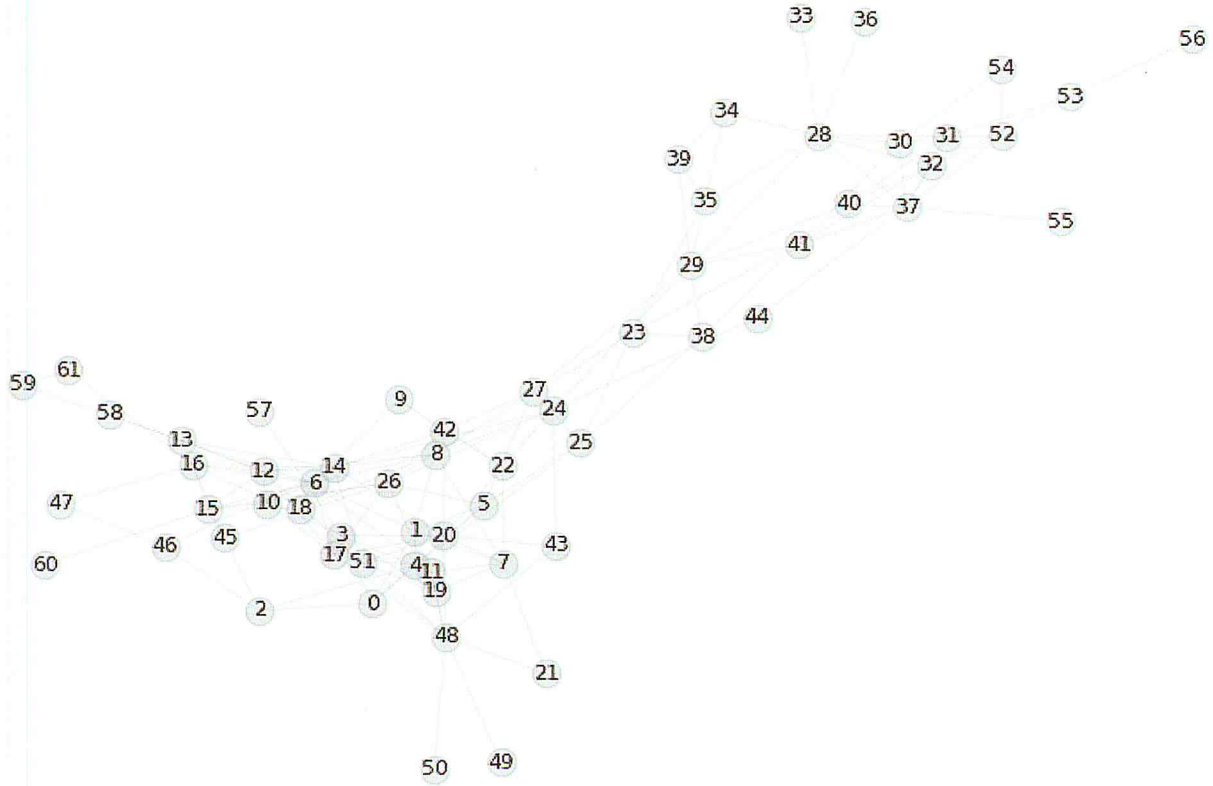


Figure 3.12 - Structure du réseau des dauphins.

La figure suivante montre les nœuds chevauchés pour le réseau social dauphin obtenu avec notre méthode.

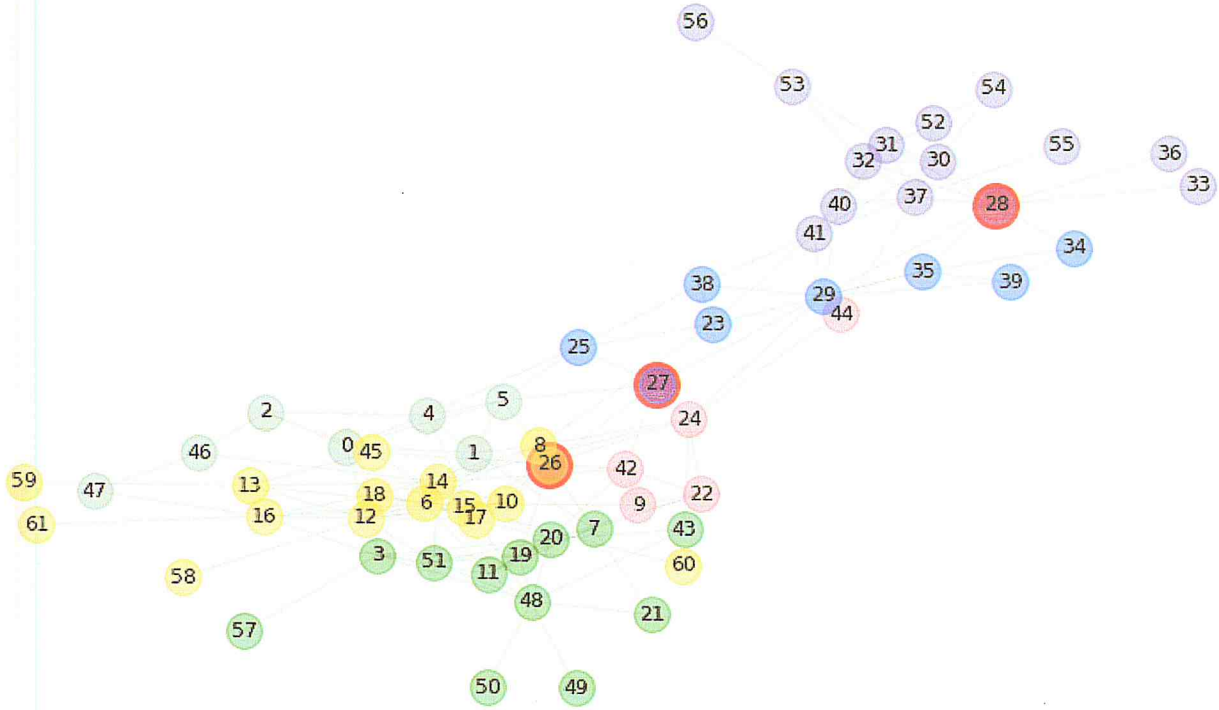


Figure 3.13 - Structure de communautés chevauchantes dans le réseau des dauphins.

Les nœuds chevauchés correspondants sont notés par la couleur rouge. Le résultat c'est dans le cas d'une partition à six communautés obtenue par la méthode Louvain de ce réseau, avec $\alpha = 1.2$, la modularité correspondante $Q_{ol} = 0.4975$. Les nœuds 26, 27, 28 et 31 appartiennent à différentes communautés qui se chevauchent en même temps. Où le nœud 26 appartient aux communautés $C_3 et\ C_4$, le nœud 27 appartient aux communautés $C_4 et\ C_5$ et le nœud 27 appartient aux communautés $C_5 et\ C_6$. Dans le diagramme suivant nous pouvons voir le changement de la modularité par rapport à α.

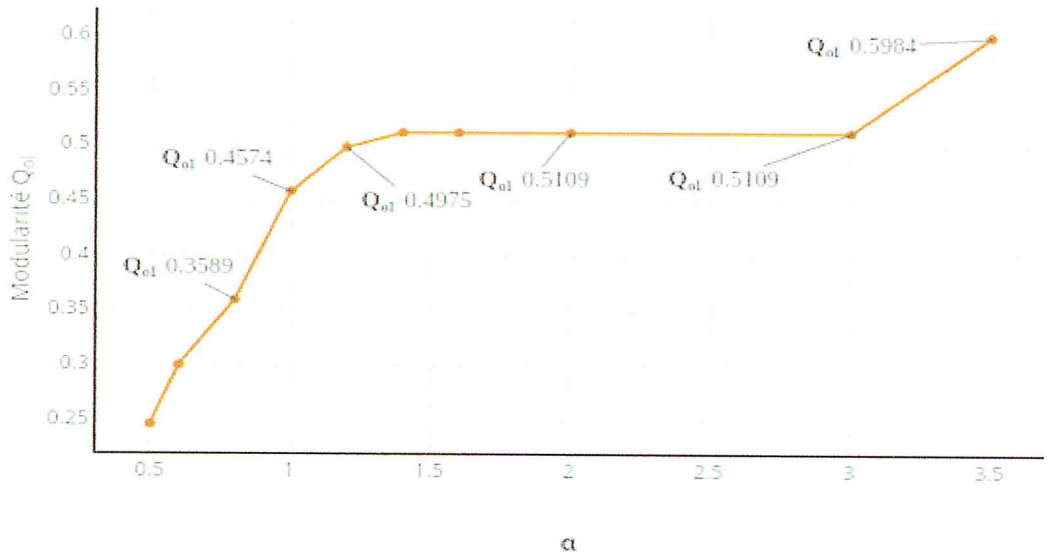


Figure 3.14 - La modularité par rapport au paramètre α pour le réseau des dauphins.

La modularité maximale peut être obtenu pour $\alpha > 1.2$.

5.1.4 Réseau de musiciens de jazz

Le dernier est le réseau de musiciens de jazz, c'est un réseau de collaboration entre les musiciens de jazz. Chaque nœud est un musicien de jazz et un bord indique que deux musiciens ont joué ensemble dans un groupe. Les données ont été recueillies en 2003.

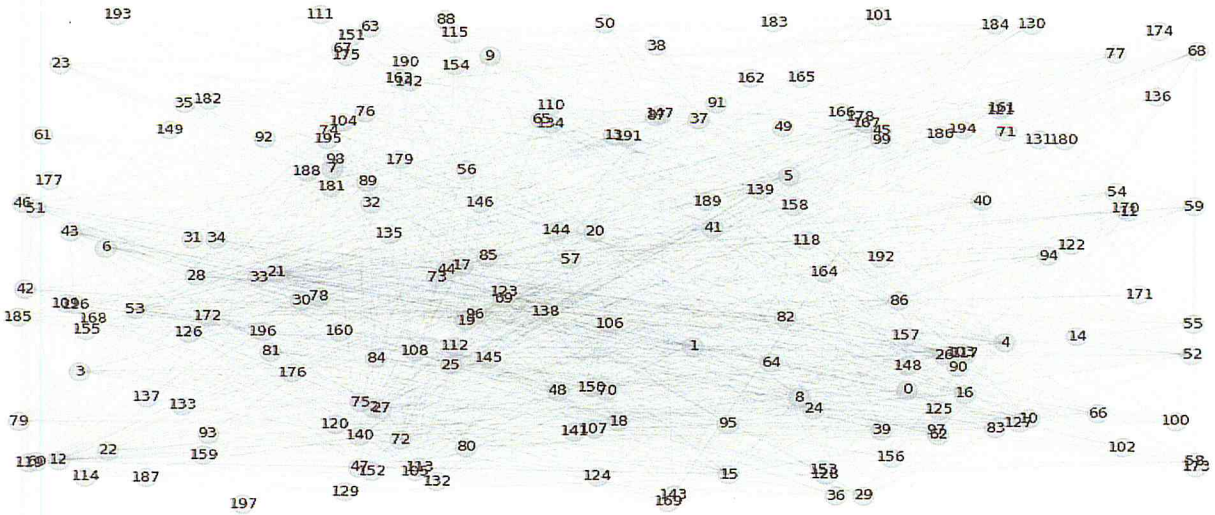


Figure 3.15 - Structure du réseau des musiciens.

La figure suivante montre les nœuds chevauchés pour le réseau social des musiciens obtenu avec notre méthode.

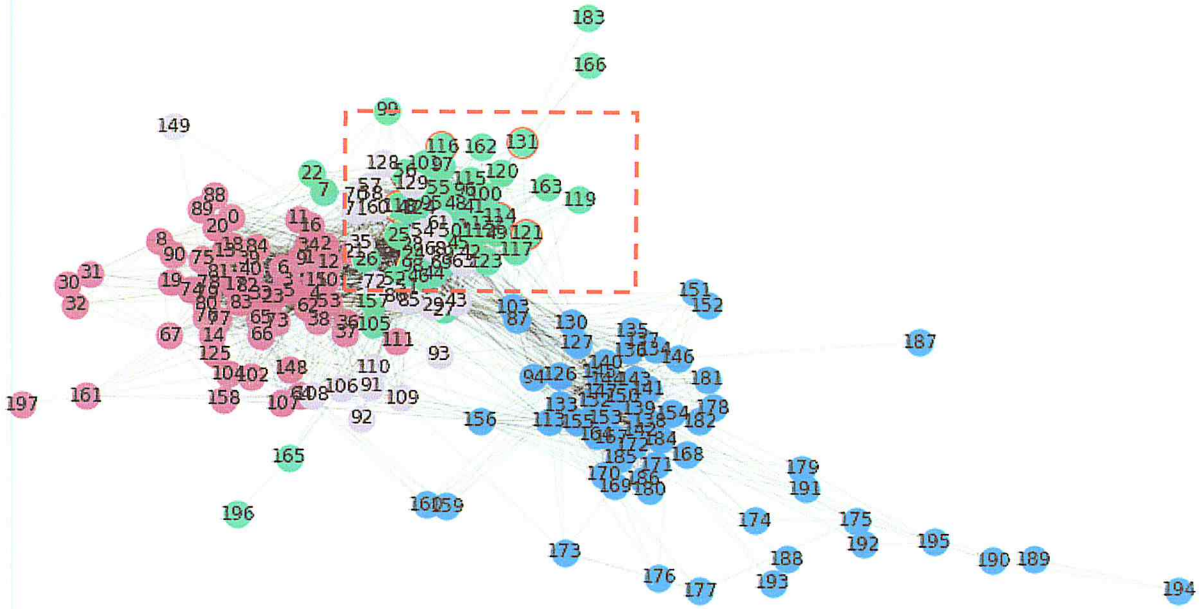


Figure 3.16 - Structure de communautés chevauchantes dans le réseau des musiciens.

Les nœuds chevauchés correspondants sont notés par la couleur rouge.

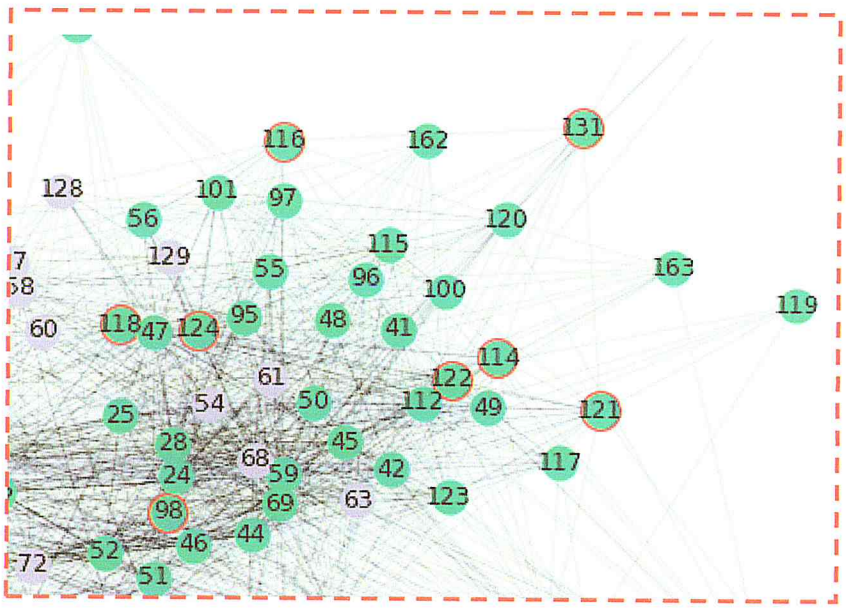


Figure 3.17 - Image agrandie de la figure précédente montre les nœuds chevauchés

.

Le résultat c'est dans le cas d'une partition à six communautés obtenue par la méthode Louvain de ce réseau, avec $\alpha = 1.6$, la modularité correspondante $Q_{ol} = 0.4281$. Les nœuds 98, 114, 116, 118, 121, 122, 124 et 131 appartiennent à différentes communautés qui se chevauchent en même temps. Tous les nœuds appartiennent aux communautés $C_2 et\ C_3$. Dans le diagramme suivant nous pouvons voir le changement de la modularité par rapport à α.

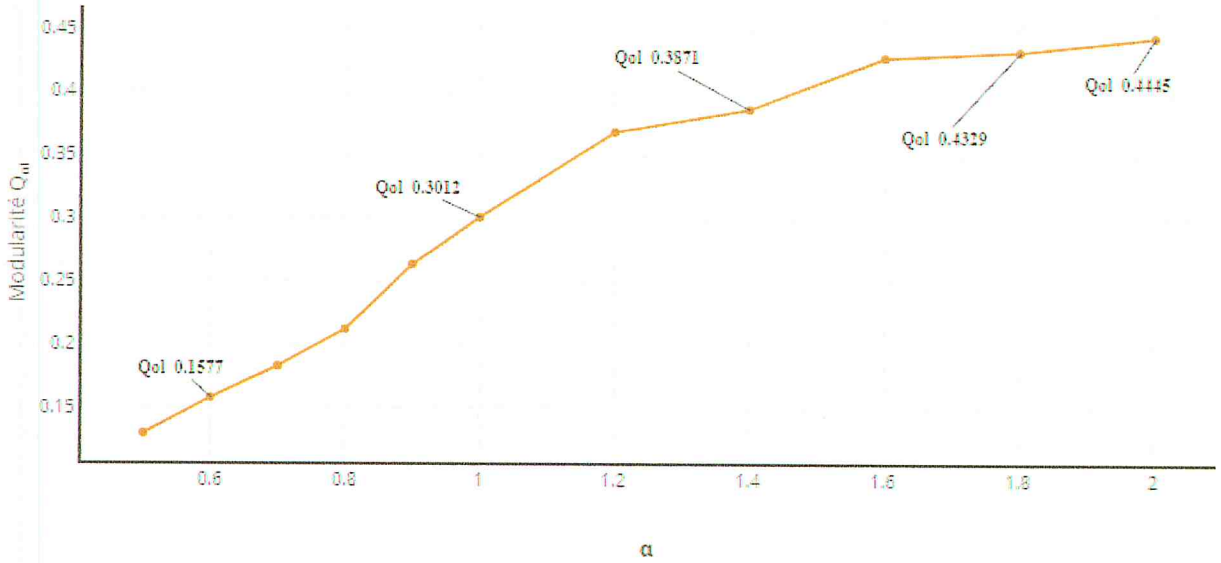


Figure 3.18 - La modularité par rapport au paramètre α pour le réseau des musiciens.

La modularité maximale peut être obtenu pour $\alpha > 1.2$.

5.2 Evaluation de l'approche proposée

Nous avons testé notre approche proposée avec différents algorithmes sur des réseaux sociaux répertoriés dans le tableau 3. Au total, 12 algorithmes ont été collectés et testés **[85]**. Ils sont énumérés dans le tableau 4. Nous comparons la modularité de différents algorithmes.

Réseau	N	Référence
Club de karaté **(KR)**	34	**[70]**
Football **(FB)**	115	**[71]**
Les misérables **(LS)**	77	**[75]**

Dauphins (DP)	62	[72]
CA-GrQc (CA)	5242	[76]
PGP (PGP)	10680	[77]

Tableau 3 – Les réseaux réels utilisés dans notre comparaison.

Algorithme	Référence
SLPA	[57]
COPRA	[44]
GCE	[78]
LFM	[55]
Game	[79]
MOSES	[80]
CIS	[81]
LINK	[50]
iLCD	[82]
UOEC	[83]
Infomap	[84]

Tableau 4 – Les algorithmes inclus dans l'évaluation.

Dans la figure suivante, les réseaux sont représentés dans l'ordre du nombre croissant d'arêtes le long de l'axe des x. Les points de connexion des lignes sont destinés à aider le lecteur à différencier les points du même algorithme. L'extension de modularité utilisé ici est la modularité de Shen et al [63], nous supposons que le paramètre $\alpha = 1$.

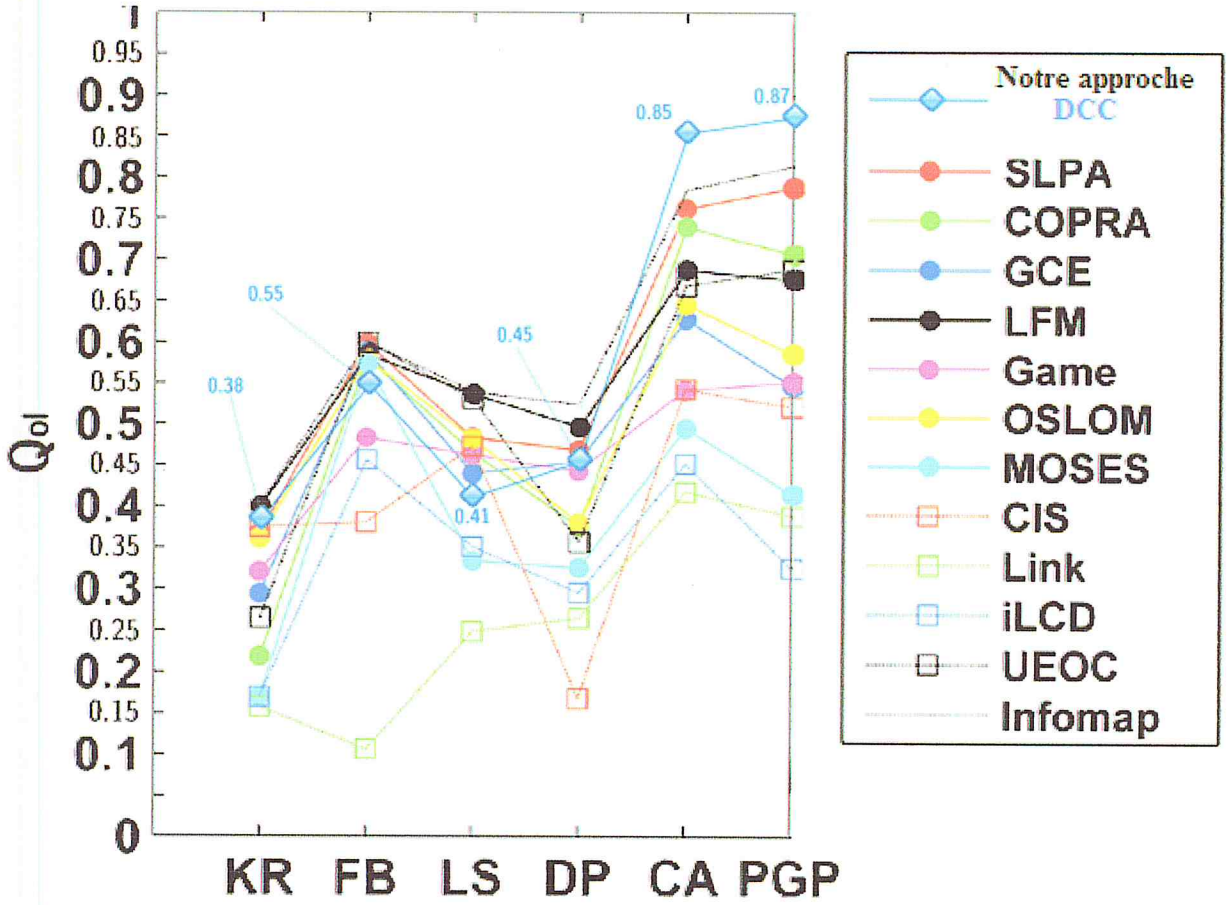


Figure 3.19 – Comparaison de la modularité entre les algorithmes.

En général, Link et iLCD atteignent une modularité inférieure par rapport aux autres. Tandis que notre méthode, SLPA, LFM, COPRA, OSLOM, et GCE atteignent une modularité plus élevée sur les réseaux à grande échelle (par exemple, les deux derniers réseaux), où notre approche marque une bonne valeur de modularité c'est-à-dire un bon partitionnement. Les résultats obtenus permettent de constater que l'algorithme est capable de détecter une structure de communautés chevauchantes significatives.

5.3 Réseaux synthétiques (LFR benchmark)

Nous avons testé l'approche proposées sur un ensemble de réseaux artificiels pour calculer le temps d'exécution. Nous avons choisi le modèle de graphes artificiels présenté par Lancichinetti et al **[69]** dont les degrés des nœuds suivent une loi de puissance. L'algorithme de génération détermine aléatoirement un graphe et une structure communautaire avec des nœuds chevauchants à partir des paramètres suivants :

- Le nombre de nœuds N.
- Le degré moyen de nœud K et le degré de nœud maximum $maxK$.
- L'indice de communauté (mixing parameter), noté μ qui représente la part des arêtes externes d'un nœud, c'est-à-dire des arêtes ayant une extrémité en dehors de la communauté du nœud.
- L'exposant de distribution en loi de puissance des degrés de nœuds, noté τ_1.
- L'exposant de distribution en loi de puissance des tailles de communautés, noté τ_2.
- Les tailles minimale et maximale de communautés, notées $minC$ et $maxC$.
- Nombre de nœuds qui se chevauchent, noté on.
- Nombre d'appartenances des nœuds qui se chevauchent dans les communautés, noté om.

La liste des graphes LFR générés et les paramètres utilisés sont représentés dans le tableau suivant.

N	K	$maxK$	μ	τ_1	τ_2	$minC$	$maxC$	on	om
1000	15	50	0.2	2	1	20	50	100	4
5000	15	50	0.2	2	1	20	50	500	4
10000	15	50	0.2	2	1	20	50	1000	4
50000	15	50	0.2	2	1	20	50	5000	4
100000	15	50	0.2	2	1	20	50	10000	4

Tableau 5 – Liste des graphes LFR générés.

Nous considérons que l'ensemble de données généré par cet ordre de paramètres peut représenter une sorte de réseaux réels.

Le temps d'exécution a été calculé sur une machine avec les caractéristiques (soft et hard) suivantes :

Système d'exploitation	**CPU**	**RAM**	**Disque dur**	**Implémentation de programme**
Windows 8.1 64 bits	Intel i3-3120m 2.5GHz	4 Go	500 GO	Python 3.6

Tableau 6 – Caractéristique (Hard est Soft) de la machine.

La figure suivante montre le résultat obtenu après l'exécution de l'ensemble de graphes précédent.

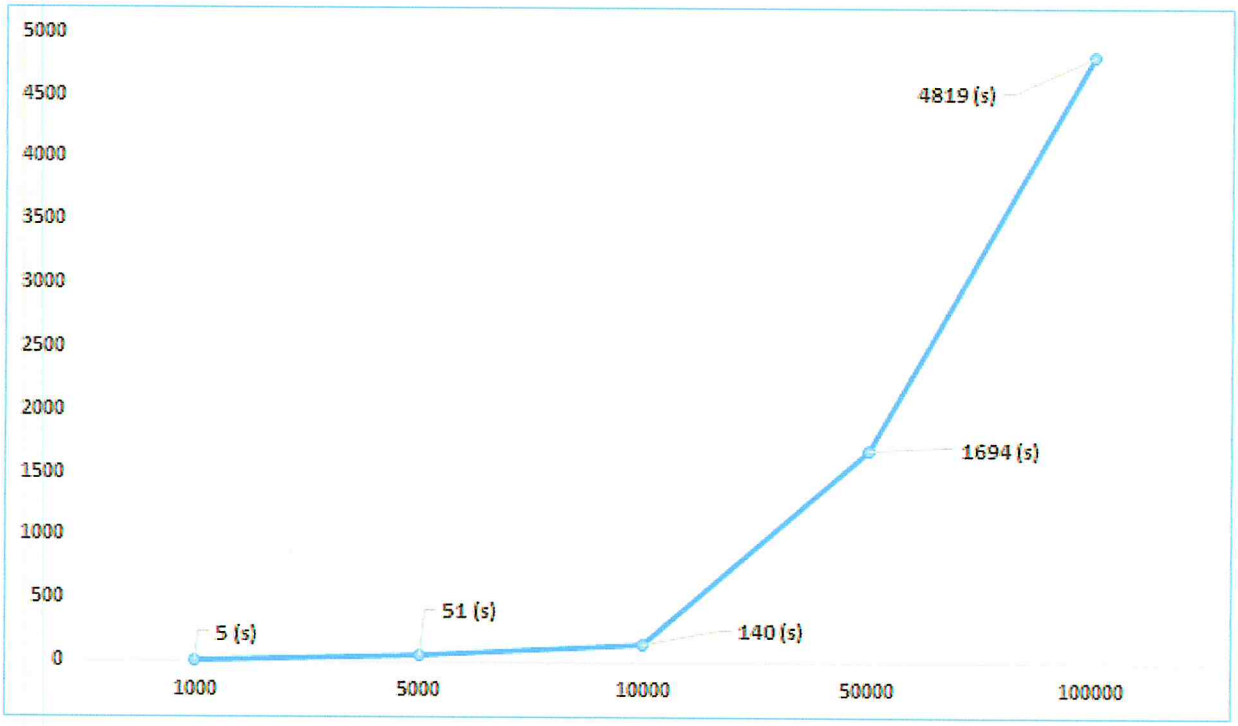


Figure 3.20 – le temps d'exécution de notre approche (DCC) sur des graphes LFR.

L'axe horizontal est le nombre de nœuds dans un graphe, l'axe vertical est le temps d'exécution en seconde.

Le temps d'exécution de DCC reste presque inchangé pour les réseaux de 1000, 5000 et 10000 nœuds. Et comme le montre la figure, le temps d'exécution des deux derniers réseaux 50000 et 100000 nœuds augmente de manière significative. Dans des graphes très denses, elle peut avoir une complexité et une consommation mémoire prohibitives.

5.4 Conclusion

Dans ce chapitre, nous avons présenté une approche de détection de communautés chevauchantes dans les réseaux sociaux. L'approche proposée contient deux phases.

Dans la première phase, l'algorithme de Louvain utilisé pour générer des communautés disjointes. La deuxième phase consiste à ajuster les nœuds situés à la frontière entre les communautés disjointes de manière à les affecter dans un ensemble de bonnes communautés qui se chevauchent.

Nous avons testé notre approche proposée sur un ensemble de réseaux réels et des réseaux synthétiques. Ensuite, on compare notre approche avec différents algorithmes afin d'évaluer les valeurs de modularité. Notre approche DCC atteint une modularité plus élevée sur des réseaux à grande échelle, elle marque une bonne valeur de modularité c'est-à-dire un bon partitionnement.

Le temps d'exécution de DCC reste presque inchangé pour des réseaux allant de 1000 jusqu'à 10000 nœuds, tandis que dans des graphes très denses, elle peut avoir une complexité et une consommation mémoire prohibitives.

Les résultats obtenus permettent de constater que notre approche est capable de détecter une structure de communautés chevauchantes significatives en assurant une meilleure qualité de partitionnement.

Conclusion générale et perspectives

La détection de communautés est un domaine qui est encore dans une phase d'exploration et pour lequel il faudra encore attendre quelques années avant d'arriver à un stade de maturation. Bien que de nombreux algorithmes de détection de communautés aient été développés récemment dans divers domaines, la grande majorité des algorithmes existants ne trouvent que des communautés disjointes. L'état de l'art a permis de décrire et analyser quelques méthodes existantes et algorithmes pour la détection de communautés disjointes et chevauchantes, ce qui nous a permis de voir que l'algorithme de Louvain basé sur l'optimisation de la modularité qui est une mesure de qualité d'un partitionnement des réseaux à grande échelle.

Dans ce mémoire, nous proposons une approche pour la détection de communautés chevauchantes dans les réseaux à grande échelle. Notre travail est composé de deux phases. Durant la première phase nous avons appliqué l'algorithme de Louvain pour générer des communautés disjointes, il offre un bon compromis entre l'optimisation de la modularité et la vitesse de calcul. Dans la deuxième phase nous proposons un ajustement pour vérifier si les nœuds situés à la frontière entre les communautés (nœuds frontaliers) peuvent être appartenir à plusieurs communautés, en y incluant une fonction basée sur une définition locale de la force de la communauté avec un paramètre α, où une petite valeur de α conduira à un nombre important de nœuds chevauchants.

Enfin, nous validons notre approche en l'appliquant à des réseaux réels et réseaux synthétiques. Ensuite, on compare notre approche avec différents algorithmes afin d'évaluer les valeurs de modularité. En résumé, l'analyse et les expériences nous incitent à croire que notre approche est capable de découvrir des communautés chevauchantes significatives.

Dans le but d'améliorer ce travail, deux suggestions peuvent être émises. La première consiste à adapter cette approche pour des graphes orientés et pondérés. Car dans un certain nombre de réseaux du monde réel tel que les réseaux sociaux, tous les liens d'un réseau n'ont pas la même capacité. En fait, les liens sont souvent associés à des poids qui représentent par exemple la quantité de trafic circulant le long des connexions dans les réseaux sociaux.

La deuxième consiste à chercher de diminuer le temps d'exécution dans des réseaux très denses afin de permettre une meilleure détection des communautés chevauchantes.

Bibliographie

[1] - Jacob L Moreno : Who shall survive, volume 58. JSTOR, 1934.

[2] - Jacob Levy Moreno : Sociometry, experimental method and the science of society. 1951.

[3] - John Arundel Barnes : Class and committees in a Norwegian island parish. Plenum New York, 1954.

[4] - Stanley Wasserman et Katherine Faust : Social network analysis : Methods and applications, volume 8. Cambridge university press, 1994.

[5] - George Peter Murdock : Social structure. 1949.

[6] - Robert Harry Lowie : Social organization. 1950.

[7] - George C Homans : The human group new york. Harpers, 1950.

[8] - S Nadel : The theory of social structure (cohen and west, london). 1957.

[9] - Murray Glanzer et Robert Glaser : Techniques for the study of group structure and behavior : Ii. empirical studies of the efects of structure in small groups. Psychological Bulletin, 58(1):1, 1961.

[10] - Phillip Bonacich : Technique for analyzing overlapping memberships. Sociological methodology, 4:176-185, 1972.

[11] - Ronald L Breiger : The duality of persons and groups. Social forces, 53(2):181-190, 1974.

[12] - Albert-Làszlo Barabàsi et Eric Bonabeau : Scale-free networks. ScientifcAmerican, 288(5):50{59, 2003.

[13] - Mark EJ Newman : The structure and function of complex networks. SIAM review, 45(2):167{256, 2003.

[14] - Aaron Clauset, Cosma Rohilla Shalizi et Mark EJ Newman : Power-law distributions in empirical data. SIAM review, 51(4):661-703, 2009.

[15] - D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks.," Nature,vol. 393, no. 6684, pp. 440–2, 1998.

[16] - Siméon-Denis Poisson, Recherches sur la probabilité des jugements en matière criminelle et en matière civile ; précédées des Règles générales du calcul des probabilités [archive] disponible sur Gallica, 1837, passage 81, p. 205.

[17] - P. Erdos and A. Rényi, "On the evolution of random graphs," Evolution, vol. 5, no. 1, pp. 17–61, 1960.

[18] - S. Milgram, "The small-world problem," Psychology Today, vol. 1, no. 1, pp. 60–67, 1967.

[19] - Beyer, W. H. CRC Standard Mathematical Tables, 28th ed. Boca Raton, FL: CRC Press, p. 531, 1987.

[20] - M. E. J. Newman, "Citebase - who is the best connected scientist ? a study of scientific coauthorship networks," 2004.

[21] - M. Steyvers and J. Tenenbaum, "The large-scale structure of semantic networks : statistical analyses and a model of semantic growth.," Cognitive Science, vol. 29, no. 1, pp. 41–78, 2005.

[22] - R. Albert, H. Jeong, and A.-L. Barabasi, "The diameter of the world wide web," Nature, vol. 401, no. September, p. 5, 1999.

[23] - V. Latora and M. Marchiori, "Is the boston subway system a small-world network," Physica A, vol. 314, pp. 109–113, 2002.

[24] - Q. C. Q. Chen, H. C. H. Chang, R. Govindan, and S. Jamin, "The origin of power laws in internet topologies revisited," 2002.

[25] - H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási, "The large-scale organization of metabolic networks," Nature, vol. 407, no. 6804, pp. 651–654, 2000.

[26] - T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki, "A comprehensive two-hybrid analysis to explore the yeast protein interactome," Proceedings of the National Academy of Sciences, vol. 98, no. 8, pp. 4569–4574, 2001.

[27] - N. Guelzim, S. Bottani, P. Bourgine, and F. Képès, "Topological and causal structure of the yeast transcriptional regulatory network.," Nature Genetics, vol. 31, no. 1, pp. 60–63, 2002.

[28] - R. Albert, "Scale-free networks in cell biology," Journal of cell science, vol. 118, no. 21, pp. 4947–4957, 2005.

[29] - X. Zhu, M. Gerstein, and M. Snyder, "Getting connected : analysis and principles of biological networks.," Genes & Development, vol. 21, no. 9, pp. 1010–1024, 2007.

[30] - B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (1970) 291-307.

[31] - E.R. Barnes, An algorithm for partitioning the nodes of a graph, SIAM J. Algebr. Discrete Methods 3 (1982) 541-550.

[32] - L.C. Freeman, A set of measures of centrality based on betweenness, Sociometry 40 (1977) 35-41.

[33] - U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikolski, D. Wagner, On modularity np-completeness and beyond.

[34] - M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.

[35] - S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671-680.

[36] - S. Boettcher, A.G. Percus, Optimization with extremal dynamics, Phys. Rev. Lett. 86 (2001) 5211-5214.

[37] - Xiaojin Zhu, Zoubin Ghahramani, Learning from Labeled and Unlabeled Data with Label Propagation, CMU CALD tech report CMU-CALD-02-107, 2002.

[38] - U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.

[39] - W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (1977) 452-473.

[40] - Stephen Kelley : The existence and discovery of overlapping communities in large-scale networks. These de doctorat, RENSSELAER POLYTECHNIC INSTITUTE, 2009.

[41] - Andrea Lancichinetti, Santo Fortunato et Janos Kertesz : Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics, 11(3):033015, 2009.

[42] - Conrad Lee, Fergal Reid, Aaron McDaid et Neil Hurley : Detecting highly overlapping community structure by greedy clique expansion. In SNAKDD Workshop, page 4533{42, 2010.

[43] - Jorg Reichardt et Stefan Bornholdt : Statistical mechanics of community detection. Physical Review E, 74(1):016110, 2006.

[44] - Steve Gregory : Finding overlapping communities in networks by label propagation. New Journal of Physics, 12(10):103018, 2010.

[45] - Jianxin Wang, Jun Ren, Min Li et Fang-Xiang Wu : Identi cation of hierarchical and overlapping functional modules in ppi networks. IEEE transactions on nanobioscience, 11(4):386{393, 2012.

[46] - Marta Sales-Pardo, Roger Guimera, Andre A Moreira et Luis A Nunes Amaral : Extracting the hierarchical organization of complex systems. Proceedings of the National Academy of Sciences, 104(39):15224{15229, 2007.

[47] - T.S. Evans, R. Lambiotte, Line graphs, link partitions, and overlapping communities, Phys. Rev. E 80 (1) (2009) 016105.

[48] - Y.-Y. Ahn, J.P. Bagrow, S. Lehmann, Communities and hierarchical organization of links in complex networks.

[49] - V.K. Balakrishnan, Schaum's Outline of Graph Theory, McGraw-Hill, New York, USA, 1997.

[50] - Y.-Y. Ahn, J.P. Bagrow, S. Lehmann, Communities and hierarchical organization of links in complex networks, eprint

[51] - Jaccard, P. (1901) Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines. Bulletin de la Société Vaudoise des Sciences Naturelles 37, 241-272.

[52] - EVANS, T. AND LAMBIOTTE, R. 2010. Line graphs of weighted networks for overlapping communities. Euro. Phys.J. B 77, 265.

[53] - EVANS, T. S. AND LAMBIOTTE, R. 2009. Line graphs, link partitions and overlapping communities. Phys. Rev. E 80, 1.

[54] - FORTUNATO, S. 2010. Community detection in graphs. Phys. Rep. 486, 75–174.

[55] - LANCICHINETTI, A.FORTUNATO, S., AND KERTESZ, J. 2009. Detecting the overlapping and hierarchical community structure of complex networks. New J. Phys. 11, 3.

[56] - Gennaro Cordasco and Luisa Gargano, Community Detection via Semi–Synchronous Label Propagation Algorithms, Int. J. of Social Network Mining 2012 - Vol. 1, No.1 pp. 3 - 26

[57] - Jierui Xie, Boleslaw K Szymanski et Xiaoming Liu : Slpa : Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In Data Mining Workshops (ICDMW), 2011 IEEE 11th Internatio- nal Conference on, pages 344{349. IEEE, 2011.

[58] - V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. P10008 (2008).

[59] - G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (2005) 814-818.

[60] - M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.

[61] - S. Lloyd, Least squares quantization in PCM, IEEE Trans. Inform. Theory 28 (2) (1982) 129-137.

[62] - A. Clauset, Phys. Rev. E 72 (2005) 026132.

[63] - H. Shen, X. Cheng, K. Cai and M. B. Hu, "Detect Overlapping and Hierarchical Community Structure in Net- works," Physica A, Vol. 388, No. 8, 2009, pp. 1706-1712.

[64] - T. Nepusz, A. Petróczi, L. Négyessy, F. Bazsó, "Fuzzy communities and the concept of bridgeness in complex networks" Phys. Rev. E 77 (2008) 016107.

[65] - CHEN, W., LIU, Z., SUN, X., AND WANG, Y. 2010b. A game-theoretic framework to identify overlapping communities in social networks. Data Mining Knowl. Discov. 21, 2, 224–240.

[66] - J.B. MacQueen, Some methods for classification and analysis of multivariate observations, in: L.M.L. Cam, J. Neyman (Eds.), Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, University of California Press, Berkeley, USA, 1967, pp. 281-297.

[67] -J.C. Dunn, A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters, J. Cybernet. 3 (1974) 32-57.

[68] - P. Schuetz, A. Caflisch Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement, Physical Review E, 77 (2008), p. 046112

[69] - Lancichinetti, A., & Fortunato, S. (2009). Benchmarks for testing communitydetection algorithms on directed and weighted graphs with overlappingcommunities. Physical Review E,80, 016118

[70] - W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (1977) 452-473.

[71] - M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. USA 99 (12) (2002) 7821-7826.

[72] - D. Lusseau, The emergent properties of a dolphin social network, Proc. R. Soc. London B 270 (2003) S186-S188.

[73] - P. Gleiser, L. Danon, Community structure in jazz, Adv. Complex Syst. 6 (2003) 565.

[74] - D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson, Behav. Ecol. Sociobiol. 54 (2003) 396.

[75] - coappearance network of characters in the novel Les Miserables. Please cite D. E. Knuth, The Stanford GraphBase: A Platform for Combinatorial Computing, Addison-Wesley, Reading, MA (1993).

[76] - J. Leskovec, J. Kleinberg and C. Faloutsos. Graph Evolution: Densification and Shrinking Diameters. ACM Transactions on Knowledge Discovery from Data (ACM TKDD), 1(1), 2007.

[77] - Marián Boguñá, Romualdo Pastor-Satorras, Albert Díaz-Guilera, and Alex Arenas. Models of social networks based on social distance attachment. Phys. Rev. E, 70(5):056122, 2004.

[78] - Conrad Lee, Fergal Reid, Aaron McDaid, Neil Hurley (Clique Research Cluster, University College Dublin, Ireland), Phys. Rev. E 83, 066107 (2011)

[79] - CHEN, W., LIU, Z., SUN, X., AND WANG, Y. 2010b. A game-theoretic framework to identify overlapping communities in social networks. Data Mining Knowl. Discov. 21, 2, 224–240.

[80] - MCDAID, A. AND HURLEY, N. 2010. Detecting highly overlapping communities with model-based overlapping seed expansion. In Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM'10). 112–119.

[81] - KELLEY, S. 2009. The existence and discovery of overlapping communities in large-scale networks. Ph.D. thesis, Rensselaer Polytechnic Institute, Troy, NY.

[82] - CAZABET, R.,AMBLARD, F., AND HANACHI, C. 2010. Detection of overlapping communities in dynamical social networks. In Proceedings of the 2nd IEEE International Conference on Social Computing (SOCIALCOM'10). 309–314.

[83] - JIN, D., YANG, B., BAQUERO, C., LIU, D., HE, D., AND LIU, J. 2011. A markov random walk under constraint for discovering overlapping communities in complex networks. J. Statist. Mech. 2011, 5.

[84] - ROSVALL, M. AND BERGSTROM, C. T. 2008. Maps of random walks on complex networks reveal community structure. Proc. Natl. Acad. Sci. 105, 1118–1123.

[85] - Jierui Xie, Stephen Kelley, Boleslaw K. Szymanski, Overlapping Community Detection in Networks: the State of the Art and Comparative Study, ACM Computing Surveys 45(4), Article 43 (August 2013).

[86] - M. E. J. Newman, "Citebase - who is the best connected scientist ? a study of scientific coauthorship networks," 2004.

[87] - X. Zhu, M. Gerstein, and M. Snyder, "Getting connected : analysis and principles of biological networks.," Genes & Development, vol. 21, no. 9, pp. 1010–1024, 2007.

[88] - Jaewon Yang et Jure Leskovec : De ning and evaluating network communities based on ground-truth. Knowledge and Information Systems, 42(1):181{213, 2015.

Webographie

[web1] - https://fr.wikipedia.org/wiki/Graphe_simple.

[web2] - https://fr.wikipedia.org/wiki/Graphe_orient%C3%A9

[web3] - https://en.wikipedia.org/wiki/Degree_(graph_theory)

[web4] - https://fr.wikipedia.org/wiki/Voisinage_(th%C3%A9orie_des_graphes)

[web5] - https://fr.wikipedia.org/wiki/Distance_(th%C3%A9orie_des_graphes)

[web6] - https://fr.wikipedia.org/wiki/Diam%C3%A8tre_(th%C3%A9orie_des_graphes)

[web7] - https://fr.wikipedia.org/wiki/Densit%C3%A9_d%27un_graphe

[web8] - https://fr.wikipedia.org/wiki/Matrice_d%27adjacence

[web9] - https://fr.wikipedia.org/wiki/Graphe_complet

[web10] - https://fr.wikipedia.org/wiki/Graphe_connexe

[web11] - https://fr.wikipedia.org/wiki/Sous-graphe

[web12] - https://fr.wikipedia.org/wiki/Clique_(th%C3%A9orie_des_graphes)

[web13] - https://fr.wikipedia.org/wiki/Graphe_connexe

[web14] - https://fr.wikipedia.org/wiki/Partitionnement_de_graphe

[web15] - https://fr.wikipedia.org/wiki/Graphe_al%C3%A9atoire