UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHEMATIQUES ET INFORMATIQUE
APPLIQUEES

PAR
BILLAL MAZOUZ

TRAVAIL EXPLORATOIRE SUR LA DÉTECTION DES COMMUNAUTÉS DANS LES RÉSEAUX

AOÛT 2021

Résumé

La détection de communauté est devenue un outil de plus en plus populaire pour l'analyse et la recherche en lien avec les réseaux. De nombreuses méthodes ont été proposées pour la détection de communautés, et l'une d'entre elles est la classification spectrale.

La plupart des algorithmes de classification spectrale rencontrent des difficultés qui affectent la qualité de partitionnement du graphe. Par conséquent, ce travail propose une méthode de classification spectrale. Elle consiste à agglomérer les nœuds selon la ressemblance et la conductance des arêtes entre eux. Dans cette méthode, la ressemblance entre les nœuds se détermine par une fonction qui combine l'information préliminaire du graphe et l'information obtenue de la projection du graphe dans un espace métrique en utilisant les valeurs et les vecteurs propres.

Afin de montrer l'efficacité de notre approche nous allons la tester sur plusieurs graphes artificiels de différentes tailles en comparant les résultats de cette méthode avec les résultats d'une autre méthode spectrale et les résultats fournis par les benchmarks.

Mots clés : détection de communautés, classification spectrale, théorie des graphes.

Remerciements

Je rends grâce à Dieu, miséricordieux de m'avoir donné la volonté, la persévérance et l'obstination pour réaliser ce travail.

Je tiens à remercier mon directeur de recherche Monsieur Ismaïl Biskri et ma codirectrice Madame Nadia Ghazzali pour m'avoir guidé et encouragé tout au long de ce travail.

Je remercie également, tous ceux et celles qui, de près ou de loin, par leurs encouragements, leur amitié et leur support moral, ont contribué à l'aboutissement de ce travail.

Finalement je remercie les personnes qui me sont très chères : mes parents.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

1.1 Mise en contexte

La théorie des graphes ou la science des réseaux, est une branche de l'étude des graphes, qui sont des structures mathématiques utilisées pour modéliser des relations entre des objets. Un réseau (ou graphe) contient un ensemble de sommets (ou nœuds) et un ensemble d'arêtes tel que chaque arête relie une paire de sommets (Strogatz, 2001; Girvan et Newman, 2002). Généralement, l'utilité des réseaux est la représentation de relations entre objets. Par exemple, dans un réseau routier entre plusieurs villes, les villes sont considérées comme les nœuds du graphe et les routes entres celles-ci comme les arêtes du graphe (LeBlanc *et al.*, 1975). En biologie les interactions protéines-protéines peuvent être représentées par un graphe tel que les protéines sont les nœuds et les arêtes représentent l'interaction entre les protéines (Iman *et al.*, 2014). Dans les réseaux sociaux, par exemple, Facebook est l'un des grands réseaux sociaux où plus de deux milliards d'utilisateurs sont connectés via ce réseau, dans ce cas les nœuds peuvent représenter les utilisateurs et les arêtes peuvent représenter le lien d'amitié (Otte et Rousseau, 2002). Il y a beaucoup d'autres domaines où les réseaux sont utilisés tel que la physique, l'économie, l'ingénierie, l'informatique, l'écologie, le marketing, les sciences sociales, la politique, etc. La figure 1.1 montre un autre exemple célèbre, soit Internet,

où nous voyons le réseau physique d'ordinateurs, de routeurs et de modems qui sont interreliés par des câbles ou des signaux sans fil dans le monde en 2003.

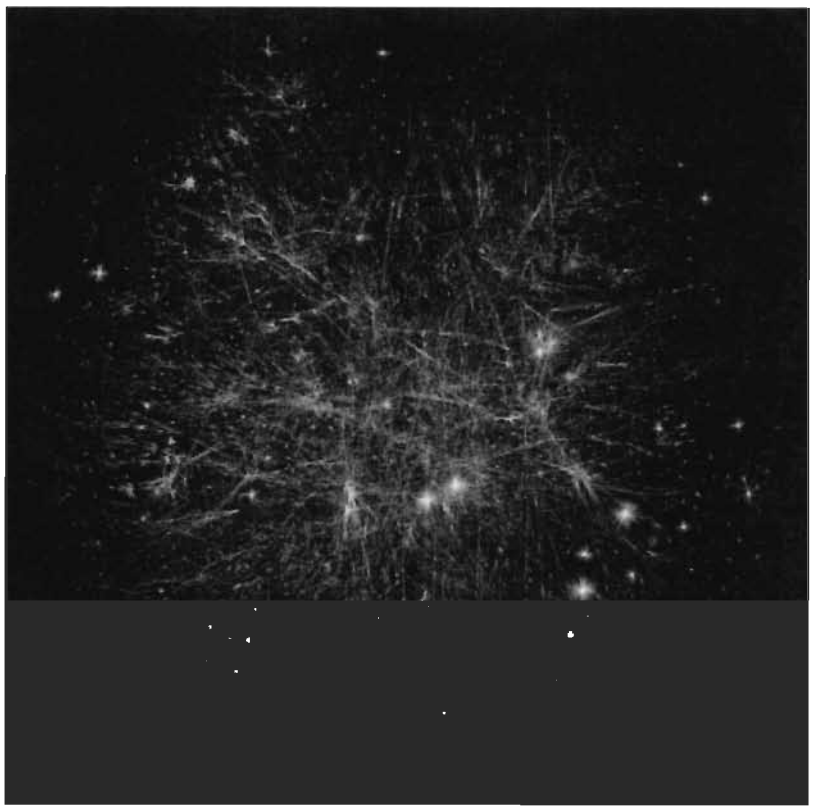


FIGURE 1.1: Réseau Internet. téléchargé depuis www.opte.org.

Dans les réseaux d'intérêt, l'organisation des nœuds et la façon dont ils sont reliés montrent une structure communautaire. Les nœuds d'un réseau d'intérêt donné sont organisés en groupes, appelés communautés, classes ou modules (Fortunato et Hric, 2016). Une communauté peut être définie comme un groupe d'entités plus proches les unes des autres par rapport aux autres entités du réseau. La communauté est formée par des objets. Les objets d'une communauté interagissent les uns avec les autres plus fréquemment qu'avec les objets extérieurs à cette communauté. La proximité entre les objets d'un groupe peut être évaluée par des mesures de similarité ou de distance. En biologie, les protéines ayant une fonction similaire dans les réseaux d'interactions

protéine-protéine peuvent former une communauté. La communauté dans les réseaux routiers peut être un ensemble de villes qui ont des petites distances entre elles. Sur les réseaux sociaux les communautés peuvent être des groupes d'utilisateurs qui habitent dans la même région ou qui travaillent pour la même entreprise. Le réseau Internet peut avoir des communautés qui sont des groupes de sites Web portant sur des sujets similaires. La figure 1.2 montre un graphe de trois communautés.

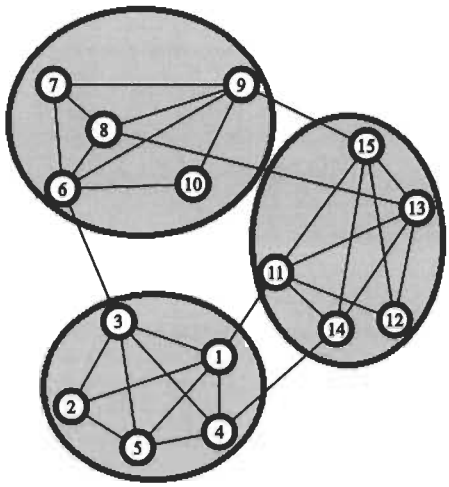


FIGURE 1.2: Exemple d'un graphe de trois communautés

L'identification des communautés dans un réseau peut donner un aperçu sur la façon dont le réseau est organisé. Elle nous permet aussi de comprendre la structure de ce réseau. Depuis la naissance de ce problème appelé détection de communauté dans les réseaux, plusieurs études (Strogatz, 2001; Girvan et Newman, 2002; Radicchi *et al.*, 2004; Hu *et al.*, 2008; Ding *et al.*, 2016; Girvan et Newman, 2004) ont été menées pour apporter une solution élégante à ce problème. Notons, toutefois que le problème de détection de communauté reste mal défini. À ce jour il n'a pas de formulation mathématique.

La détection de communauté dépend tout d'abord des conditions à remplir par une communauté. Une des définitions les plus populaires est : «la densité de la connexion des arêtes doit être plus élevée à l'intérieur de la communauté qu'à l'extérieur » (Yang et

Leskovec, 2012). Ainsi, plusieurs méthodes et algorithmes ont été inventés pour donner une solution approximative au problème de détection de communauté dans les réseaux. Nous trouvons dans la littérature plusieurs catégories de méthodes dont celle-ci :

1. Des méthodes basées sur la classification. Nous distinguons deux types de classification : la classification supervisée et la classification non supervisée. Cette dernière est la plus utilisée dans la détection de communauté. Nous avons, par exemple, la classification hiérarchique divisive ou agglomérative, ainsi que la classification par partitionnement dont l'application de l'algorithme $K - means$ qui prend en entrée le nombre prédéterminé de classes K.
2. Des méthodes basées sur l'optimisation d'une fonction objective. Cette dernière peut décrire la qualité de partitionnement d'un graphe. Généralement, plus la valeur de cette fonction est élevée plus la qualité du partitionnement est meilleure.
3. Des méthodes basées sur les concepts de la théorie des graphes. Elles prennent en considération l'importance des nœuds et des arêtes dans le graphe en utilisant des mesures comme la centralité du graphe, la centralité intermédiaire des nœuds et la centralité intermédiaire des arêtes. Une des méthodes les plus utilisées est basée sur la détection des sous-graphes dont les nœuds sont tous connectés entre elles.
4. Des méthodes basées sur les propriétés spectrales des matrices qui peuvent être associées à un graphe, comme la matrice d'adjacence et la matrice Laplacienne. Cette catégorie de méthodes consiste à projeter le graphe dans un espace métrique engendré par les vecteurs propres d'une matrice associée à ce graphe en générant des données numériques. Ensuite, elles appliquent des différentes méthodes de classification sur ces données numériques, pour donner un partitionnement du graphe.

1.2 Motivations

La classification spectrale est une méthodes populaire pour la détection de communautés. Plusieurs approches spectrales ont été proposées au cours des cinquante dernières années. Parmi les travaux existants, certains (Fiedler, 1973, 1975; E.R.Barnes et A.J.Hoffman, 1981) ont proposé d'utiliser le vecteur propre associé à la plus petite valeur propre non nulle. Ce vecteur est appelé le vecteur de Fiedler. L'information stockée dans ce vecteur est exploité pour générer une bissection du graphe. D'autres chercheurs (Shi et Malik, 2000; Ng *et al.*, 2001), ont proposé d'utiliser l'information stockée dans le reste des vecteurs propres. Ils ont utilisé l'algorithme $K-means$ pour diviser le graphe en K communautés, K étant le nombre de vecteurs choisis. Certaines approches spectrales utilisent des techniques d'optimisation, l'un des algorithmes les plus connus est le *Leading Eigenvector* proposé par (Newman, 2006a). Il consiste à utiliser les propriétés spectrale du graphe, en optimisant une fonction objective. D'autres approches spectrales utilisent des techniques de classification hiérarchique (divisive ou agglomérative), qui consistent à former des communautés de façon hiérarchique à partir d'une projection dans un espace métrique, en utilisant une mesure de ressemblance. En dépit de ces méthodes spectrales existantes, la détection de communauté dans les graphes continue de représenter un défi à la classification spectrale. En effet, les méthodes existantes présentent un ou plusieurs inconvénients parmi les suivants :

1. Certaines méthodes exigent de fournir le nombre de communauté en entrée à l'algorithme, ce qui n'est pas toujours possible car l'utilisateur ne peut pas toujours connaitre exactement ce nombre. Les résultats dépendent fortement de ce paramètre d'entrée. Ils sont différents lorsque ce paramètre change. Les résultats peuvent donner des communautés vides qui ne contiennent aucun nœud, ou des communautés dominantes qui peuvent contenir la majorité des nœuds.
2. Les méthodes spectrales qui utilisent des techniques de classification hiérarchique ont de la difficulté à déterminer à quel moment l'algorithme doit s'arrêter. En

effet, l'approche spectrale hiérarchique divisive considère le graphe comme une seule communauté au début et l'algorithme s'arrête quand le graphe est divisé de façon que chaque nœud représente une communauté. Dans le cas de l'approche spectrale hiérarchique agglomérative chaque nœud est considéré comme une communauté, puis les nœuds seront fusionnés. L'algorithme s'arrête quand tous les nœuds sont agglomérés en une seule communauté. Pour les deux approches, divisive et agglomérative, les résultats ne montrent pas une structure communautaire, ce qui conduit à appliquer un autre processus d'évaluation de la qualité de partitionnement pour chaque itération de l'approche divisive ou agglomérative.

3. Quelques méthodes spectrales se basent seulement sur les données qui sont obtenues à partir de la projection du graphe dans l'espace métrique. Celles-ci ne sont pas toujours suffisantes car les algorithmes qui sont utilisés pour calculer les valeurs et les vecteurs propres ne sont pas exacts, ce qui peut causer une perte d'information et donc une déformation de la structure originale du graphe.

1.3 Contribution

Nous proposons dans le cadre de ce travail une nouvelle approche de détection de communauté dans les graphes. Celle-ci est une classification spectrale agglomérative. Notre approche se distingue par ce qui suit :

1. Elle consacre la détection de communautés de manière automatique. L'utilisateur n'a pas besoin de fournir le nombre de communautés dans les paramètres d'entrée. Le seul paramètre d'entrée requis est le graphe, qui peut être représenté par plusieurs structures mathématiques différentes.

2. L'agglomération des nœuds se fait sur la base d'une mesure de ressemblance que nous avons proposée. Celle-ci est composée de plusieurs facteurs. Certains facteurs sont extraits à partir de l'information obtenue de la projection du graphe dans

l'espace métrique. D'autres facteurs sont extraits de l'information préliminaire du graphe. Cette combinaison contribue à diminuer la perte d'information et conserver plus d'information sur la structure du graphe.

3. Le processus d'agglomération des nœuds, consiste à fusionner les nœuds deux par deux. Il a la capacité de faire plus qu'une agglomération dans chaque itération, ce qui diminue le temps de calcul. Avant chaque agglomération, des conditions doivent être vérifiées pour assurer que cette agglomération respecte la définition de la bonne communauté, et aussi pour obtenir une structure communautaire significative. L'algorithme s'arrête lorsqu'aucune agglomération ne peut satisfaire les conditions. Le résultat obtenu à cette étape est le résultat final de détection de communautés.

1.4 Organisation et structure de mémoire

Notre mémoire est organisé comme suit :

Le premier chapitre est une introduction pour bien situer notre travail. Nous avons donné un aperçu sur le problème de détection de communautés, puis nous avons abordé quelques obstacles rencontrés par certaines méthodes de détection de communautés. Nous avons présenté les étapes que nous allons appliquer pour contribuer à éliminer ces obstacles.

Le deuxième chapitre présente des notions et des outils de la théorie des graphes nécessaires pour la compréhension de notre travail. De plus, nous exposons un résumé de quelques travaux importants pour la détection de communautés dans les réseaux. Nous en ressortons quelques définitions sur la structure communautaire ainsi que certaines méthodes de détection de communautés.

Dans le troisième chapitre nous abordons la méthode de détection de communautés que nous proposons en détaillant chaque étape. Nous expliquons chacune des formules utilisées dans notre approche en illustrant chaque étape par un exemple.

Au quatrième chapitre, nous présentons nos expérimentations. Nous appliquons notre algorithme sur plusieurs exemples du graphe générés par des benchmarks. Nous comparons les résultats de notre approche avec les solutions données par les benchmarks et aussi avec les résultats donnés par l'algorithme «Leading Eigenvector».

Nous terminons ce mémoire par une conclusion générale et les perspectives de ce travail.

Chapitre 2

Revue de la littérature

Au cours des dernières années, plusieurs définitions de la communauté ont été proposées. De plus, il existe plusieurs algorithmes de détection de communautés dans les réseaux. Ils peuvent être regroupés en catégories selon différents critères. Dans ce chapitre, nous allons présenter quelques notions relatives aux graphes. Ensuite, nous nous concentrons sur les définitions de la communauté. Nous présenterons également quelques catégories d'approches de détection de communautés.

2.1 Concepts fondamentaux

2.1.1 Notions relatives aux graphes

Un graphe généralement noté G est une structure mathématique, plus exactement un couple (V, E) où $V = \{v_1, ...v_n\}$ est l'ensemble des nœuds (ou sommets) et E est l'ensemble des arêtes (ou liens) $E = \{e_1, ...e_m\}$. La notation usuelle pour définir un graphe G est $G = (V, E)$.

Une arête $e_k \in E$ est un couple de nœuds (v_i, v_j) reliant les sommets v_i et v_j.

Nous notons $|V| = n$ l'ordre du graphe, c'est-à-dire le nombre de nœuds du graphe et $|E| = m$ le nombre d'arêtes du graphe.

Deux nœuds $(v_i, v_j) \in V$ sont considérés voisins (ou adjacents) si une arête les relie, c'est-à-dire s'il existe une arête $e \in E$ telle que $e = (v_i, v_j)$.

Il existe des graphes orientés (ou dirigés) et des graphes non orientés (non dirigés). Un graphe orienté est un graphe pour lequel les liens entre les nœuds sont des arcs, qui sont des arêtes orientées, c'est-à-dire si deux nœuds v_i et v_j sont connectés dans un graphe orienté alors $v_i \rightarrow v_j \neq v_j \rightarrow v_i$. Ce qui n'est pas le cas pour un graphe non orienté tel que $v_i \rightarrow v_j = v_j \rightarrow v_i$.

Un graphe peut contenir des boucles. Une boucle est une arête e d'un nœud v_i vers lui-même, c'est-à-dire $e = (v_i, v_i)$. Un graphe peut également être pondéré, lorsqu'il existe une fonction $W : e \in E \rightarrow \mathbb{R}$ qui associe une valeur réelle à chaque lien, la notation d'un graphe pondéré est $G = (V, E, W)$.

Un graphe simple, c'est un graphe $G = (V, E)$ qui est non orienté, non pondéré et qui ne contient aucune boucle.

Le degré K_i d'un nœud $v_i \in V$ est le nombre d'arêtes incidentes à v_i.

Un graphe est dit complet si tous ses sommets sont adjacents deux à deux :

$$\forall (v_i, v_j) \in V \times V,\ e = (v_i, v_j) \in E$$

Un sous-graphe $G' = (V', E')$ de G, avec $V' \subset V,\ E' \subset E$ est composé des sommets de V' et des arêtes de E ayant leurs deux extrémités dans V'.

Une clique $G' = (V', E')$ est un sous-graphe de G où tous les couples de sommets de V' sont reliés par une arête, c'est donc un sous-graphe complet de G.

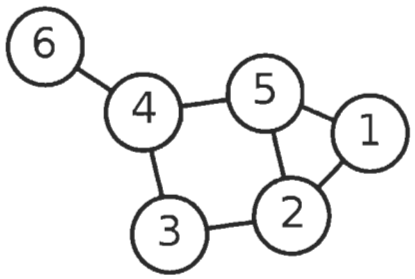
Dans un graphe simple, une chaîne reliant le sommet v_i au sommet v_j, notée $\sigma(v_i, v_j)$, est définie par une suite finie d'arêtes consécutives dont les extrémités sont les sommets v_i et v_j. La notion correspondante dans les graphes orientés est celle de chemin.

Un graphe G est dit connexe si $\forall(v_i, v_j) \in V \times V$ deux sommets quelconques peuvent être reliés par une chaîne.

Une composante connexe d'un graphe G est un sous-graphe connexe.

Représentation d'un graphe

Il existe plusieurs structures permettant de représenter un graphe. L'une des structures les plus intuitives est la représentation matricielle. Un graphe peut être représenté par une matrice d'adjacence notée A de taille $n \times n$ où $n = |V|$ (voir Figure 2.1), dont l'élément non-diagonal principal noté A_{ij} représente le nombre d'arêtes liant le nœud v_i au nœud v_j. Dans un graphe simple, la diagonale principale de la matrice ne comprend que des zéros. Cette représentation permet d'avoir des informations sur la topologie du graphe et sur les relations entre chaque paire de nœuds.



$$A = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Graphe Matrice d'adjacence

FIGURE 2.1: Exemple d'un graphe et sa matrice d'adjacence

2.1.2 Théorie spectrale des graphes

La théorie spectrale des graphes (Nascimento et Carvalho, 2011) s'intéresse aux propriétés spectrales des différentes matrices qui peuvent être associées à un graphe. C'est une branche de la théorie algébrique des graphes. Elle s'intéresse, en général, à la matrice d'adjacence et à la matrice Laplacienne. La matrice des degrés D est une matrice diagonale où les éléments D_{ii} correspondent au degré de chaque nœud K_i. Chaque élément de la diagonale principale de la matrice D égal à la somme de tous les éléments de la ligne qui lui correspond dans la matrice d'adjacence. La matrice Laplacienne est la soustraction de la matrice d'adjacence A de la matrice des degrés D :

$$L = D - A$$

La matrice Laplacienne L et la matrice des degrés pour le graphe de la figure 2.1 sont comme suit :

$$\underset{D}{\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}} - \underset{A}{\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}} = \underset{L}{\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}}$$

Matrice des degrés Matrice d'adjacence Matrice Laplacienne

L'étude des propriétés spectrales, plus précisément des valeurs et des vecteurs propres de la matrice Laplacienne permet également l'obtention de l'information sur la topologie du graphe. Soit L une matrice Laplacienne associée à un graphe G, les vecteurs propres de la matrice L sont associés à des valeurs propres. Le calcul des valeurs propres se fait en trouvant les racines d'un polynôme caractéristique qui peut être associé à une matrice carrée. Le polynôme caractéristique de la matrice Laplacienne L d'ordre n, noté $p_L(x)$, est défini par :

$$p_L(x) = \det(xI_n - L)$$

Où det est le déterminant des matrices, x est une variable inconnue telle que $x \in \mathbb{R}$ et I est la matrice identité d'ordre n. Ce polynôme est de degré n, donc il a au plus n racines $\lambda_0, \lambda_1, \cdots \lambda_{n-1}$, qui sont les valeurs propres de L. Le vecteur propre X_i associé à la valeur propre λ_i se calcule par la résolution de l'équation suivante :

$$LX_i = \lambda_i X_i$$

La matrice Laplacienne est un outil important dans la théorie spectrale des graphes à cause de ses propriétés (Luxburg, 2007). Par exemple, L est une matrice symétrique et définie positive, ce qui assure que ses valeurs propres sont réelles et supérieur ou

égal à 0. Ainsi, le nombre des valeurs propres qui égal à 0 représente le nombre de composantes connexes du graphe G.

2.1.3 Modularité

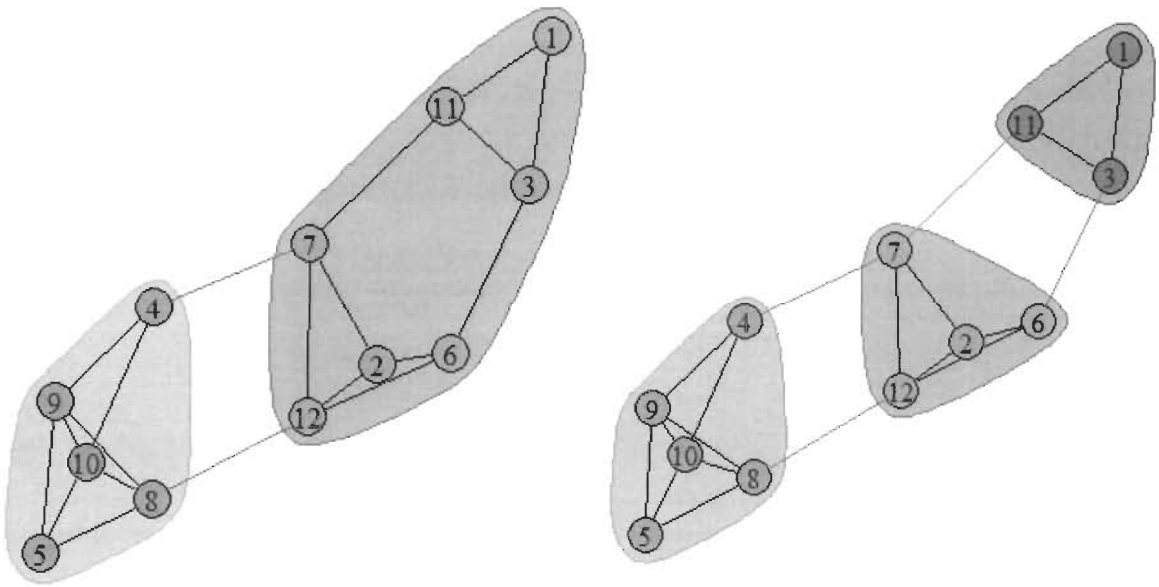
La modularité notée Q, est une mesure proposée par Newman et Girvan (Girvan et Newman, 2004; Newman, 2006b). Elle a pour but de caractériser l'efficacité d'une partition des nœuds d'un graphe. Elle prend en considération le nombre d'arêtes internes des communautés et le nombre d'arêtes entre les communautés. Elle représente la différence entre la proportion des arêtes du graphe qui relient les nœuds internes d'une communauté et la proportion attendue dans un graphe équivalent dans le sens où les nœuds ont la même distribution des degrés, mais les arêtes sont disposées au hasard entre les nœuds du graphe. La modularité Q se calcule comme suit :

$$Q = \frac{1}{2|E|} \sum_{(v_i,v_j)\in V^2} \left[A_{ij} - \frac{k_i.k_j}{2|E|} \right] \delta(c_i, c_j)$$

Où :

- V est l'ensemble des nœuds du graphe.
- $|E|$ est le nombre d'arêtes du graphe.
- A_{ij} vaut 1 si les nœuds v_i et v_j sont connectés et 0 dans le cas contraire.
- k_i et k_j sont respectivement le degré de nœud v_i et le degré de nœud v_j.
- c_i indique la communauté à laquelle le nœud v_i est affecté.
- c_j indique la communauté à laquelle le nœud v_j est affecté.
- δ est le symbole de $Kronecker$ qui vaut 1 si v_i et v_j appartiennent à la même communauté et 0 sinon.
- le couple (v_i, v_j) prend ses valeurs dans $V \times V$ (de façon à prendre en compte les boucles, v_i peut être égal à v_j).

La modularité prend sa valeur entre -1 (lorsque toutes les arêtes sont inter-communautés) et 1 (lorsque toutes les arêtes sont intra-communautés). Selon (Girvan et Newman, 2004), les valeurs de modularité pour les réseaux avec des structures communautaires sont généralement supérieurs à 0.3. Prenons l'exemple de la figure 2.2, où nous remarquons que le même graphe est partitionné de deux façons différentes, (a) en deux communautés et (b) en trois communautés.



(a) Partition en deux communautés (b) Partition en trois communautés

FIGURE 2.2: Exemple d'un graphe partitionné en (a) deux communautés et (b) en trois communautés.

Comment juger quelle est la meilleure partition entre la partition (a) et la partition (b) ? La modularité Q peut répondre à cette question, en calculant la modularité de la partition (a) en deux communautés, nous trouvons $Q = 0.39$. Pour la modularité de la partition (b) en trois communautés, nous trouvons $Q = 0.44$. Nous concluons que la partition (b) en trois communautés est la meilleure partition car la valeur de modularité pour la partition (b) est plus élevée que la valeur de la modularité de la partition (a).

2.2 Structures communautaires

Dans un graphe simple (non orienté et non pondéré) noté G, le nombre de sommets est n et le nombre d'arêtes est m, la matrice d'adjacence associée à G est notée A, ses éléments a_{ij} valent 1 si les sommets i et j sont connectés, sinon ils sont égal à 0. Un sous-graphe G' du graphe G a le nombre de sommets est n_c et le nombre des arêtes est m_c. Les communautés sont généralement connectées à l'intérieur, nous supposons, que les sous-graphes sont connectés à l'intérieur. La communauté est un sous-graphe avec des caractéristiques spécifiques que nous verrons par la suite. La figure 2.3 montre la structure d'un sous-graphe connecté à l'intérieur, ses sommets sont les cinq qui sont entourés par une ellipse. Ses arêtes sont celles qui relient les sommets du sous-graphe entre elles.

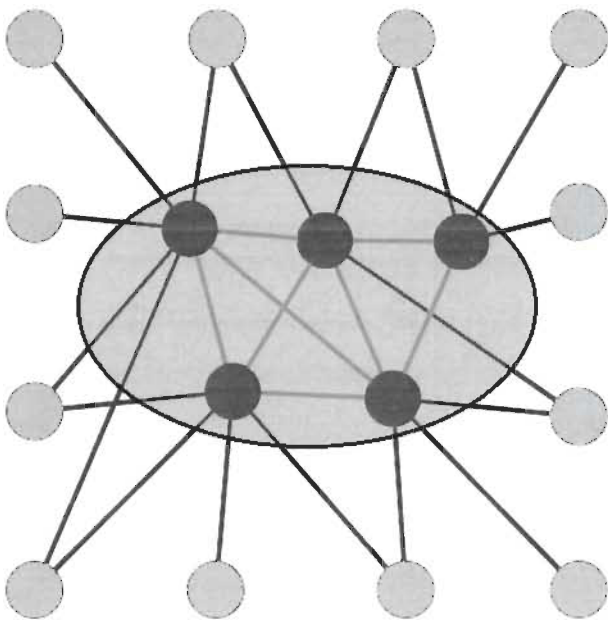


FIGURE 2.3: Structure d'un sous-graphe connecté.

2.2.1 Vision classique

Les chercheurs voient la structure de la communauté comme dans la figure 2.4. C'est un réseau qui contient trois groupes de sommets dont le nombre d'arêtes à l'intérieur de chaque groupe est plus élevé que le nombre d'arêtes entre les groupes (Fortunato et Hric, 2016).

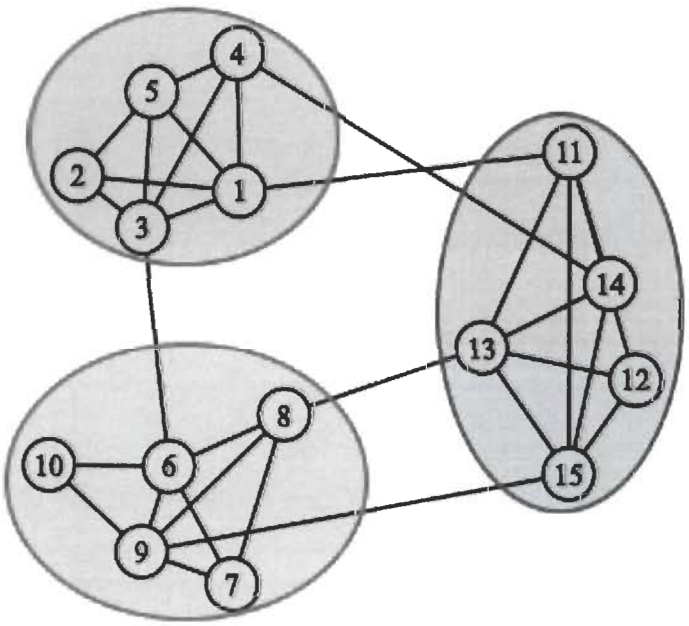


FIGURE 2.4: Vision classique de la structure de la communauté. Structure d'un réseau avec trois communautés.

Les chercheurs ont mis au point des définitions qui sont considérées comme d'anciennes définitions de communautés. Celles-ci se concentrent sur le nombre d'arêtes internes qui relient les sommets d'un sous-graphe. L'un des concepts les plus populaires est celui de clique (Luce et Perry, 1949) qui est été défini dans la section 2.1.1. Il s'agit d'un sous-graphe complet qui n'est pas inclus dans un autre sous-graphe complet plus grand. Par exemple, les triangles dans un graphe sont les cliques les plus simples. L'utilité de la notion de clique est importante, mais elle n'est pas considérée comme un bon

choix pour une définition de communauté.

Il existe aussi des définitions qui sont basées sur le nombre de sommets adjacents à un sommet donné dans un sous-graphe. L'idée est qu'un sommet doit être connecté à un nombre minimum d'autres sommets dans le sous-graphe. La cohésion interne des arêtes de la communauté et sa séparation du reste du réseau sont les deux facteurs qui doivent être pris en considération pour avoir une bonne définition de la communauté.

Un concept populaire de la définition de la communauté est que celle-ci est un sous-graphe tel que « le nombre d'arêtes internes est plus grand que le nombre d'arêtes externes» (Fortunato et Hric, 2016). De ce concept plusieurs définitions en sont ressorties. Considérons une première définition, soit celle de (Radicchi *et al.*, 2004) qui indique que dans le cas d'une faible communauté, le degré interne est supérieur au degré externe. Pour qu'une communauté soit considérée comme étant forte, il faut que le degré interne de chaque sommet soit supérieur à son degré externe. Selon leur définition, une communauté forte est toujours faible, mais une communauté faible n'est pas nécessairement une communauté forte.

Une seconde définition de fortes et faibles communautés provenant du concept a été proposée par (Radicchi *et al.*, 2004). Un sous-graphe est considéré comme étant une communauté forte si le degré interne de n'importe quel de ses sommets est plus élevé que le degré interne du sommet dans n'importe quel autre sous-graphe. Une communauté est faible si son degré interne dépasse le degré interne total de ses sommets dans toutes les autres communautés. Ainsi, une forte communauté au sens de (Radicchi *et al.*, 2004) est aussi une forte communauté selon (Hu *et al.*, 2008). Cependant, une forte communauté au sens de (Hu *et al.*, 2008) n'est pas nécessairement une forte communauté selon (Radicchi *et al.*, 2004). Toutefois, ils s'entendent pour la définition de communautés faibles.

Cela est démontré par un exemple à la figure 2.5 qui est un graphe divisé en quatre sous-graphes représentant quatre communautés faibles selon les définitions de (Radicchi *et al.*, 2004) et (Hu *et al.*, 2008). Selon (Hu *et al.*, 2008), celles-ci sont aussi de fortes communautés. Cependant, selon (Radicchi *et al.*, 2004), trois des sous-graphes qui contiennent un sommet bleu ne sont pas des communautés fortes puisqu'ils ont un degré interne plus petit que leur degré externe.

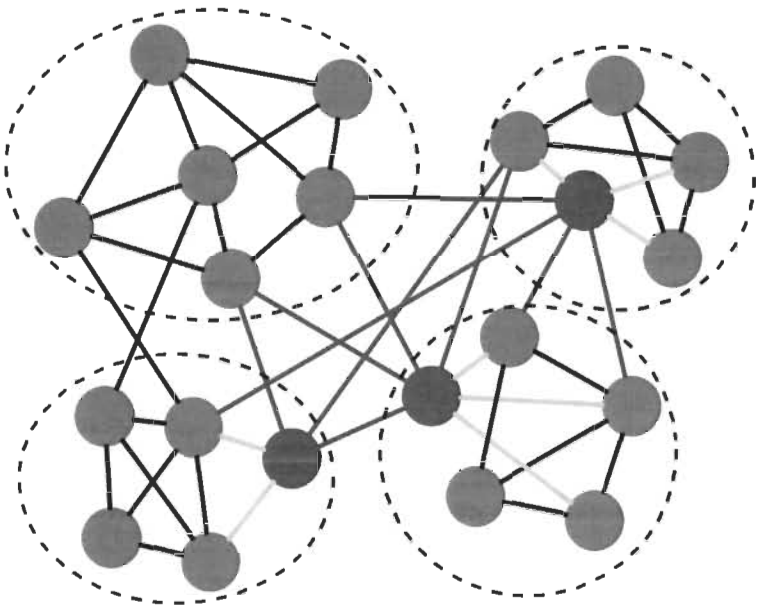


FIGURE 2.5: Communautés fortes et faibles. Source : (Fortunato et Hric, 2016)

2.2.2 Vision récente

Les définitions traditionnelles de la communauté privilégient le nombre d'arêtes internes et externes de la communauté. L'existence de communautés implique que les sommets interagissent plus fortement avec les sommets de leur communauté qu'avec les sommets des autres communautés. Cela peut être formulé en supposant que les sommets de la même communauté ont une probabilité élevée d'avoir des arêtes avec

leurs partenaires qu'avec les autres sommets (Fortunato et Hric, 2016).

Supposons que les probabilités d'avoir une arête entre chaque paire de sommets soient estimées, par une méthode ou une autre. Les communautés peuvent être définies en utilisant ces probabilités. Les définitions d'une communauté forte et faible sont :

- Une communauté forte pourrait être un sous-graphe dont les sommets ont une probabilité plus élevée d'être connecté à chaque sommet du même sous-graphe qu'au reste des sommets du graphe.
- Une communauté faible est un sous-graphe dont la probabilité moyenne que chaque sommet soit lié aux sommets du même sous-graphe est supérieur à la probabilité moyenne d'être lié aux sommets d'une autre communauté.

La différence entre les deux définitions est que, dans le concept de communauté forte, l'inégalité entre les probabilités d'avoir des arêtes se maintient au niveau de chaque paire de sommets, alors que dans le concept de communauté faible, l'inégalité ne vaut que pour les moyennes. Par conséquent, une communauté forte est aussi une communauté faible, mais le contraire n'est pas vrai, en général.

Comment calculer les probabilités d'arête entre les sommets ? Selon (Fortunato et Hric, 2016), c'est encore un problème mal défini, à moins qu'il n'y ait un modèle qui puisse donner une indication sur la façon dont les arêtes sont formées. Le processus de formation des arêtes peut avoir de nombreuses hypothèses. Les réseaux sociaux sont un exemple où la probabilité que deux personnes se connaissent peut être considérée comme une fonction de leur distance géographique (Liben-Nowell *et al.*, 2005).

2.2.3 Communautés chevauchantes

Comme nous venons de le voir, il existe plusieurs définitions de la communauté différentes les unes des autres. Mais un problème qui a été omis est que les communautés peuvent se chevaucher, c'est-à-dire qu'un sommet peut appartenir à plus d'une

communauté (Ding *et al.*, 2016).

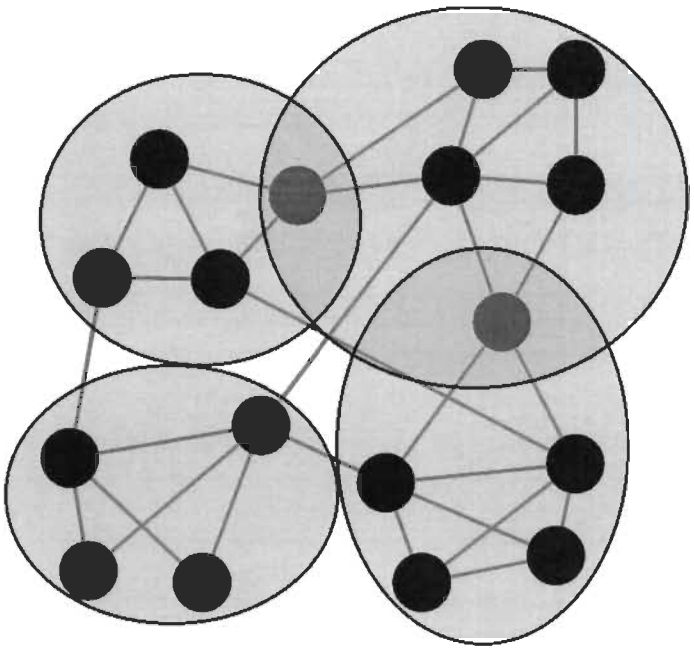


FIGURE 2.6: Communautés chevauchantes : Un réseau de quatre communautés, entouré par les contours en noir. Deux sommets (en bleu), chacun appartenant à deux communautés différentes.

Dans cette optique, prenons l'exemple des réseaux sociaux dans lesquels les utilisateurs peuvent appartenir à plusieurs groupes tels que la famille, les amis, les collègues de travail, etc. Nous pouvons trouver un utilisateur qui est un ami commun avec deux groupes d'amis, dans ce cas, en considérant la relation d'amitié, cet utilisateur appartient à deux groupes différents en même temps. La figure 2.6 montre des communautés chevauchantes dont un sommet appartient à deux communautés. Le fait que les sommets peuvent faire partie de plus d'une communauté augmente l'espace de solutions possibles. Ce problème a conduit à proposer une autre façon de subdiviser le réseau en communautés chevauchantes (Ahn *et al.*, 2010; Evans et Lambiotte, 2009), cette méthode consiste à regrouper les arêtes au lieu des sommets.

2.3 Méthodes de détection de communautés

Au cours des années, plusieurs méthodes de détection de communautés ont été proposées, celles-ci peuvent être regroupées en catégories selon différents critères. Cette section présente quelques catégories de méthodes les plus connues.

2.3.1 Méthodes basées sur la classification

Les méthodes de classification visent à identifier la catégorie d'un objet parmi plusieurs catégories. « Le seul moyen de faire une méthode instructive et naturelle, est de mettre ensemble des choses qui se ressemblent et de séparer celles qui diffèrent les unes des autres» (Buffon, 1749). Les techniques de classification ont généralement pour but de minimiser la dissimilarité entre les individus d'une même classe et de maximiser la dissimilarité entre les individus de deux classes distinctes. La classification automatique peut être divisée en deux catégories soit :

- **Supervisée** : les classes sont connues a priori, elles ont en général une catégorie associée.
- **Non-supervisée** : les classes sont fondées sur la structure des objets, la catégorie associée aux classes est plus difficile à déterminer.

Classification supervisée

Selon (Govaert, 2003), «la classification supervisée (appelée aussi classement ou classification inductive) a pour objectif « d'apprendre » par l'exemple. Elle cherche à expliquer et à prédire l'appartenance d'objets à des classes connues a priori. Ainsi c'est l'ensemble des techniques qui visent à deviner l'appartenance d'un individu à une classe en s'aidant uniquement des valeurs qu'il prend», donc de prédire à partir d'un ensemble d'échantillons bien classé une procédure de classification.

Classification non supervisée

Classification par partitionnement

En général, la seule information préliminaire disponible pour n'importe quel algorithme est la structure du graphe. Quelles paires de nœuds sont connectées entre elles et lesquelles ne le sont pas ? Naturellement, il serait utile d'avoir, au préalable, des informations sur la division inconnue du réseau, car nous pourrions réduire considérablement l'espace de solutions possibles et augmenter les chances d'identifier les classes avec succès. Parmi toutes les entrées possibles, le nombre k de classes joue un rôle important. De nombreux algorithmes requièrent la spécification de k avant leur exécution. L'algorithme $K - means$, qui a été introduit par (MacQueen, 1967), consiste à construire une partition en k classes où k est un paramètre d'entrée pour cet algorithme. Ainsi, le $K - means$ a des avantages et des inconvénients qui sont résumés dans le tableau 2.1.

Avantages	Inconvénients
-Peut traiter un très grand nombre de données	-Le nombre k de classes doit être connu
-Pas de problème de mémoire et de temps de calcul	-L'algorithme converge le plus souvent vers un optimum local
-La convergence est rapide	-Sensible aux exceptions et aux données bruitées
	-Données numériques seules

TABLE 2.1: Avantages et inconvénients de l'algorithme $K - means$

Classification hiérarchique

Une propriété intéressante des réseaux est qu'ils présentent souvent une structure hiérarchique avec de plus petites communautés incluses à l'intérieur de communautés plus grandes. Ce fait justifie un groupe spécial d'algorithmes appelés algorithmes de détection de communautés hiérarchiques qui peuvent révéler la hiérarchie des commu-

nautés dans les réseaux. Ces algorithmes consistent à regrouper les nœuds de notre graphe en différentes communautés de telle sorte que les nœuds d'une même communauté soient le plus similaires possibles (Murtagh et Contreras, 2011). Deux approches de classification hiérarchique existent :

1. **Classification hiérarchique agglomérative**

   Cette approche ascendante (Murtagh et Contreras, 2011), considère initialement que chaque nœud du graphe est une communauté donc, en partant de n communautés, le but est d'arriver à les rassembler en une seule et même communauté grâce à un processus itératif. Dans la version de base, à chaque étape, les deux communautés qui sont les plus similaires seront fusionnées en une même communauté. Plus nous avançons dans ce processus, plus nous aurons des communautés dont les nœuds seront de moins en moins similaires.

2. **Classification hiérarchique divisive**

   Cette approche descendante (Murtagh et Contreras, 2011), considère le graphe en entier comme une communauté qui sera divisé tout au long d'un processus itératif dont le but est d'arriver a un graphe avec n communautés, où chaque communauté est constituée d'un seul nœud. À chaque étape de ce processus, une partition sur le graphe est faite afin de couper notre graphe en sous-graphe (qui représentent nos communautés) sur lesquels la même méthode sera appliquée dans l'itération suivante et ce pour avoir des communautés de plus en plus petites jusqu'à arriver à n communautés.

Bien que la classification hiérarchique présente certains avantages, elle présente également plusieurs inconvénients. L'un d'eux est que cette méthode a tendance à faire d'un sommet une communauté, ce qui n'a généralement pas de sens dans les réseaux. De plus, la similarité de chaque paire de sommets est calculée, même pour les paires de sommets non reliés ce qui augmente considérablement le temps d'exécution des algorithmes.

2.3.2 Classification consensus

La classification consensus est un concept qui s'applique sur des méthodes de classification à caractère stochastique, celles-ci ne fournissent pas toujours la même solution. Ce concept consiste à analyser les informations de différentes solutions pour extraire une meilleure solution (Fortunato et Hric, 2016).

L'objectif de l'approche de la classification consensus (Goder et Filkov, 2008; Streh et Ghosh, 2002; Topchy *et al.*, 2006) est la recherche d'une solution consensus qui combine les informations de différentes solutions. Cette approche est un problème d'optimisation combinatoire difficile consistant à générer une matrice consensus basée sur la co-occurrence de sommets dans les communautés des différentes solutions. La matrice de consensus est utilisée comme entrée pour la technique de classification de graphe adopté, conduisant à un nouvel ensemble de solutions produisant une nouvelle matrice de consensus. Ce processus est répété jusqu'à ce qu'une solution unique soit atteinte. Cette dernière ne sera pas modifiée par d'autres itérations.

L'un des inconvénients de cette méthode est que la matrice de consensus est une matrice pondérée, donc les algorithmes qui utilisent cette méthode doivent être adaptés pour pouvoir être appliqués sur des graphes pondérés. Un autre inconvénient est le temps de calcul de cette méthode, qui est considéré comme étant énorme.

2.3.3 Méthodes basées sur l'optimisation

Dans la littérature, les chercheurs portent une attention significative aux algorithmes basés sur les techniques d'optimisation. Le concept de ces approches est d'optimiser une fonction objective, dont le but est de trouver son maximum en général. Cette fonction objective indique à quel point la qualité du partitionnement d'un graphe est bonne. L'une des fonctions les plus populaires est *la modularité* introduite par (Girvan et

Newman, 2004).

Plusieurs algorithmes sont basés sur la maximisation de la modularité qui est un problème NP-difficile (Brandes *et al.*, 2008). Les algorithmes basés sur cette technique ne donnent qu'une approximation de la modularité maximale du problème de détection de communautés. À ce jour, l'optimisation de la modularité reste la technique de classification la plus utilisée dans les applications. Au cours des dernières années, plusieurs algorithmes basés sur l'optimisation des fonctions de qualité de communautés ont été conçus (Baumes *et al.*, 2005; Clauset, 2005; Huang *et al.*, 2011; Lancichinetti et Fortunato, 2009).

2.3.4 Classification spectrale

La méthode spectrale est dérivée de l'algèbre linéaire. Elle est basée sur la théorie spectrale des graphes que nous avons vu dans la section 2.1.2. Elle consiste à générer une projection des sommets du graphe dans un espace métrique, en utilisant comme coordonnées les entrées des vecteurs propres (voir la figure 2.7). Les $i^{ièmes}$ entrées des vecteurs propres sont les coordonnées du sommet v_i dans un espace euclidien k-dimensionnel, où k est le nombre de vecteurs propres utilisés.

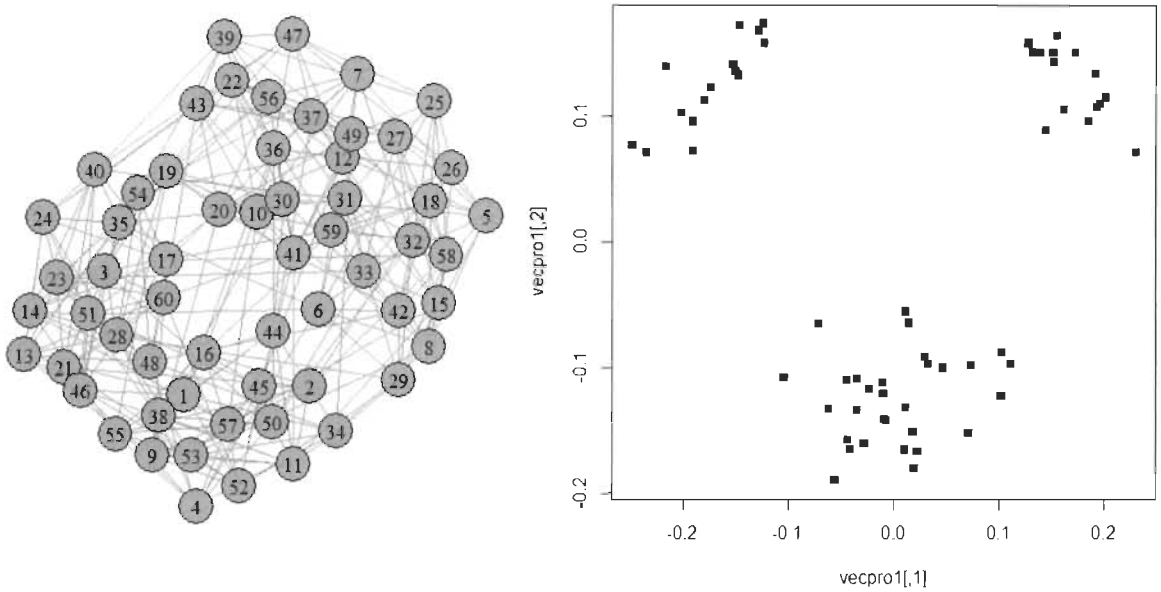


FIGURE 2.7: La projection d'un graphe quelconque dans un espace de deux dimensions engendré par les deux vecteurs propres associés aux deux plus petites valeurs propres non nulles.

Nous avons vu dans la section 2.1.2 que la matrice Laplacienne permet également l'obtention d'information sur la topologie du graphe. Si G est un graphe connexe, la seconde plus petite valeur propre λ_2 est positive. Le vecteur propre associé à λ_2, appelé vecteur de *Fiedler*. Il fut étudié par (Fiedler, 1973, 1975), le principe est d'associer chaque composante dans le vecteur de *Fiedler* au nœud qui lui correspond dans le graphe. Les travaux de (Hall, 1970) ont montré que si deux nœuds v_i et v_j sont connectés, alors la distance $|x_i - x_j|$ dans le vecteur de Fiedler est petite par rapport à une paire de nœuds qui ne sont pas connectés. Ainsi, les nœuds fortement connectés sont plus proches dans le vecteur de Fiedler. La figure 2.8 montre la projection d'un graphe quelconque de 30 nœuds et de communautés sur un espace d'une dimension engendré par le vecteur de Fiedler.

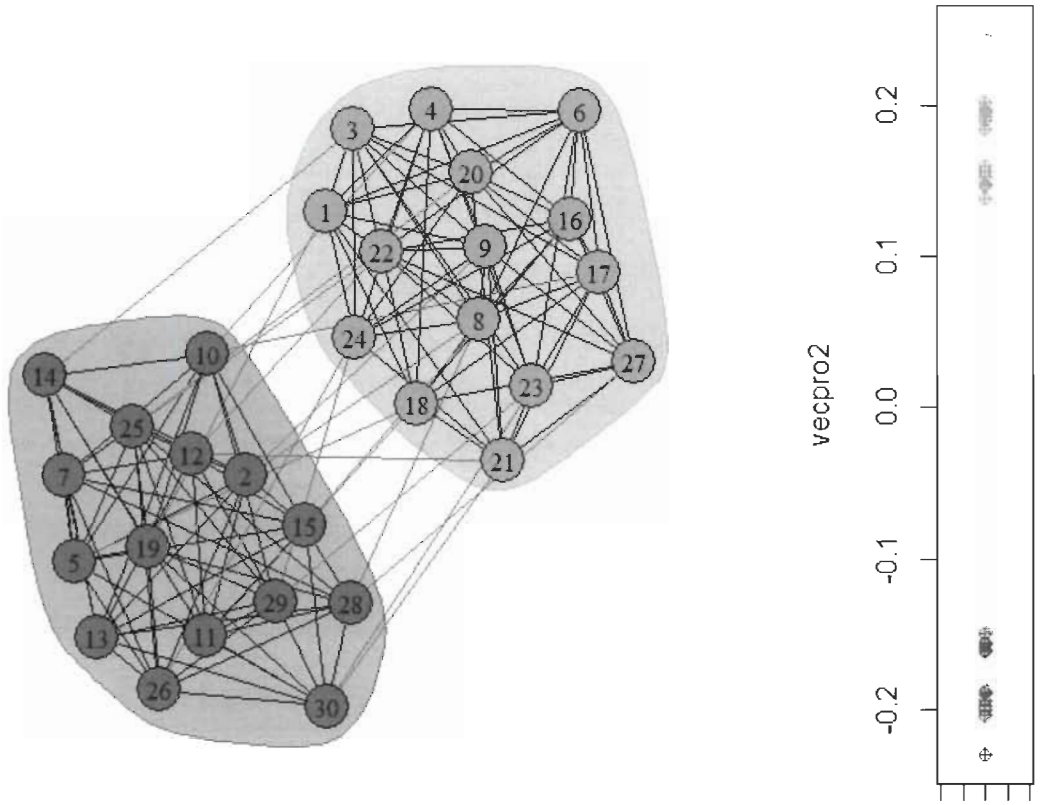


FIGURE 2.8: La projection d'un graphe quelconque de deux communautés dans un espace d'une dimension engendré par le vecteur de Fiedler.

L'une des premières méthodes exploitant ce champ fut la méthode de (E.R.Barnes et A.J.Hoffman, 1981). Cette application permet de donner une bipartition du graphe en deux groupes tels que les nœuds correspondant aux composantes négatives soient mis dans un même groupe et le reste des nœuds correspondant aux composantes positives soient placés dans l'autre groupe.(Shi et Malik, 2000), ainsi que (Ng *et al.*, 2001) eurent l'idée d'utiliser l'information stockée dans d'autres vecteurs propres pour améliorer la qualité du partitionnement. L'idée consistait à utiliser l'algorithme $k - means$ dans l'espace propre afin de trouver k communautés. Cependant, cette méthode nécessite de connaitre la valeur K qui est le nombre de communautés.

Dans les travaux de (Donetti et Munoz, 2004), ils proposèrent d'utiliser les vecteurs propres associés aux K plus petites valeurs propres non-nulles, K étant un entier ne pouvant excéder le nombre de valeurs propres. L'idée est qu'un sommet est représenté dans un espace de dimension K constitué de composantes qui lui correspondent dans les K vecteurs propres. Le meilleur partitionnement est celui qui a la plus grande valeur de la modularité.

La figure 2.9 montre les résultats de la projection d'un graphe de 60 nœuds et de 4 communautés, chacune bien visible dans un espace engendré par les deux vecteurs propres associés aux deux plus petites valeurs propres non nulles.

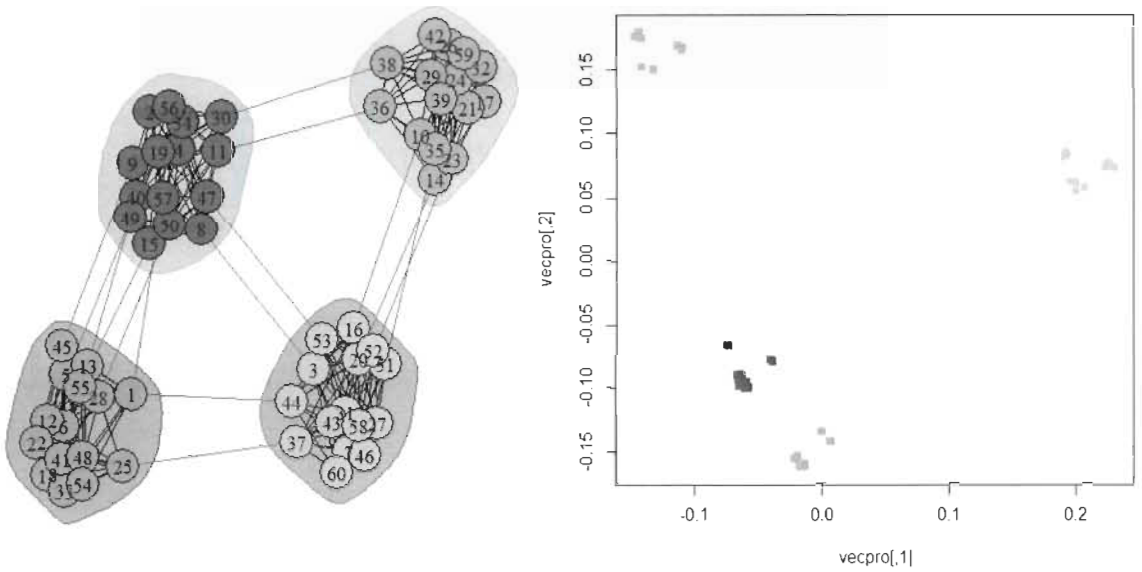


FIGURE 2.9: La projection d'un graphe aléatoire de quatre communautés dans un espace de deux dimensions engendré par les deux vecteurs propres associés aux deux plus petites valeurs propres non nulles.

2.4 Conclusion

Nous avons présenté dans ce chapitre quelques définitions de la structure communautaire. Ces dernières, se divisent en deux catégories soit les définitions classiques et

les définitions récentes. Nous avons défini quelques catégories de méthodes de détection de communautés dans les réseaux, tout en prenant compte de quelques avantages et inconvénients de certaines méthodes. Nous nous sommes basés sur la classification spectrale, en montrant ses concepts fondamentaux, ce qui va nous aider à présenter l'approche proposée dans ce mémoire, que nous allons voir dans le chapitre suivant.

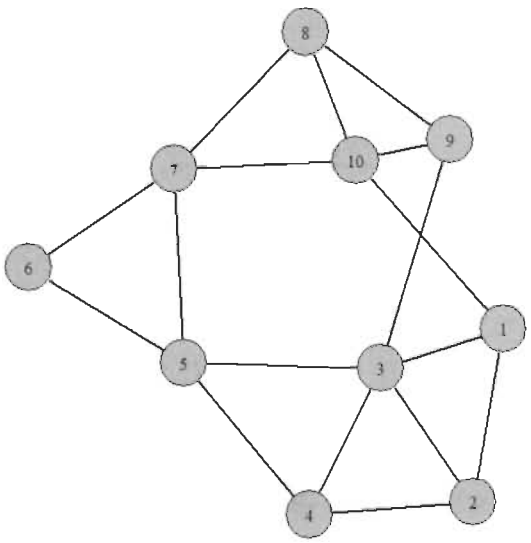
Chapitre 3

Méthode proposée

Dans ce chapitre, nous présentons les détails de la méthode de détection de communautés proposée. Il s'agit d'une classification spectrale qui utilise un processus d'agglomération conditionnelle des nœuds. L'approche proposée consiste à projeter le graphe dans un espace de deux dimensions, engendré par deux vecteurs propres spécifiques, en utilisant les informations obtenues de la projection et les informations préliminaires du graphe, nous avons créé une mesure de ressemblance entre les nœuds. Ensuite, nous appliquons le processus d'agglomération conditionnelle qui est différent de la classification hiérarchique agglomérative présentée dans la section 2.3.1. Celui-ci peut procéder plus qu'une agglomération dans une itération avant de recalculer la mesure de ressemblance entre les nœuds. Aussi, chaque agglomération doit répondre à des conditions que nous allons présenter dans ce chapitre. À chaque itération, chaque deux nœuds agglomérés forment un nouveau nœud qui est le centre des deux nœuds agglomérés. Ces nouveaux nœuds seront ajoutés au graphe et nous recalculons la mesure de ressemblance entre les nœuds pour commencer une nouvelle itération jusqu'à la fin de l'algorithme.

La méthode est applicable sur les graphes simples non orientés et non pondérés pour des raisons de simplicité. Nous allons prendre le graphe $G = (V, E)$ comme exemple pour bien clarifier chaque étape. Ce dernier est composé de 10 nœuds et de 17 arêtes.

Le graphe G est défini par sa matrice d'adjacence A (voir la figure 3.1).



$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

FIGURE 3.1: Exemple d'un graphe simple et sa matrice d'adjacence

3.1 Première étape

La première étape de cette approche est de calculer la matrice Laplacienne L à partir de la matrice d'adjacence A et de la matrice des degrés D par la formule :

$$L = D - A$$

La matrice D est une matrice diagonale tel que chaque élément de diagonal principal représente la somme des éléments de la ligne qui lui correspond dans la matrice d'adjacence. Par exemple la somme sur la première ligne de la matrice d'adjacence A est 3 qui égal à l'élément diagonal de la première ligne de la matrice D et ainsi de suite pour le reste des éléments de la matrice des degrés D :

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{pmatrix}$$

Matrice des degrés

La matrice Laplacienne L est la soustraction de la matrice d'adjacence A de la matrice des degrés D :

$$L = D - A = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

$$L = \begin{pmatrix} 3 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 5 & -1 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 4 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 4 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ -1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 4 \end{pmatrix}$$

Matrice Laplacienne

3.2 Deuxième étape

La deuxième étape de cet algorithme est de calculer les vecteurs propres de la matrice Laplacienne L. Nous allons prendre les deux vecteurs propres (X_1, X_2), associés aux deux plus petites valeurs propres (λ_1, λ_2) non nulles. Ce choix est justifié par les travaux que nous avons présentés dans la section 2.3.4.

Notre algorithme prend les deux vecteurs propres (X_1, X_2) qui forment une matrice P de taille $n \times 2$. Chaque ligne de cette matrice représente les coordonnées (x_1, x_2) d'un nœud du graphe G dans un espace en deux dimensions :

$$P = (X_1, X_2)$$

Nous avons vu dans la section 2.1.2 comment calculer les valeurs et les vecteurs propres, puisque la matrice de notre exemple est de taille 10×10 alors le calcul des valeurs propres

est plus difficile donc nous avons appliqué cette étape sur logiciel **R** qui nous a donné les résultats suivants :

$$\lambda_0 \approx 0,\ \lambda_1 \approx 1.1,\ \lambda_2 \approx 1.3$$

Les deux valeurs propres qui nous intéressent sont λ_1 et λ_2. Pour trouver le vecteur propre X_1 associé à la valeur propre λ_1 nous devons résoudre le système d'équations suivant :

$$LX_1 = \lambda_1 X_1$$

Pour trouver le vecteur propre X_2 associé à la valeur propre λ_2 nous devons résoudre le système d'équations suivant :

$$LX_2 = \lambda_2 X_2$$

Où L est la matrice Laplacienne. Les résultats donnés par logiciel R sont les suivants :

$$X_1 = \begin{pmatrix} 0.30776 \\ 0.49636 \\ 0.25811 \\ 0.37540 \\ -0.04259 \\ -0.42158 \\ -0.33529 \\ -0.33610 \\ -0.13122 \\ -0.17085 \end{pmatrix} \qquad X_2 = \begin{pmatrix} 0.16583 \\ -0.03955 \\ -0.00640 \\ -0.22680 \\ -0.34035 \\ 0.60309 \\ -0.08377 \\ 0.38855 \\ 0.41718 \\ 0.32841 \end{pmatrix}$$

Nous combinons les deux vecteurs propres X_1 et X_2 pour former la matrice P :

$$P = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} X_1 \\ \left(\begin{array}{c} 0.30776 \\ 0.49636 \\ 0.25811 \\ 0.37540 \\ -0.04259 \\ -0.42158 \\ -0.33529 \\ -0.33610 \\ -0.13122 \\ -0.17085 \end{array} \right. \end{array} \begin{array}{c} X_2 \\ \left. \begin{array}{c} 0.16583 \\ -0.03955 \\ -0.00640 \\ -0.22680 \\ -0.34035 \\ -0.60309 \\ -0.08377 \\ 0.38855 \\ 0.41718 \\ 0.32841 \end{array} \right) \end{array}$$

L'objectif de cette étape est de projeter le graphe G dans un espace vectoriel propre en deux dimensions, c'est-à-dire d'associer à chaque nœud du graphe des coordonnées cartésiennes dans un espace en deux dimensions. Après avoir obtenu la projection du graphe dans un espace métrique, nous appliquons un processus d'agglomérations des nœuds selon leurs valeurs de ressemblance que nous allons définir à l'étape suivante. Ce processus d'agglomération est différent de la classification hiérarchique agglomérative, tel qu'il peut faire plus qu'une agglomération dans une itération. Chaque agglomération produit un nouveau nœud qui est le centre des nœuds agglomérés dans l'espace métrique, et il contient les deux nœuds et l'arête qui les relie. Après quelques itérations, les communautés commencent à se former. Au début de cette étape, nous considérons chaque nœud comme une communauté qui contient un nœud interne et aucune arête interne. Nous notons N_i le nombre de nœuds à l'intérieur d'une communauté i, E_i le nombre des arêtes internes de la communauté j et E_{ij} le nombre d'arêtes entre les communautés i et j.

3.3 Troisième étape

La troisième étape consiste à calculer la mesure de ressemblance entre chaque paire de nœuds qui sont connectés par au moins une arête. Cette mesure notée S_{ij} représente la ressemblance entre les deux nœuds v_i et v_j. Elle se calcule à partir des informations préliminaires du graphe et les informations obtenues de la projection du graphe, nous construisons la matrice de dissimilarité S qui contient les éléments S_{ij}. Ces éléments sont calculés par la mesure de ressemblance S_{ij} suivante :

$$S_{ij} = \frac{d_{ij}}{e^{(NC_{ij}+E_{ij}+F_{ij})}}$$

Où

- d_{ij} est la distance euclidienne au carré entre les nœuds v_i et v_j dans l'espace métrique. Soit (x_i, y_i) et (x_j, y_j) les coordonnées associées respectivement aux nœuds v_i et v_j, la distance euclidienne au carré se calcule comme suit :

$$d_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2$$

.

- e est la fonction exponentielle.
- NC_{ij} est le nombre des nœuds voisins communs entre les nœuds v_i et v_j.
- E_{ij} est le nombre d'arêtes entre les nœuds v_i et v_j.
- F_{ij} est le rapport entre le nombre d'arêtes E_{ij} et le nombre de nœuds N_i plus le rapport entre le nombre d'arêtes E_{ij} et le nombre de nœuds N_j. Soit v_i et v_j deux nœuds (ou communautés), N_i et N_j le nombre de nœuds internes respectivement de v_i et v_j et E_{ij} le nombre d'arêtes qui les relient, F_{ij} se calcule comme suit :

$$F_{ij} = \frac{E_{ij}}{N_i} + \frac{E_{ij}}{N_j}$$

Notre mesure de ressemblance est toujours positive, aussi plus que la valeur de S_{ij} est petite plus que la similarité entre les nœuds v_i et v_j est élevée et elle dépend de

plusieurs facteurs. Dans le cas du facteur de la distance euclidienne au carré d_{ij}, plus la distance d_{ij} est petite plus la valeur de S_{ij} diminue, donc plus la similarité augmente. De plus, pour les trois facteurs NC_{ij}, E_{ij} et F_{ij}, plus ils sont élevés plus la valeur S_{ij} diminue, donc la similarité est élevée.

Prenons l'exemple de la mesure de ressemblance $S_{1,2}$, entre les nœuds v_1 et v_2 de l'exemple qui dans la figure 3.1. Nous remarquons que les nœuds v_1 et v_2 sont connectés par une arête, donc nous pouvons calculer la valeurs $S_{1,2}$. Nous avons :

- Les coordonnées du nœud v_1 dans l'espace métrique :

$$(x_1, y_1) = (0.30776, 0.16583)$$

.

- Les coordonnées du nœud v_2 dans l'espace métrique :

$$(x_2, y_2) = (0.49636, -0.03955)$$

.

- La distance euclidienne au carré $d_{1,2}$ entre les nœuds v_1 et v_2 dans l'espace métrique :

$$\begin{aligned} d_{1,2} &= (x_1 - x_2)^2 + (y_1 - y_2)^2 \\ &= (0.30776 - 0.49636)^2 + (0.16583 - (-0.03955))^2 \\ &= 0.0777 \end{aligned}$$

- Le nombre de nœuds voisins communs entre les nœuds v_1 et v_2 : $NC_{1,2}$=1, qui est le nœud v_3.
- Le nombre d'arêtes qui relient les nœuds v_1 et v_2 : $E_{1,2} = 1$.
- Le nombre de nœuds internes de nœuds v_1 : $N_1 = 1$.

- Le nombre de nœuds internes de nœuds v_2 : $N_2 = 1$.
- Le facteur $F_{1,2}$:

$$F_{1,2} = \frac{E_{1,2} =}{N_1} + \frac{E_{1,2} =}{N_1} = \frac{1}{1} + \frac{1}{1} = 2$$

La valeur de la mesure de ressemblance $S_{1,2}$ pour les deux nœuds v_1 et v_2 est :

$$S_{1,2} = \frac{d_{1,2}}{e^{(NC_{1,2}+E_{1,2}+F_{1,2})}} = \frac{0.0777}{e^{(1+1+2)}} = 0.0014$$

En appliquant cette étape sur le reste des nœuds qui sont connectés de l'exemple qui est dans la figure 2.3, nous obtenons la matrice S :

$$\mathbf{S} = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\ \begin{pmatrix} . & . & . & . & . & . & . & . & . & . \\ 0.0014 & . & . & . & . & . & . & . & . & . \\ 0.0005 & \mathbf{0.0003} & . & . & . & . & . & . & . & . \\ . & 0.0009 & 0.0004 & . & . & . & . & . & . & . \\ . & . & 0.0037 & 0.0034 & . & . & . & . & . & . \\ . & . & . & . & 0.0038 & . & . & . & . & . \\ . & . & . & . & \mathbf{0.0027} & 0.0050 & . & . & . & . \\ . & . & . & . & . & . & 0.0040 & . & . & . \\ . & . & 0.0164 & . & . & . & . & 0.0007 & . & . \\ 0.0127 & . & . & . & . & . & 0.0036 & 0.0002 & \mathbf{0.0001} & . \end{pmatrix} \end{array}$$

Les points (.) dans la matrice S représentent des valeurs indéterminées, car les nœuds ne sont pas connectés et la mesure Sij se calcule seulement quand deux nœuds v_i et v_j sont connectés par au moins une arête.

3.4 Quatrième étape

La quatrième étape consiste à agglomérer les nœuds les plus similaires deux par deux. À cet effet, il faut que les deux noeuds satisfassent quelques conditions. Nous commençons le processus d'agglomération par la paire de nœuds la plus similaire donc

la paire de nœuds (v_i, v_j) où S_{ij} est minimale. Pour réaliser la définition d'une bonne communauté, qui est «la densité de la connexion des arêtes doit être plus élevée à l'intérieur qu'à l'extérieur » (Yang et Leskovec, 2012), nous allons vérifier les deux conditions suivantes :

1. $N_i \leq N_j$ **et** $E_i \leq E_{ij}$
2. $N_j \leq N_i$ **et** $E_j \leq E_{ij}$

- **Si** la paire (v_i, v_j) satisfait une de ces conditions alors nous agglomérons les deux nœuds v_i et v_j et nous créons un nouveau nœud v_k qui contient à l'intérieur les deux nœuds agglomérés. Ses coordonnées dans l'espace métrique sont le centre des nœuds v_i et v_j, les voisins de v_k sont les voisins de v_i et v_j. Son nombre d'arêtes internes est $E_k = E_i + E_j + E_{ij}$ et son nombre de nœuds internes est $N_k = N_i + N_j$. Nous supprimons les nœuds v_i et v_j et nous passons à la prochaine paire de nœuds les plus similaires, c'est-à-dire la paire de nœuds (v_i, v_j) où S_{ij} est minimale.
- **Sinon** nous passons directement à la prochaine paire de nœuds les plus similaires donc la paire de nœuds (v_i, v_j) où S_{ij} est minimale.

En appliquant cette étape sur notre exemple précédent de la figure 3.1. Nous remarquons dans la matrice S que la paire (v_9, v_{10}) a la plus petite valeur de mesure de ressemblance $S_{9,10} = 0.0001$. Nous vérifions les deux conditions précédentes sur cette paire, commençons par la première condition, nous avons :

- Le nombre de nœuds internes du nœud v_9, $N_9 = 1$.
- Le nombre de nœuds internes du nœud v_{10}, $N_{10} = 1$.
- Le nombre d'arêtes internes de nœud v_9, $E_9 = 0$.
- Le nombre d'arêtes qui relient les nœuds v_9 et v_{10}, $E_{9,10} = 1$.

Nous appliquons la première condition sur ces données :

$$(N_9 \leq N_{10} \; et \; E_9 \leq E_{9,10}) \Rightarrow (1 \leq 1 \; et \; 0 \leq 1)$$

Nous trouvons que la paire de nœuds (v_9, v_{10}) satisfait la première condition qui est suffisant pour agglomérer les nœuds v_9 et v_{10}. En fusionnant les nœuds (v_9, v_{10}) nous construisons un nœud noté $v_{9,10}$ qui a les propriétés suivantes :

- Le nombre de nœuds internes du nœud $v_{9,10}$:

$$N_{v_{9,10}} = N_9 + N_{10} = 1 + 1 = 2$$

- Le nombre d'arêtes internes du nœud $v_{9,10}$:

$$E_{v_{9,10}} = E_9 + E_{10} + E_{9,10} = 0 + 0 + 1 = 1$$

- Les coordonnées du nœud $v_{9,10}$ dans l'espace métrique sont le centre des nœuds v_9 et v_{10} :

$$\begin{aligned}(x_{v_{9,10}}, y_{v_{9,10}}) &= (\frac{x_9 + x_{10}}{2}, \frac{y_9 + y_{10}}{2}) \\ &= \left(\frac{(-0.13122) + (-0.17085)}{2}, \frac{(0.41718) + (0.32841)}{2}\right) \\ &= (-0.15103, 0.37279)\end{aligned}$$

- Le nœud $v_{9,10}$ est connecté avec tout les voisins des nœuds v_9 et v_{10} par au moins une arête.

Nous éliminons les lignes et les colonnes associées aux nœuds v_9 et v_{10} de la matrice S, et nous continuons notre processus d'agglomération sur le reste des nœuds en utilisant les informations de la matrice S sans prendre en considération les nœuds v_9 et v_{10} et le nouveau nœud $v_{9,10}$. La matrice S devient comme suit :

$$\mathbf{S} = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\ \begin{pmatrix} . & . & . & . & . & . & . & . & . & . \\ 0.0014 & . & . & . & . & . & . & . & . & . \\ 0.0005 & \mathbf{0.0003} & . & . & . & . & . & . & . & . \\ . & 0.0009 & 0.0004 & . & . & . & . & . & . & . \\ . & . & 0.0037 & 0.0034 & . & . & . & . & . & . \\ . & . & . & . & 0.0038 & . & . & . & . & . \\ . & . & . & . & \mathbf{0.0027} & 0.0050 & . & . & . & . \\ . & . & . & . & . & . & 0.0040 & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \end{pmatrix} \end{array}$$

Nous identifions les deux nœuds les plus ressemblants en prenant le minimum des S_{ij} de la matrice S. La valeur minimale de la matrice S est $S_{2,3} = 0.0003$ qui correspond à la paire (v_2, v_3). Nous appliquons les mêmes calculs que pour la paire de nœuds (v_9, v_{10}). La matrice S devient comme suit :

$$\mathbf{S} = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\ \begin{pmatrix} . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & 0.0034 & . & . & . & . & . & . \\ . & . & . & . & 0.0038 & . & . & . & . & . \\ . & . & . & . & \mathbf{0.0027} & 0.0050 & . & . & . & . \\ . & . & . & . & . & . & 0.0040 & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \end{pmatrix} \end{array}$$

De la matrice S, la prochaine paire à fusionner est (v_5, v_7, par une valeur $S_{5,7} = 0.0027$. En agglomérant les deux nœuds v_5 et v_7 la matrice S devient comme suit :

$$
\mathbf{S} = \begin{array}{c} v_i \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array}
\begin{array}{c}
\begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\
\begin{pmatrix}
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & .
\end{pmatrix}
\end{array}
$$

Toutes les valeurs de la matrice S sont indéterminées. À cette étape, nous avons aggloméré toutes les paires de nœuds qui sont possible à fusionner pour la première itération. Nous avons aggloméré trois paires de nœuds qui sont (v_9, v_{10}), (v_2, v_3) et (v_5, v_7). Nous avons construit un nouveau nœud pour chaque agglomération de deux nœuds. Nous leur avons associé les informations nécessaires telles que les coordonnées, le nombre de nœuds internes, le nombre d'arêtes internes et la connexion avec le reste du graphe. La figure 3.2 montre l'état du graphe G après la première itération.

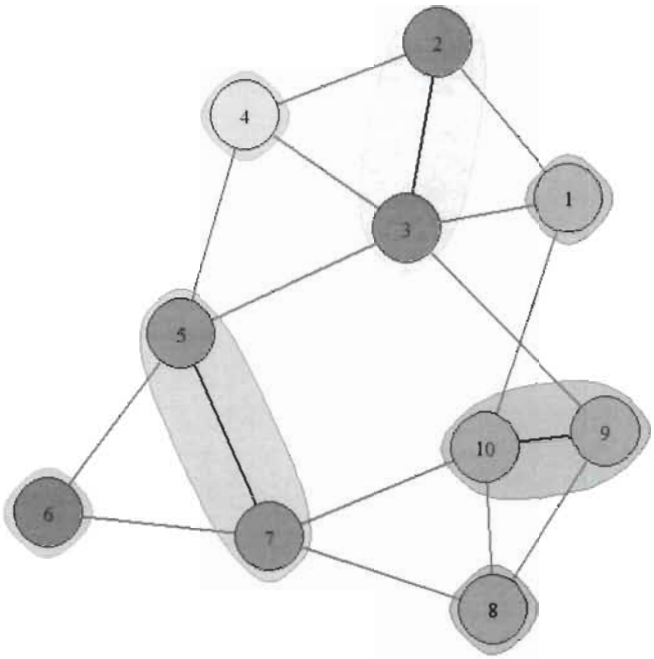


FIGURE 3.2: Résultats de la première itération de l'algorithme

Pour commencer la deuxième itération, nous mettons à jour la matrice des coordonnées P et les informations du graphe G. Ensuite, nous recalculons la nouvelle matrice S. Nous obtenons la matrice P comme suit :

$$P = \begin{array}{c} v_i \\ 1 \\ 2,3 \\ 4 \\ 5,7 \\ 6 \\ 8 \\ 9,10 \end{array} \begin{array}{c} X_1 \\ \begin{pmatrix} 0.30776 \\ 0.37724 \\ 0.37540 \\ -0.18894 \\ -0.42158 \\ -0.33610 \\ -0.15103 \end{pmatrix} \end{array} \begin{array}{c} X_2 \\ \begin{pmatrix} 0.16583 \\ -0.02297 \\ -0.22680 \\ -0.21206 \\ -0.60309 \\ 0.38855 \\ 0.37279 \end{pmatrix} \end{array}$$

En utilisant la nouvelle matrice P et les nouvelles informations du graphe G, nous calculons les mesures de ressemblances S_{ij} entre chaque paire de nœuds du graphe modifié et nous construisons la nouvelle matrice S :

$$\mathbf{S} = \begin{array}{c} \\ 1 \\ 2,3 \\ 4 \\ 5,7 \\ 6 \\ 8 \\ 9,10 \end{array} \begin{array}{c} \begin{array}{ccccccc} v_i & 1 & 2,3 & 4 & 5,7 & 6 & 8 & 9,10 \end{array} \\ \begin{pmatrix} . & . & . & . & . & . & . \\ \frac{3.69}{10^5} & . & . & . & . & . & . \\ . & \frac{3.78}{10^5} & . & . & . & . & . \\ . & \frac{8.83}{10^4} & \frac{3.54}{10^3} & . & . & . & . \\ . & . & . & \frac{5.13}{10^4} & . & . & . \\ . & . & . & \frac{4.24}{10^3} & . & . & . \\ \frac{2.81}{10^3} & \frac{1.08}{10^3} & . & \frac{8.51}{10^4} & . & \frac{3.14}{10^5} & . \end{pmatrix} \end{array}$$

En appliquant les mêmes étapes que dans la première itération nous obtenons les résultats qui sont dans la figure 3.3.

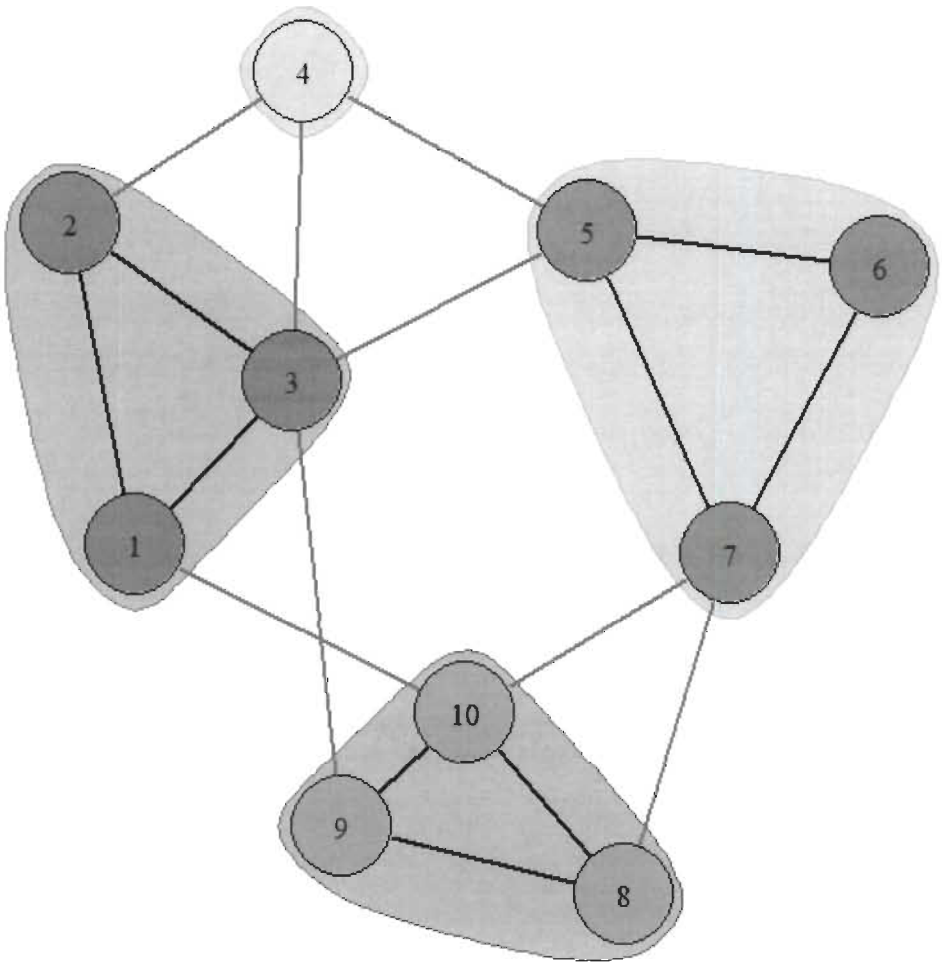


FIGURE 3.3: Résultats de la deuxième itération de l'algorithme

Nous remarquons que le nœud v_8 a été fusionné avec le nœud $v_{9,10}$, donc nous avons obtenu le nœud $v_{8,9,10}$. Pour le nœud v_6, il a été fusionné avec le nœud $v_{5,7}$ ce qui nous a donné le nœud $v_{5,7,6}$. Aussi, le nœud v_1 qui a été aggloméré avec le nœud $v_{2,3}$ ce qui a produit le nœud $v_{2,3,1}$. Passons à la prochaine itération, les résultats de la troisième itération sont dans la figure 3.4.

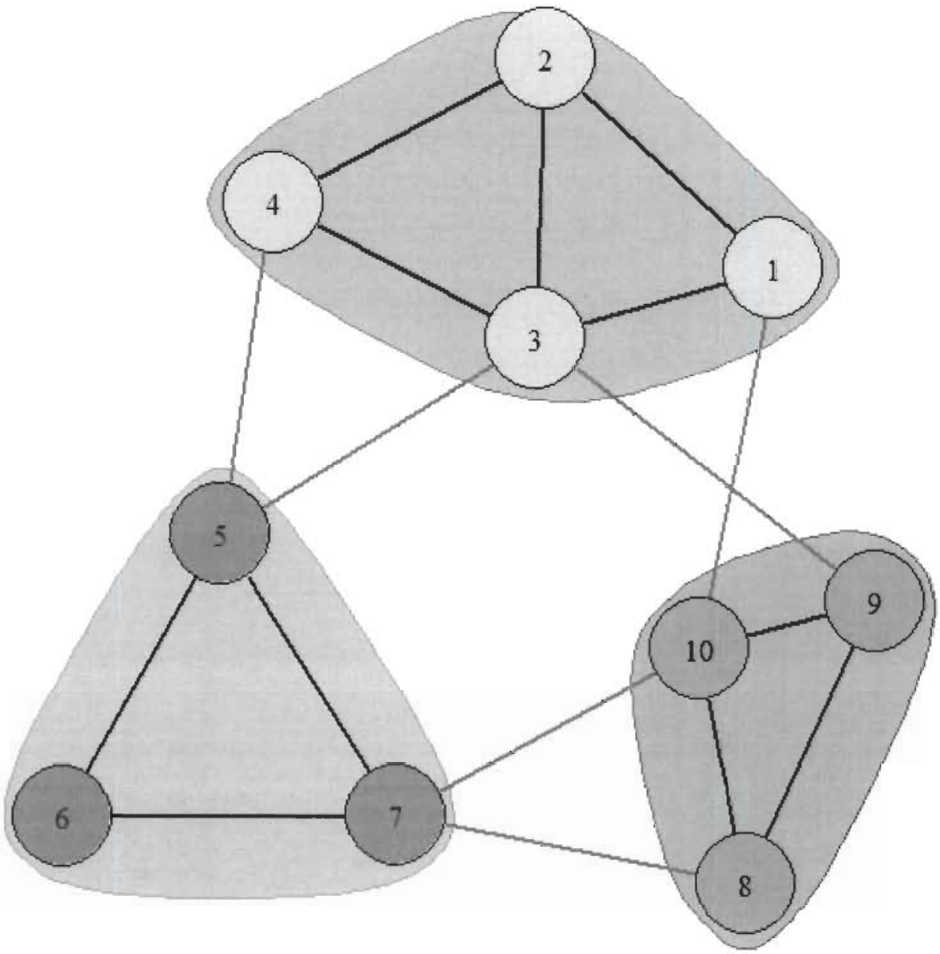


FIGURE 3.4: Résultats de la troisième itération de l'algorithme

Dans la troisième itération le nœud v_4 a été aggloméré avec le nœud $v_{2,3,1}$ ce qui nous a donné le nœud $v_{2,3,1,4}$. À cette étape, notre graphe est divisé en trois nœuds qui sont :

- Le nœud $v_{2,3,1,4}$ qui contient 4 nœuds internes et 5 arêtes internes.
- Le nœud $v_{5,7,6}$ qui contient 3 nœuds internes et 3 arêtes internes.
- Le nœud $v_{9,10,8}$ qui contient 3 nœuds internes et 3 arêtes internes.

Nous remarquons sur la figure 3.4 que le nombre d'arêtes entre chaque paire de nœuds est égal à 2, ce qui est strictement inférieur à tous les nombres d'arêtes internes de chaque nœud. D'après la définition de la bonne communauté que nous avons utilisée, aucune paire de nœuds ne peut être agglomérée dans la prochaine itération. Ce qui

signifie la fin de l'algorithme pour le graphe G, et les trois nœuds finaux obtenus sont les communautés détectées dans le graphe G.

3.5 Pseudo code

Le pseudo code 3.1 représente le résumé des différentes étapes de notre approche.

Pseudo code
Entrée : Graphe simple $G = (V, E)$.
Sortie : K communautés
1. Calculer la matrice Laplacienne $L = D - A$
2. Calculer les vecteurs propres X_1, X_2 et construire la matrice P.
3. Calculer la mesure de ressemblance S_{ij} entre chaque paire de nœuds et construire la matrice S.
4. Agglomérer les paires de nœuds qui peuvent être agglomérées, en se basant sur la matrice S et la définition de la bonne communauté, mettre à jour le graphe et la matrice P, aller à l'étape (3).
5. Si aucune paire de nœuds ne peut être agglomérée, alors l'algorithme s'arrête.

TABLE 3.1: Pseudo code de notre algorithme

3.6 Conclusion

Nous avons présenté dans ce chapitre une nouvelle approche de détection de communauté dans les réseaux. Celle-ci est basée sur les propriétés spectrales de la matrice Laplacienne associée à un graphe, ainsi que la façon dont les nœuds sont connectés. Nous avons proposé une fonction de ressemblance pour identifier les deux nœuds les

plus ressemblants. Cette fonction utilise les informations obtenues de la projection du graphe dans un espace métrique et les informations préliminaires du graphe. Nous avons utilisé un processus d'agglomération des nœuds deux à deux, en vérifiant des conditions spécifiques avant chaque fusionnement pour assurer une structure communautaire significative.

Nous avons illustré chaque étape de notre approche avec un exemple d'un graphe. Pour ce faire, nous avons montré la façon de calculer les matrices associées à ce graphe et l'obtention des coordonnées des nœuds dans un espace métrique de deux dimensions. Nous avons expliqué chaque facteur de la fonction de ressemblance en la calculant étape par étape sur un exemple de deux nœuds du graphe. Nous avons donné un aperçu sur le fonctionnement du processus d'agglomération utilisé et sa capacité d'agglomérer plus qu'une paire de nœuds dans une itération. Ainsi, le résultat final de l'approche est trois communautés qui ne peuvent pas être fusionnées à cause des conditions d'agglomération qui ont été mises.

Chapitre 4

Expérimentations

Cette section présente les résultats de notre méthode, nous allons comparer les résultats de notre méthode avec les résultats de l'algorithme de détection de communautés **Leading Eigenvector** introduit par (Newman, 2006a) que nous allons définir par la suite. Nous prenons comme données du test le benchmark de Girvan et Newman (Girvan et Newman, 2002) et le benchmark LFR (Lancichinetti *et al.*, 2008). La comparaison sera basée sur la valeur de la Modularité de Girvan et Newman (Girvan et Newman, 2004; Newman, 2006b).

4.1 Implémentation

Nous avons implémenté notre algorithme avec le logiciel **R**, nous avons utilisé le package *<igraph>* pour accéder facilement aux différentes propriétés des graphes. De plus, le package *<RSpectra>* a servi à calculer les vecteurs propres de la matrice Laplacienne L.

4.1.1 Logiciel R

R est un langage de programmation pour les calculs statistiques. Il peut être utilisé pour nettoyer, analyser et représenter graphiquement différents types de données. Il est largement utilisé par les chercheurs de diverses disciplines pour estimer et afficher les

résultats, ainsi que par les enseignants de statistiques. C'est un logiciel gratuit, ce qui en fait une option attrayante.

Les capacités de R sont étendues grâce à des packages créés par l'utilisateur, qui permettent des techniques statistiques spécialisées, des dispositifs graphiques, des capacités d'import/export, etc. Le grand nombre de packages disponibles pour R, et la facilité de les installer et de les utiliser, a été cité comme un facteur majeur dans l'adoption généralisée du langage dans la science des données. Nous avons choisi le logiciel R pour implémenter notre approche, en raison de sa facilité de manipuler les matrices, et les graphes à travers le package $<igraph>$, ainsi, les propriété spectrales des matrices en utilisant le package $<RSspectra>$.

4.1.2 Package *igraph*

Le package $<igraph>$ est une collection de différents outils pour créer et manipuler les graphes et analyser les réseaux. Il est écrit en langage C et il existe également des versions pour les logiciels Python et R. Le package est largement utilisé dans la recherche liée à la théorie des graphes, la science des réseaux et dans les domaines similaires.

Il peut être utilisé pour générer des graphes à partir de plusieurs formes de données comme la matrice d'adjacence, la matrice d'incidence, la liste des arêtes, etc. Il est capable de calculer plusieurs mesures associées aux graphes comme le plus court chemin entre deux sommets. Nous avons choisi ce package car il nous a permis de construire plus facilement des graphes à partir des données géuérées par les benchmarks, et il nous a permis d'obtenir les informations préliminaires du graphe.

4.1.3 Package *RSpectra*

La décomposition des valeurs propres est une technique couramment utilisée dans de nombreux problèmes statistiques. Dans logiciel R, la méthode standard pour calculer les valeurs propres est la fonction *eigen*(). Cependant, lorsque la matrice devient grande, *eigen*() peut prendre beaucoup de temps car elle n' a pas d'option pour limiter le nombre de valeurs propres à calculer. Alors que dans notre approche, nous n'avons besoin de calculer que les deux plus petites valeurs propres non nulles, pour calculer leurs vecteurs propres associés. Pour cette raison nous avons utilisé le package <*RSpectra*>. Il est capable de résoudre des problèmes de valeurs propres à grande échelle et de calculer seulement les valeurs propres que nous demandons, ce qui réduit le temps de calcul.

4.2 Benchmarks

4.2.1 Benchmark de Girvan et Newman (GN)

Le benchmark de Girvan et Newman (Girvan et Newman, 2002) est un algorithme de création de graphes inspiré du modèle de communauté qui est défini comme suit : les nœuds d'une même communauté ont une probabilité plus élevée de former des arêtes avec leurs partenaires qu'avec les autres nœuds. Ses caractéristiques sont les suivantes :

- Nombre de communautés du graphe $C = 4$
- Taille de chaque communauté $N = 32$
- Nombre total de nœuds $n = 128$
- Le degré total moyen de chaque nœud $k = 16$.

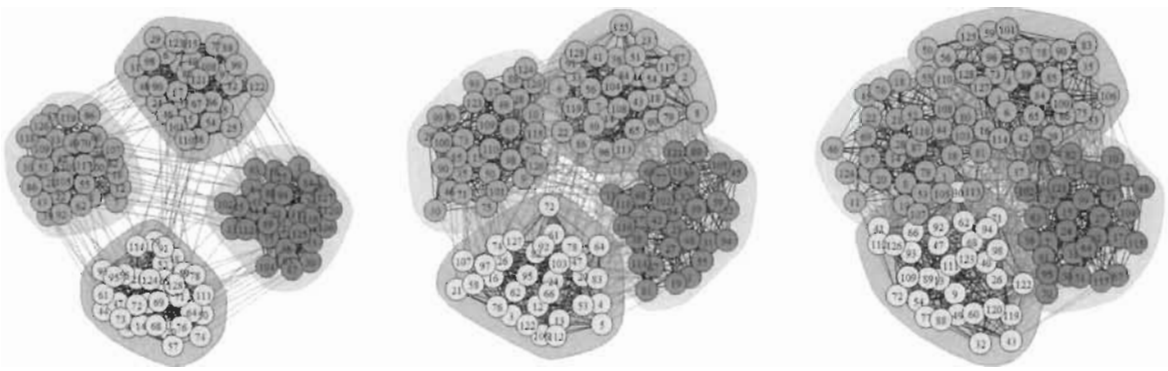
4.2.2 Benchmark de Lancichinetti-Fortunato-Radicchi (LFR)

Le benchmark LFR (Lancichinetti-Fortunato-Radicchi) (Lancichinetti *et al.*, 2008) est un algorithme de création de graphes aléatoires dont les caractéristiques se veulent aussi proches que des graphes complexes car il tient compte de l'hétérogénéité de la distribution des degrés des nœuds et des tailles de communautés. Il fournit en même temps

la solution, c'est-à-dire le partitionnement en communautés. L'algorithme demande un certain nombre de paramètres tels que :

- n : le nombre de nœuds
- k : le degré interne moyen des nœuds.
- $maxk$: le degré maximum des nœuds.
- μ : le paramètre de mixage.
- $minc$: taille minimum des communautés.
- $maxc$: taille maximum des communautés.

Il existe d'autres paramètres que nous pouvons ajouter, mais dans notre cas ils sont inutiles. Le paramètre le plus important à varier est le paramètre de mixage μ. Ce dernier représente la moyenne (pour tous les nœuds) du rapport entre le nombre de liens propres à un nœud se trouvant à l'extérieur de sa communauté et le nombre total de liens de ce nœud. Lorsque cette valeur est faible, les communautés au sein du graphe sont bien distinguées et peuvent être vues même visuellement, mais plus il augmente plus leur identification sera difficile. Par exemple, nous pouvons générer le benchmark GN en utilisant l'algorithme de benchmark LFR, tout en posant $n = 128$, $k = 16$, $maxk = 16$, $\mu = 0.1$, $minc = 32$, $maxc = 32$. En variant la valeur de μ, la distribution des arêtes change (voir la figure 4.1).



$\mu = 0.1$ $\mu = 0.2$ $\mu = 0.3$

FIGURE 4.1: Le benchmark GN généré par l'algorithme de benchmark LFR

4.3 L'algorithme Leading Eigenvector (LE)

L'algorithme **Leading Eigenvector(LE)** a été proposé par Newman (Newman, 2006a). Ce dernier est basé sur l'optimisation spectrale de la modularité en utilisant les vecteurs propres de la matrice de modularité. En calculant le vecteur propre principal de la matrice de modularité, le graphe se divise à chaque étape en deux parties de telle sorte que cette division améliore la modularité. Il s'arrête une fois que la valeur de la contribution de modularité devient négative. Nous avons choisi de comparer nos résultats avec les résultats de cet algorithme parce que c'est un algorithme spectral, c'est-à-dire, il est basé sur les propriété spectrale du graphe. Ainsi, cet algorithme est l'un des algorithmes de détection de communautés les plus populaires, aussi, il est prédéfini dans le package $<igraph>$.

4.4 Test sur le benchmark GN généré par le benchmark LFR

Nous commençons par appliquer notre algorithme noté « PM » et l'algorithme *Leading Eigenvector* noté « LE », sur le benchmark *GN* généré par l'algorithme *LFR*. Nous calculons la *modularité* Q pour chaque graphe généré et nous comparons les résultats de notre algorithme, ceux de l'algorithme *Leading Eigenvector* et ceux de la solution obtenue au moment de la génération du graphe par le benchmark *LFR*, nous notons cette solution par « RC ». Nous générons le même graphe à chaque fois en variant le paramètre de mixage μ. Nous arrêtons quand la *modularité* Q prend une valeur inférieur à 0.3 pour la solution « RC » car, comme nous avons vu dans la section 2.1.3, si la modularité Q est inférieur à 0.3 alors le graphe n'a pas une structure communautaire significative. Les résultats de ce test sont dans le tableau 4.1 et la figure 4.2.

n=128	Modularité Q		
	PM	LE	RC
$\mu = 0.05$	0.7	0.7	0.7
$\mu = 0.1$	0.65	0.65	0.65
$\mu = 0.15$	0.6	0.58	0.6
$\mu = 0.2$	0.54	0.51	0.54
$\mu = 0.25$	0.5	0.45	0.5
$\mu = 0.3$	0.43	0.41	0.45
$\mu = 0.35$	0.36	0.35	0.39
$\mu = 0.4$	0.29	0.23	0.34
$\mu = 0.45$	0.26	0.22	0.29

TABLE 4.1: Résultats de la modularité pour le benchmark GN, en variant le paramètre de mixage μ.

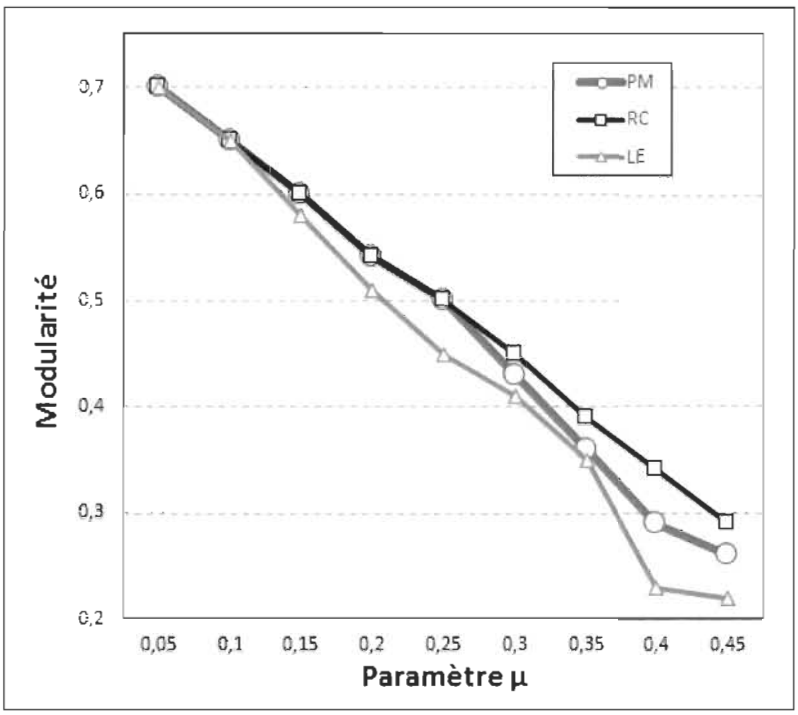


FIGURE 4.2: Variation de la modularité en fonction du paramètre de mixage μ pour le benchmark GN de 128 nœuds.

Nous remarquons que pour les deux premières valeurs $\mu = 0.05$ et $\mu = 0.1$, notre algorithme « PM » et l'algorithme « LE » ont donné les mêmes résultats que la solution « RC » générée par le benchmark LFR. Pour les trois prochaines valeurs $\mu = 0.15$, $\mu = 0.2$ et $\mu = 0.25$, notre algorithme « PM » a donné les mêmes résultats que la solution « RC », cependant les résultats donnés par l'algorithme « LE » sont moins bons que les résultats donnés par notre algorithme « PM » et la solution « RC ». Pour le reste des valeurs de $\mu = 0.3$ jusqu'à $\mu = 0.45$, nous remarquons que les résultats donnés par notre algorithme « PM » sont moins bons par rapport aux résultats de la solution « RC », mais sont meilleurs que les résultats donnés par l'algorithme « LE ».

4.5 Test sur le benchmark LFR

Dans ce test, nous générerons plusieurs graphes en utilisant le benchmark LFR. À chaque fois, nous varions le paramètre de mixage μ. Nous calculons la modularité et nous comparons les résultats des différents algorithmes. Les caractéristiques des graphes générés sont dans le tableau 4.2 :

Paramètre	Valeur
Nombre de nœuds n	de 1000 jusqu'à 50000
Le degré moyen de nœuds k	20
Le degré maximum de nœuds $maxk$	$0.02 \times n$
Taille minimale des communautés $minc$	20
Taille maximale des communautés $maxc$	$0.02 \times n$
Paramètre de mixage μ	de 0.05 jusqu'à 0.7

TABLE 4.2: Paramètres de benchmark LFR

Commençons par générer un graphe de 1000 nœuds, avec un degré moyen de nœuds de $k = 20$ et un degré maximal de nœuds de $maxk = 20$. La taille minimale des communautés à $minc = 20$ et la taille maximale des communautés à $maxc = 20$. Le paramètre de mixage μ varie de 0.05 jusqu'à 0.7. Les résultats sont dans le tableau 4.3 et la figure 4.3.

n=1000	Modularité Q		
	PM	LE	RC
$\mu = 0.05$	0.93	0.88	0.93
$\mu = 0.1$	0.88	0.8	0.88
$\mu = 0.15$	0.83	0.74	0.83
$\mu = 0.2$	0.78	0.63	0.78
$\mu = 0.25$	0.73	0.58	0.73
$\mu = 0.3$	0.65	0.55	0.68
$\mu = 0.35$	0.61	0.49	0.63
$\mu = 0.4$	0.55	0.41	0.58
$\mu = 0.45$	0.5	0.39	0.53
$\mu = 0.5$	0.43	0.34	0.48
$\mu = 0.55$	0.38	0.29	0.43
$\mu = 0.6$	0.31	0.24	0.38
$\mu = 0.65$	0.24	0.21	0.33
$\mu = 0.7$	0.22	0.2	0.28

TABLE 4.3: Résultats de la modularité pour le graphe de 10000 nœuds, en variant le paramètre de mixage μ.

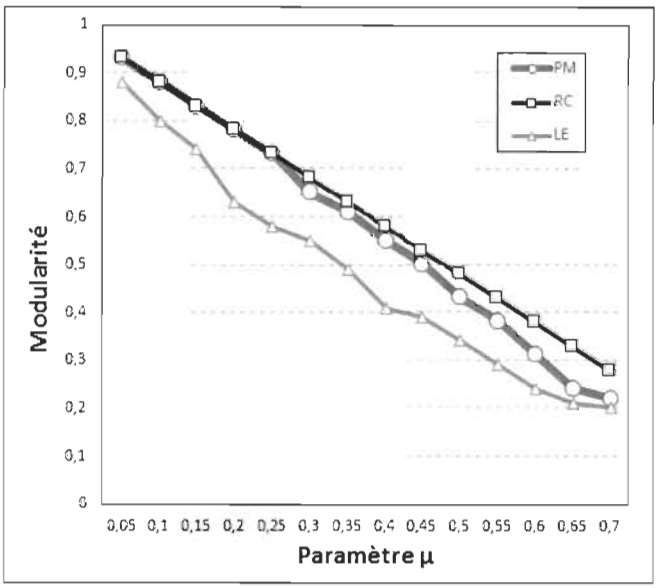


FIGURE 4.3: Variation de la modularité en fonction du paramètre de mixage μ pour le graphe de 1000 nœuds.

Nous observons que pour $\mu = 0.05$ jusqu'à $\mu = 0.25$ notre algorithme « PM » a donné les mêmes résultats que la solution « RC », ensuite nous constatons que les résultats de « PM » commencent à s'éloigner petit à petit des résultats de « RC » avec l'augmentation de la valeur de μ. Cependant, les résultats de l'algorithme « LE » sont moins bons par rapport aux résultats de « PM » et « RC » pour toutes les valeurs de μ.

Pour le prochain graphe à générer, nous augmentons le nombre de nœuds à 10000, le degré maximal de nœuds à $maxk = 200$ et la taille maximale des communautés à $maxc = 200$. Nous gardons le reste des paramètres comme le graphe précédent. Les résultats sont dans le tableau 4.4 et la figure 4.4.

n=10000	Modularité Q		
	PM	LE	RC
$\mu = 0.05$	0.85	0.47	0.93
$\mu = 0.1$	0.74	0.34	0.88
$\mu = 0.15$	0.67	0.28	0.83
$\mu = 0.2$	0.61	0.22	0.78
$\mu = 0.25$	0.55	0.21	0.73
$\mu = 0.3$	0.51	0.21	0.68
$\mu = 0.35$	0.47	0.16	0.63
$\mu = 0.4$	0.42	0.18	0.58
$\mu = 0.45$	0.38	0.16	0.53
$\mu = 0.5$	0.35	0.15	0.48
$\mu = 0.55$	0.32	0.14	0.43
$\mu = 0.6$	0.27	0.13	0.38
$\mu = 0.65$	0.23	0.13	0.33
$\mu = 0.7$	0.19	0.12	0.28

TABLE 4.4: Résultats de la modularité pour le graphe de 10000 nœuds, en variant le paramètre de mixage μ.

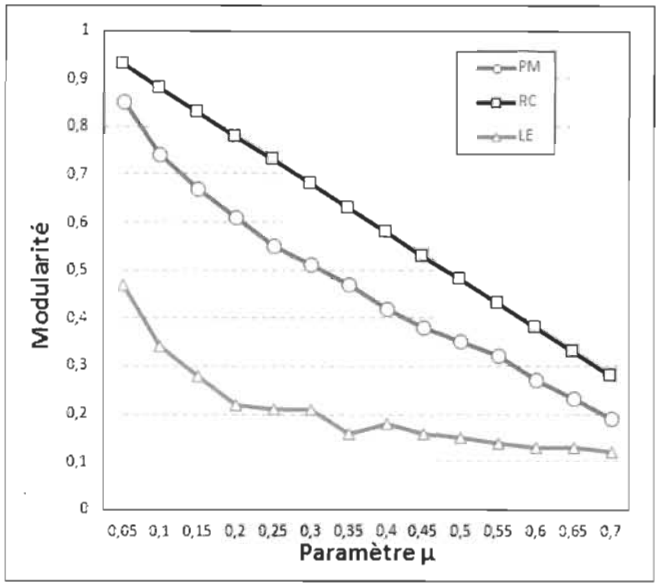


FIGURE 4.4: Variation de la modularité en fonction du paramètre de mixage μ pour le graphe de 10000 nœuds.

Nous remarquons que les résultats de notre algorithme « PM » sont moins bons par rapport aux résultats de la solution « RC » pour toutes les valeurs, mais les résultats de « PM » restent proches des résultats de « RC ». Par contre, les résultats de l'algorithme « LE » sont beaucoup moins bons par rapport aux résultats de « PM » et « RC ».

Le graphe suivant est de 25000 nœuds, avec un degré moyen de nœuds à $k = 20$ et un degré maximal de nœuds à $maxk = 500$. La taille minimale des communautés est de $minc = 20$ et la taille maximale des communautés est de $maxc = 500$. Le paramètre de mixage μ varie de 0.05 jusqu'à 0.7. Les résultats sont dans le tableau 4.5 et la figure 4.5.

n=25000	Modularité Q		
	PM	LE	RC
$\mu = 0.05$	0.83	0.43	0.93
$\mu = 0.1$	0.73	0.24	0.88
$\mu = 0.15$	0.66	0.29	0.83
$\mu = 0.2$	0.59	0.21	0.78
$\mu = 0.25$	0.54	0.14	0.73
$\mu = 0.3$	0.48	0.17	0.68
$\mu = 0.35$	0.45	0.16	0.63
$\mu = 0.4$	0.41	0.12	0.58
$\mu = 0.45$	0.38	0.12	0.53
$\mu = 0.5$	0.33	0.12	0.48
$\mu = 0.55$	0.3	0.12	0.43
$\mu = 0.6$	0.27	0.11	0.38
$\mu = 0.65$	0.23	0.11	0.33
$\mu = 0.7$	0.19	0.11	0.28

TABLE 4.5: Résultats de la modularité pour le graphe de 25000 nœuds, en variant le paramètre de mixage μ.

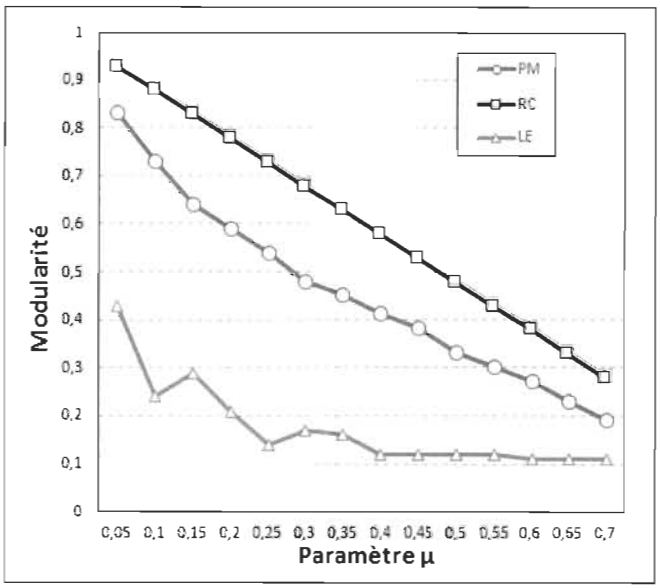


FIGURE 4.5: Variation de la modularité en fonction du paramètre de mixage μ pour le graphe de 25000 nœuds.

Pour ce graphe de 25000 nœuds, nous notons que la différence entre les résultats de notre « PM » et les résultats de la solution « RC » est presque la même que pour le graphe précédent de 10000 nœuds. Mais les résultats de l'algorithme « LE » sont moins efficaces et nous observons beaucoup plus de variations qui ne sont pas monotone au niveau de la valeur de la modularité par rapport à l'augmentation du paramètre de mixage μ.

Le dernier graphe est de 50000 nœuds, avec un degré moyen de nœuds de $k = 20$ et un degré maximal de nœuds de $maxk = 1000$. La taille minimale des communautés est de $minc = 20$ et la taille maximale des communautés est de $maxc = 1000$. Le paramètre de mixage μ varie de 0.05 jusqu'à 0.7. Les résultats sont dans le tableau 4.6 et la figure 4.6.

n=50000	Modularité Q		
	PM	LE	RC
$\mu = 0.05$	0.85	0.29	0.93
$\mu = 0.1$	0.73	0.31	0.88
$\mu = 0.15$	0.63	0.18	0.83
$\mu = 0.2$	0.52	0.15	0.78
$\mu = 0.25$	0.49	0.16	0.73
$\mu = 0.3$	0.45	0.11	0.68
$\mu = 0.35$	0.44	0.14	0.63
$\mu = 0.4$	0.41	0.13	0.58
$\mu = 0.45$	0.32	0.13	0.53
$\mu = 0.5$	0.31	0.12	0.48
$\mu = 0.55$	0.28	0.12	0.43
$\mu = 0.6$	0.24	0.12	0.38
$\mu = 0.65$	0.21	0.11	0.33
$\mu = 0.7$	0.19	0.09	0.28

TABLE 4.6: Résultats de la modularité pour le graphe de 50000 nœuds, en variant le paramètre de mixage μ.

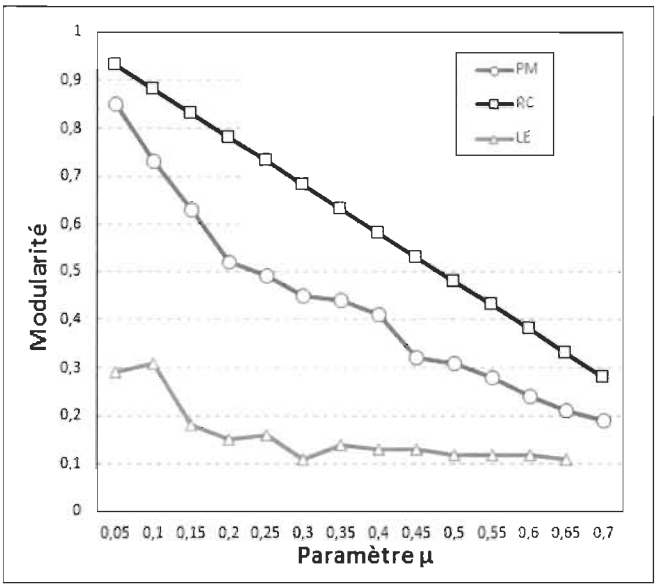


FIGURE 4.6: Variation de la modularité en fonction du paramètre de mixage μ pour le graphe de 50000 nœuds.

Pour les résultats du dernier graphe de 50000 nœuds, nous remarquons que les résultats de notre algorithme « PM » sont meilleures que les résultats de l'algorithme « LE », mais ils n'atteignent pas les résultats de la solution « RC ».

4.6 Conclusion

Dans ce chapitre, nous avons présenté les outils utilisés pour développer notre approche. Aussi, les benchmarks utilisés pour générer les données de tests, et l'algorithme *Leading Eigenvector* (LE) avec lequel nous avons comparé ses résultats avec les résultats de notre approche.

Ce chapitre a présenté les résultats de notre approche sur le benchmark de Girvan et Newman (GN), généré par l'algorithme du benchmark LFR en variant le paramètre de

mixage μ. En utilisant la modularité comme une mesure de qualité de nos résultats, nous avons vu que notre approche a donné de meilleurs résultats dans la majorité des cas par rapport aux résultats de l'algorithme LE. De plus, nous avons testé plusieurs graphes générés par le benchmark LFR, qui ont des tailles différentes, en termes du nombre de nœuds, avec des valeurs différentes du paramètre de mixage μ. Notre approche a donné les mêmes résultats que les résultats de l'algorithme LE dans quelques cas, mais dans la majorité des cas notre approche est meilleure que l'algorithme LE.

Chapitre 5

Conclusion générale et perspectives

5.1 Conclusion générale

La théorie des graphes est un outil qui peut représenter plusieurs systèmes complexes. Dans ce mémoire, nous avons introduit une nouvelle approche de détection des communautés dans les réseaux, en nous basant sur la classification spectrale et un processus d'agglomération des nœuds.

Nous avons montré l'utilité des graphes et leur rôle de pouvoir représenter plusieurs problèmes en lien avec divers sujets. Ainsi que la façon dont l'analyse des graphes peut conduire à résoudre beaucoup de ces problèmes. Nous avons vu que le problème de détection de communauté peut donner un aperçu de la structure du réseau et la façon dont il est organisé. En outre, nous avons représenté quelques définitions de la structure communautaire qui ont été proposées dans la littérature, en plus de quelques méthodes de détection de communauté les plus populaires. Nous nous sommes concentrés sur la classification spectrale, et sur la projection d'un graphe dans un espace métrique, en étudiant différentes matrices que nous pouvons lui associer.

Nous avons détaillé les différentes étapes de l'approche proposée, qui est basée sur la théorie spectrale des graphes, un processus d'agglomération des nœuds et la définition

de la bonne communauté. Cette approche, consiste à projeter le graphe dans un espace métrique de deux dimensions, ensuite, en utilisant les informations obtenues de cette projection et les informations préliminaires du graphe, nous avons créé une fonction de ressemblance entre les nœuds, afin d'agglomérer les nœuds les plus similaires, en utilisant un processus d'agglomération qui peut faire plusieurs agglomérations dans une itération. Nous avons appliqué une définition de la bonne communauté en créant des conditions pour limiter les agglomérations qui n'améliorent pas la qualité du partitionnement. Nous avons évalué les résultats de notre algorithme en les comparant avec les résultats d'un autre algorithme et les solutions données par les benchmarks utilisés.

Nous concluons que notre approche est une contribution aux travaux précédemment effectués pour résoudre le problème de détection de communauté, et plus spécifiquement pour les méthodes spectrales. Les résultats que nous avons obtenus prouvent l'efficacité de notre approche en termes de la qualité du partitionnement qui a été mesurée par la modularité.

5.2 Perspectives

Notre travail pourrait considérer ouvert pour différentes améliorations afin de rendre l'approche proposée plus compétitive. Par exemple, notre approche projette le graphe dans un espace métrique de deux dimensions, ce qui n'est pas toujours le meilleur choix. Nous pourrions ajouter une étape qui choisit le meilleur nombre de dimensions. Ce changement aurait un impact sur la complexité du calcul, mais il améliorerait la qualité du partitionnement. Aussi, nous pourrions faire des changements dans la mesure de ressemblance tel que l'ajout de d'autres facteurs pouvant être à caractère stochastique. De plus, la définition utilisée dans cette approche pourrait être remplacée par une autre définition plus moderne basée sur les probabilités des connexions entre les nœuds.

Bibliographie

AHN, Y.-Y., BAGROW, J. P. et LEHMANN, S. (2010). Link communities reveal multiscale complexity in networks. *nature*, 466:761.

BAUMES, J., GOLDBERG, M., KRISHNAMOORTHY, M., MAGDON-ISMAIL, M. et PRESTON, N. (2005). Finding communities by clustering a graph into overlapping subgraphs. *IADIS International Conference on Applied Computing*, 2:22–25.

BRANDES, U., DELLING, D., GAERTLER, M., GORKE, R., HOEFER, M., NIKOLOSKI, Z. et WAGNER, D. (2008). On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172 – 188.

BUFFON (1749). Histoire naturelle. 1:21.

CLAUSET, A. (2005). Finding local community structure in networks. *PHYSICAL REVIEW E*, 72(2).

DING, Z., ZHANG, X., SUN, D. et LUO, B. (2016). Overlapping community detection based on network decomposition. *Scientific Reports*, 6(24115).

DONETTI, L. et MUNOZ, M. A. (2004). Detecting network communities : a new systematic and efficient algorithm. *Journal of Statistical Mechanics : Theory and Experiment*, 2004.

E.R.Barnes et A.J.Hoffman (1981). On bounds for eigenvalues of real symmetric matrices. *Linear Algebra and its Applications*, 40:217–223.

Evans, T. et Lambiotte, R. (2009). Line graphs, link partitions and overlapping communities. *PHYSICAL REVIEW E*, 80(016105).

Fiedler, M. (1973). Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305.

Fiedler, M. (1975). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633.

Fortunato, S. et Hric, D. (2016). Community detection in networks : A user guide. *Physics Reports*, 659:1–44.

Girvan, M. et Newman, M. (2002). Community structure in social and biological networks,". *Proceedings of the National Academy of Sciences*, page 821–826.

Girvan, M. et Newman, M. (2004). Finding and evaluating community structure in networks. *Statistical Mechanics*, 69(026113).

Goder, A. et Filkov, V. (2008). Consensus clustering algorithms : comparison and refinement. *Proceedings of the Meeting on Algorithm Engineering and Expermiments*, pages 109–117.

Govaert, G. (2003). *Analyse des donnees (traite ic2, serie traitement du signal et de l'image)*.

Hall, K. M. (1970). An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229.

Hu, Y., Chen, H., Zhang, P., Li, M., Di, Z. et Fan, Y. (2008). Comparative definition of community and corresponding identifying algorithm. *PHYSICAL REVIEW E*, 78(026121).

Huang, J., Sun, H., Liu, Y., Song, Q. et Weninger, T. (2011). Towards online multiresolution community detection in large-scale networks. *PLoS ONE*, 6(8).

Iman, H., Effat, S. E. et Ali, A. (2014). Quantitative analysis of intracellular communication and signaling errors in signaling networks. *BMC Systems Biology*, 8:89.

Lancichinetti, A. et Fortunato, S. (2009). Community detection algorithms : A comparative analysis. *PHYSICAL REVIEW E*, 80(5).

Lancichinetti, A., Fortunato, S. et Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review*, 78(4).

LeBlanc, L. J., Morlok, E. K. et Pierskalla, W. P. (1975). An effcient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research*, 9:309–318.

Liben-Nowell, D., Novak, J., Kumar, R., Raghavan, P. et Tomkins, A. (2005). Geographic routing in social networks. *Proceedings of the National Academy of Sciences*, 102(33)(1623).

Luce, R. D. et Perry, A. D. (1949). A method of matrix analysis of group structure. *Psychometrika*, 14:95–116.

Luxburg, U. v. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Berkeley Symposium on Mathematical Statistics and Probability*, 1:281–297.

Murtagh, F. et Contreras, P. (2011). Methods of hierarchical clustering. *Information Retrieval*.

Nascimento, M. C. et Carvalho, A. C. d. (2011). Spectral methods for graph clustering – a survey. *European Journal of Operational Research*, 211(2):221–231.

Newman, M. E. J. (2006a). Finding community structure in networks using the eigenvectors of matrices. *Physical Review*, 74(3).

Newman, M. E. J. (2006b). Modularity and community structure in networks. *PNAS*, 103(23):8577–8582.

Ng, W., Liew, F., Ang, L. et Wong, K. (2001). Potential of mealworm (tenebrio molitor) as an alternative protein source in practical diets for african catfish, clarias gariepinus. *Aquaculture Research*, 32(1):273–280.

Otte, E. et Rousseau, R. (2002). Social network analysis : a powerful strategy, also for the information sciences. *Journal of Information Science*, 28(6):441–453.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. et Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, 101(2658).

Shi, J. et Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888 – 905.

Streh, A. et Ghosh, J. (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3(3):583–617.

Strogatz, S. H. (2001). Exploring complex networks. *nature*, 410(6825):268–276.

Topchy, A., Jain, A. K. et Punch, W. F. (2006). Clustering ensembles : Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1866–81.

YANG, J. et LESKOVEC, J. (2012). Defining and evaluating network communities based on ground-truth. *2012 IEEE 12th International Conference on Data Mining*, page 745–754.