République Algérienne Démocratique et populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté de Génie Electrique et d'Informatique
Département d'Informatique

THESE DE DOCTORAT LMD

DISCIPLINE : Informatique

Option : Intelligence Artificielle et Systèmes d'Information

Présentée par :

Fariza Bouhatem

Sujet

Approche d'optimisation pour le suivi de l'évolution de la structure communautaire des réseaux dynamiques

Devant le jury d'examen composé de :

-Mr Ahmed Ouamer Rachid;	Professeur,	UMMTO	Président
-Mr Aït El Hadj Ali;	MCA,	UMMTO	Rapporteur
-Mr Hamadouche Djamal;	Professeur,	UMMTO	Examinateur
-Mr Atif Karim;	MCA,	USTHB	Examinateur
-Mr Kechid Samir;	Professeur,	USTHB	Examinateur

Soutenue le 01/07/2020

Remerciements

Gloire et Louange à Dieu, le tout Puissant, de m'avoir donné courage et persévérance pour finaliser cette thèse.

Je tiens à remercier les membres de jury d'avoir accepté d'évaluer mon travail :

-Mr Rachid Ahmed Ouamer, Professeur au département d'informatique (FGEI, UMMTO), d'avoir accepté d'être président de ce jury.

-Mr Hamadouche Djamal, Professeur à la faculté des sciences de l'UMMTO d'avoir accepté d'être examinateur de ce jury.

-Mr Atif Karim, Maitre de conférences (USTHB) d'avoir répondu favorablement pour évaluer ce travail de recherche.

-Mr Kechid Samir, Professeur (USTHB) d'avoir accepté gentiment d'être membre de ce jury.

Je voudrais adresser mes vifs remerciements à mon directeur de thèse Mr Aït El Hadj Ali, Maître de Conférences (UMMTO), pour toute l'aide qu'il m'a fournie tout au long de ces trois années. Je le remercie profondément pour ses encouragements continus et sa disponibilité.

Je remercie également Mme Ait El Hadj-Souam Fatiha, Maitre de conférences à l'université de Tizi Ouzou (UMMTO), pour son investissement dans ce travail. Je tiens à la remercier très sincèrement pour ses critiques, sa disponibilité, son encouragement, et de m'avoir aidée tout au long de cette thèse.

Enfin, je tiens à remercier tous les membres de ma famille et en particulier ma chère Mère, mes adorables sœurs à leur encouragement, leur soutien et leur compréhension durant ces trois années de travail.

Résumé

L'évolution rapide des réseaux sociaux au cours des dernières années a retenu l'attention de plusieurs chercheurs en quête de solutions adéquates pour la gestion de ces réseaux. À cet effet, plusieurs algorithmes efficaces dédiés au suivi de la structure communautaire et la détection de communautés ont été proposés. Ces algorithmes ont été conçus pour les réseaux dynamiques évoluant par l'ajout et / ou la suppression de nœuds et d'arêtes. Il existe cependant, également des réseaux évoluant uniquement par l'ajout de nœud et d'arêtes (liens), appelés *réseaux incrémentiels.* Ces derniers sont un cas particulier des réseaux dynamiques qui augmentent considérablement en taille.

Dans cette thèse, nous présentons une nouvelle approche pour le suivi de l'évolution de la structure communautaire dans les réseaux incrémentiels. L'approche en question est basée sur la densité du réseau et la double optimisation de celle-ci. Le premier niveau d'optimisation consiste en l'intégration d'un nouveau nœud avec ses liens à la communauté appropriée maximisant la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées. Le second niveau d'optimisation vise à améliorer davantage le score de la densité du réseau par des opérations sur les communautés infectées. Cette double optimisation permet de réduire le problème de limite de résolution dont souffre la majorité des algorithmes d'optimisation. Rappelons que ce problème de limite de la résolution est induit par l'optimisation de la modularité. L'algorithme présenté est incrémental dans le sens où il utilise la structure communautaire précédente pour identifier la structure courante. Pour la validation, nous avons effectué des tests sur des réseaux sociaux dynamiques du monde réel. Nos résultats expérimentaux sont comparés aux résultats obtenus par des algorithmes dédiés aux réseaux statiques ainsi que des résultats obtenus avec des algorithmes conçus pour des réseaux dynamiques. Ces tests montrent que notre algorithme donne de bonnes structures communautaires avec une complexité relativement faible.

Mots clés : Structure communautaire, évolution, réseaux dynamiques, optimisation, modularité, graphe.

Abstract

The rapid evolution of social networks in recent years has attracted the attention of several researchers to find adequate solutions for the management of these networks. For this purpose, several efficient algorithms dedicated to the tracking of the community structure and to the detection of community have been proposed. These algorithms have been designed for dynamic networks evolving by adding and / or removing nodes and edges. However, there are also networks evolving only by adding nodes and edges (links), called *incremental networks*. The latter are a special case of dynamic networks which increase considerably in size.

In this thesis, we present a new approach for tracking the evolution of the community structure in incremental networks. The approach in question is based on the density of the network and the double optimization of the latter. The first level of optimization consists of integrating a new node with its links to the appropriate community, maximizing the sum of the differences between the internal density and the external density of all infected communities. The second level of optimization aims to further improve the network density score through operations on infected communities. This double optimization reduces the resolution limit problem from which most optimization algorithms suffer. Recall that this problem of resolution limit is induced by the optimization of modularity. The algorithm presented is incremental in the sense that it uses the previous community structure to identify the current one. For validation, we performed tests on dynamic social networks in the real world. Our experimental results are compared to the results obtained by algorithms dedicated to static networks as well as to results obtained with algorithms designed for dynamic networks. These tests show that our algorithm gives good community structures with relatively low complexity.

Keywords: Community structure, evolution, dynamic networks, optimization, modularity, graph.

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Introduction générale

Contexte de travail

Dans plusieurs domaines de recherche, les réseaux sont modélisés par des graphes afin de faciliter l'étude et la compréhension de leur structure. Dans les réseaux sociaux, par exemple, les nœuds représentent des individus, tandis que les liens représentent les relations et les interactions sociales entre ces individus. De même, dans les réseaux de citations, les nœuds représentent les papiers scientifiques publiés dans un journal et les liens indiquent les citations entre ces papiers. Ces derniers, quand ils partagent des caractéristiques communes, sont regroupés dans des zones dont la densité de connexions internes est plus forte que la densité de connexions vers l'extérieur. Ces zones sont appelées communautés et leur détection consiste à décomposer le réseau en un certain nombre de communautés. En réalité, dans les réseaux sociaux comme Facebook ou LinkedIn, les individus et les relations sociales peuvent apparaître, disparaître d'un instant à l'autre. De plus, dans les réseaux de citations, les réseaux de collaboration entre auteurs, les nœuds et les liens peuvent être ajoutés au réseau et ils ne peuvent pas être supprimés ultérieurement, ce qui confère à ce type de réseau le statut de réseau dynamique qui évolue au cours du temps. En effet, ces événements (changements) survenant sur le réseau, peuvent affecter les structures communautaires du réseau ; d'où la nécessité de ré-identifier et de suivre l'évolution de ces dernières. Les travaux de cette thèse s'inscrivent dans le contexte de l'identification et de suivi de la structure communautaire dans des réseaux dynamiques. Nous nous sommes intéressés particulièrement aux réseaux sociaux qui évoluent uniquement par l'ajout des nœuds et de leurs liens, autrement dit, les réseaux incrémentiels.

Problématique

Le suivi et l'identification de la structure communautaire dans les réseaux dynamiques représentent un domaine de recherche important classé dans la catégorie des problèmes NP-difficile. Parmi les premières solutions envisagées, la première solution proposée pour le traitement des réseaux dynamiques est l'utilisation des algorithmes statiques pour le suivi [24] [52] [41] de leurs structures communautaires. L'idée générale de ces approches est de scinder le réseau dynamique en une série d'instantanés qui sont tous des graphes statiques, ensuite la détection est réalisée en deux étapes : La première étape consiste à appliquer un algorithme statique sur chacun de ces instantanés, ce qui permet d'obtenir une série de partitions, une pour chaque instantané. La deuxième étape consiste à analyser la dynamique, c'est-à-dire à trouver une correspondance (association) entre les communautés existant dans des instantanés consécutifs. Ces approches permettent de réutiliser les méthodes de détection de communautés traditionnelles. Cependant, ces méthodes souffrent de problème de stabilité du fait que les algorithmes peuvent fournir des résultats très différents pour deux réseaux presque identiques. De plus, répéter le processus de détection de communautés statique sur chaque instantané et sur le réseau tout entier semble très coûteux en termes de temps avec une complexité algorithmique relativement élevée. Néanmoins, ces problèmes restent importants et toutes les autres approches tentent de les surmonter.

Il existe aussi des méthodes fondées sur l'optimisation de la modularité [47] [15], mais elles souffrent généralement du problème de limite de résolution dans le sens où elles ne peuvent pas distinguer des communautés plus petites d'une certaine taille limite. Cependant une communauté est d'autant plus forte qu'elle se rapproche d'une clique. La modularité ne traduit pas exactement ce critère de densité car elle a une portée globale dans tout le réseau. De plus, dans les réseaux dynamiques évoluant par l'ajout d'un nouveau nœud et de ses liens, le processus d'ajout dans ces méthodes se fait par le traitement de ses voisins (lien par lien) ce qui augmente systématiquement le coût de calcul.

Ainsi, afin d'assurer la stabilité des communautés du réseau, d'atténuer le problème de limite de résolution et de diminuer le temps de calcul, il va falloir suivre et identifier scrupuleusement la structure communautaire sur les réseaux dynamiques, plus particulièrement sur les réseaux incrémentiels dans notre cas.

Principales contributions

Pour répondre à la problématique citée, nous proposons une nouvelle approche basée sur la densité avec une double optimisation pour le suivi de la structure communautaire dans des réseaux dynamiques. Rappelons que le suivi se fait sur des réseaux sociaux qui évoluent par l'ajout des nœuds et de leurs liens.

Notre approche permet de :

1) Réduire le temps de traitement et les coûts de calcul :

- En utilisant la structure communautaire précédente pour identifier la structure communautaire courante.
- En ne prenant en compte que les communautés affectées par le changement.
- En ajoutant le nouveau nœud et ses liens simultanément au réseau considéré.

2) Pallier au problème de la limite de résolution de la modularité en maximisant la densité du réseau qui est localisée au niveau des communautés grâce à une double optimisation. Cette optimisation est réalisée par deux algorithmes d'optimisation locale. Le premier algorithme consiste en l'intégration d'un nœud avec ses liens à la communauté maximisant ainsi la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées. Tandis que le deuxième niveau d'optimisation cherche à maximiser au mieux le score de la densité du réseau par le test des opérations sur les communautés.

3) Assurer la stabilité des communautés entre deux instantanés de réseau. En effet, elle ne traite que les communautés (infectées) touchées par le changement tandis que les autres communautés restent inchangées.

Pour valider notre approche et vérifier la faisabilité de ces propositions, nous avons conduit nos expérimentations sur trois fronts.

Dans le premier, nos tests montrent la capacité de notre approche à suivre et à identifier efficacement les communautés et à évaluer sa performance en fonction de son temps de traitement sur des données du monde réel.

Dans le second, nos tests consistent à examiner la structure communautaire obtenue en termes de qualité, stabilité et validité en comparant les résultats à ceux obtenus avec des algorithmes dédiés aux réseaux statiques et dynamiques respectivement.

Enfin, nos tests montrent également la capacité de l'algorithme à atténuer le problème de la limite de résolution dans un réseau incrémental.

Organisation du document

Ce document est organisé comme suit :

- Une introduction (**Introduction générale**) où nous avons délimité le contexte de notre étude et identifié les différentes problématiques associées. Nous avons dans ce contexte tracé quelques brins de pistes qui donnent une idée sur l'objectif principal que nous nous sommes assigné. Nous avons au passage donné un aperçu général sur nos contributions dont la démarche est détaillée au chapitre 4. Nous avons également tracé les grandes lignes de la mise en œuvre et des expérimentations réalisées pour valider notre approche. Enfin, nous terminons par l'organisation du document.

- Dans le Chapitre 1 : **Ch1** (**Graphes et détection de communautés**) nous avons introduit quelques concepts de base sur les graphes et les communautés. En premier lieu, nous avons présenté quelques définitions relatives à la théorie des graphes ainsi que des exemples de systèmes modélisés par des graphes. En second lieu, nous avons donné les différentes définitions relatives au terme communauté. Nous avons aussi décrit et expliqué ce qu'est une fonction de qualité d'une partition d'un réseau en communauté et ses limites d'optimisation.

- Dans le Chapitre 2 : **Ch2** (**Réseaux dynamiques et détection de communautés**), nous avons commencé par un aperçu introductif afin d'avoir une idée sur les réseaux et les communautés dynamiques. Nous avons ensuite consacré une première section à la définition des réseaux dynamiques, appuyée par un exemple sur un réseau social. La section suivante est dédiée aux communautés dynamiques dans lesquelles nous avons décrit les communautés dynamiques et leurs opérations évolutives.

- Le Chapitre 3 : **Ch3** consiste en un (**Etat de l'art**) des méthodes existantes de détection de communautés. Il décrit les principales approches proposées dans la littérature. Ces dernières sont organisées en trois catégories. La première catégorie englobe les méthodes de partitionnement de graphes, qui constituent les premières solutions au problème de détection de communauté. La seconde catégorie

est dédiée aux méthodes de détection de communautés statiques. Enfin, la dernière catégorie est consacrée aux méthodes de détection de communautés dynamiques.

- Le Chapitre 4 : **Ch4 (Approche pour le suivi des structures communautaires dans les réseaux dynamiques)** constitue le vif du sujet. Il a été consacré entièrement à la description de l'approche proposée dans cette thèse. Dans un premier volet les points essentiels de l'approche ont été présentés. Ensuite quelques définitions utilisées tout au long du chapitre ont été ajoutées. Enfin, sont expliqués et détaillés également les deux niveaux d'optimisation de la densité du réseau en s'appuyant sur des exemples illustratifs et très significatifs.

- Le chapitre 5 : **Ch5 (Expérimentation et tests de validation)** est dédié, comme son nom l'indique, à l'expérimentation des différentes propositions. Ces expérimentations ont été menées sur différentes collections de données du monde réel. Les premiers tests visent à montrer la capacité de la méthode proposée à détecter efficacement les communautés dynamiques et à évaluer ses performances en matière de temps de traitement, tandis que les seconds tests visent à démontrer la qualité, la stabilité et la validité de la structure communautaire obtenue. Les derniers tests ont été effectués sur un réseau incrémental HEP-TH pour prouver la capacité de la méthode à atténuer le problème de la limite de résolution. Les résultats montrent clairement l'intérêt de l'approche comparativement à celles rapportées dans l'état de l'art.

- Enfin, une **(Conclusion générale)**, où nous avons confronté notre contribution avec certains travaux connexes (Méthode proposée Vs. Certaines méthodes rapportées dans l'état de l'art). Nous avons ensuite récapitulé les points essentiels de notre approche et rappelé quelles sont ses perspectives de recherche future (Synthèse et perspectives) envisagées.

Ch1

Graphes et Détection de Communautés

1.1 Introduction

Le concept de réseau existe dans plusieurs domaines de recherche comme l'informatique, la biologie, la linguistique, etc. Ces systèmes complexes peuvent être modélisés en termes de graphes, où un nœud représente un membre individuel du système et une arête représente un lien entre les membres selon une relation bien déterminée du système.

Dans de nombreux réseaux, il existe des zones où les nœuds sont fortement connectés entre eux et faiblement reliées aux autres zones du réseau. Ces zones sont appelées communautés et leur détection constitue une aide pour la compréhension et l'analyse des réseaux. La détection de communautés consiste à trouver une manière de scinder le réseau en un nombre inconnu de groupes de nœuds partageant l'idée de connexions internes fortes et de connexions externes faibles.

Ce chapitre a pour but principal de faciliter la lecture de ce mémoire, il rapporte ainsi l'essentiel concernant les graphes. Dans notre étude, nous nous focaliserons particulièrement sur des graphes non orientés et non pondérés. Nous commençons d'abord par donner quelques définitions de certains concepts de base de la théorie des graphes. On s'appuiera à cet effet sur des exemples de modélisation moyennant les graphes. Nous décrirons au passage les caractéristiques communes des graphes. Nous donnerons par ailleurs les différentes définitions relatives au terme

"communauté" dans les réseaux. Nous terminons par la description de ce qu'on appelle "fonction de qualité".

1.2 Définitions relatives à la théorie des graphes

Définition 1.1 (Graphe). Soit $G = (V, E)$ un graphe non orienté, avec V l'ensemble de ses sommets (nœuds) et E l'ensemble de ses arêtes (liens). Le graphe est dit d'ordre n et de taille m avec :

$n = |V|$ est le nombre de nœuds du graphe.

$m = |E|$ est le nombre de liens du graphe.

(i, j) une arête du graphe. i et j sont dits adjacents, ou voisins.

- **Le chemin** entre deux nœuds est une séquence de liens consécutifs dont ils sont les extrémités.
- **La longueur** d'un chemin est le nombre de liens qu'il comporte.
- **La distance** entre deux sommets est le minimum des longueurs des chemins allant d'un sommet à l'autre.
- **Le diamètre** d'un graphe est la plus grande distance entre deux sommets de ce graphe.

Définition 1.2 (Graphe complet). Un graphe possédant n sommets tous reliés deux à deux par un lien est dit complet.

Définition 1.3 (Graphe pondéré). Un graphe $G = (V, E, W)$ est pondéré lorsqu'un poids (ou une valeur) positif $w(u, v)$ est attribué à chaque lien $(u, v) \in E$.

Définition 1.4 (Graphe connexe). Un graphe $G = (V, E)$ est connexe si, quels que soient les sommets u, v de V, il existe un chemin de u vers v.

Définition 1.5 (Sous-graphe). Un sous-graphe d'un graphe G est un graphe constitué de certains sommets de G et de toutes les arêtes qui les relient.

Définition 1.6 (Clique). Une clique de G est un sous-graphe complet de G. On parle de K-clique pour désigner K sommets.

Définition 1.7 (Degré d'un sommet). Le degré d(v) d'un sommet $v \epsilon V$ est le nombre de liens incidents au sommet v, c'est-à-dire $|N(v)|$ où $N(v) = \{u \epsilon V, (v,u) \epsilon E\}$ est le voisinage du sommet v. Le degré moyen d'un graphe G, noté λ_G est la moyenne de cette valeur (degré d(v)) pour tous les sommets.

$$\lambda_G = \frac{1}{n}\sum_{v \in V} d(v) = \frac{2m}{n}$$

Définition 1.8 (Le coefficient de clustering d'un nœud). Soit un graphe $G = (V,E)$, un nœud $i \in V$, K_i est le nombre de voisins de i, n_i est le nombre d'arêtes entre ces voisins. Le coefficient de clustering du nœud i est défini par :

$$CC_i = \begin{cases} \frac{n_i}{k_i} & si\ k_i > 1 \\ 0 & si\ k_i = 0\ ou\ 1 \end{cases}$$

Ce coefficient traduit en fait la probabilité que deux voisins du nœud i soient reliés.

Définition 1.9 (Matrice d'adjacence). Une matrice d'adjacence A d'un graphe G d'ordre n est une représentation matricielle exactement équivalente au graphe. Cette matrice $(n \times n)$ est binaire, avec $a_{a,b} = 1$ s'il existe un lien entre les nœuds x_a et x_b, $a_{a,b} = 0$ dans le cas contraire. On notera que les éléments diagonaux sont nuls.

Définition 1.10 (Matrice des degrés). Etant donné un graphe G (V,E) contenant n sommets, la matrice D de degré de G est une matrice carrée $n \times n$ qui contient des informations sur le degré de chaque sommet du graphe. Elle est définie par :

$$d_{i,j} = \begin{cases} \deg(sommet_i) & si\ i = j \\ 0 & sinon \end{cases}$$

Définition 1.11 (Matrice Laplacienne). [37] La matrice Laplacienne (ou de Laplace) est définie comme la différence entre la matrice de degré D et la matrice d'adjacence A.

$$M\_lap = D - A.$$

Définition 1.12 (Coupe du graphe). Une coupe constitue une partition de G (V,E) en deux ensembles V' et $V'' = V \backslash V'$. Les arêtes reliant ces deux ensembles sont les

arêtes à retirer pour effectuer une coupe. La coupe correspond donc à l'ensemble des arêtes ayant une extrémité dans V' et l'autre dans V''.

Définition 1.13 (Coupe minimale). Une coupe minimale est une coupe, qui dans un graphe non pondéré, minimise le nombre d'arêtes retirées pour séparer le graphe en deux.

Définition 1.14 (Partition des sommets d'un graphe). Soit un ensemble S quelconque. Un ensemble P de sous-ensembles de S est appelé une partition de S si :
1. Aucun élément de P n'est vide.
2. L'union des éléments de P est égale à S.
3. Les éléments de P sont deux à deux disjoints.
Les éléments de P sont appelés les parties de la partition P.

1.3 Exemple de systèmes modélisés par des graphes

La notion de réseau existe dans plusieurs domaines de recherche, en particulier dans les disciplines de l'informatique. La modélisation des réseaux par des graphes facilite l'étude et la compréhension de leur structure en s'appuyant sur la théorie des graphes. Les graphes sont constitués de nœuds et de liens éventuellement orientés et libellés (étiquetés). Pour illustrer plus clairement l'usage des graphes comme outils de modélisation et de représentation nous pouvons à cet effet citer les exemples suivants :
- Dans *les réseaux sociaux*, les nœuds représentent des individus et les liens peuvent être de différentes natures : connaissance entre individus (deux individus sont reliés s'ils se connaissent), collaborations (deux individus sont reliés s'ils ont travaillé ensemble), appels téléphoniques (deux individus sont joints s'il y a eu un appel entre eux, échange de mails ou de fichiers, etc.). [35]
- Les *réseaux d'infrastructure* modélisent des connexions matérielles entre objets. C'est le cas de réseaux de transport (les routes entre les villes ou les liaisons aériennes entre aéroports) ou des réseaux physiques d'Internet (câble entre ordinateurs) [18].
- Plusieurs types de *réseaux biologiques* ont été modélisés par des graphes. Par exemple, les réseaux métaboliques, où les nœuds sont des gènes ou des protéines

reliées par des interactions chimiques [26], les réseaux de neurones (chaque neurone est connecté à plusieurs autres neurones).
- Les *réseaux d'information* : L'exemple classique du réseau d'information est le réseau de citations des articles de recherche. La plupart des articles citent les travaux précédents des autres auteurs sur le même sujet. Ces citations forment un réseau dont les sommets sont des articles ; un lien orienté de l'article A vers l'article B indique que B est cité par A [17]. Un autre exemple est le réseau World Wide Web, ou les sommets sont des pages web liées par des liens hypertextes.
-*Réseaux linguistiques* : Ces réseaux relient les mots d'un langage donné et regroupent entre autres les réseaux de synonymes (deux mots sont reliés s'ils sont synonymes.), les réseaux de co-occurrences (deux mots sont reliés s'ils apparaissent dans une même phrase d'un ouvrage) ou encore les réseaux de dictionnaires [5] (deux mots sont liés si l'un est utilisé dans la définition de l'autre).

1.4 Caractéristiques communes des graphes de terrains

Avant de décrire les caractéristiques communes des graphes de terrain, il convient de définir au préalable qu'est-ce qu'un graphe de terrain. Un *graphe de terrain* (*complex networks*) est un graphe obtenu à partir des données réelles correspondant à une réalité de terrain. Ces graphes peuvent être rencontrés dans le monde réel dans différents domaines tels que les sciences sociales, l'informatique, la linguistique, etc. Ceux sont des graphes variables en taille, de quelques dizaines à quelques milliards de sommets, qui possèdent des propriétés communes. Nous décrivons ici les caractéristiques communes les plus importantes de ces graphes.

Effet petit-monde

Le phénomène de petit monde (*Small World*) est l'hypothèse que chacun est relié à n'importe quel autre individu grâce à de « courtes chaînes » de relations sociales. Le concept a engendré l'expression célèbre des "six degrés de séparation" après l'expérience du petit monde de 1967, réalisée par le psychologue Stanley Milgram [36]. Dans cette expérience, il a mis en évidence des chaînes très courtes reliant deux citoyens aléatoirement choisis aux États-Unis (les chaînes effectivement ob-

tenues, au nombre de quelques dizaines, avaient une longueur moyenne de six personnes.). Cette propriété est une des caractéristiques communes des graphes de terrain : ils possèdent tous *une faible distance moyenne*. Le terme petit monde dans certains contextes implique aussi *un fort coefficient de clustering*.[1] Des expériences contemporaines via Internet continuent d'explorer ce phénomène.

Variation de densité et structure communautaire

La majorité des graphes de terrain ont un coefficient de clustering élevé. Cela signifie que s'il existe une relation d'amitié entre un individu X et un individu Y dans un réseau social et une relation entre Y et Z alors X et Z ont une grande probabilité de se connaître (propriété de transitivité). Dans ces graphes, la densité est très forte localement et faible globalement. Cela s'explique par la capacité des sommets à se regrouper en groupes appelés *communautés*. Cette caractéristique jouera un rôle clé dans cette thèse, nous la décrirons plus en détail dans la section suivante.

1.5 Détection de communautés

Nous présentons dans cette section les diverses définitions relatives à la notion de communauté. Nous donnons ensuite une explication de la fonction de qualité "la modularité" et nous décrivons ses limitations.

1.5.1 Définir une communauté

A- De manière intuitive

Les communautés peuvent être considérées comme des groupes d'entités qui partagent des caractéristiques communes ou jouent des rôles similaires : personnes proches (familles, amis, clients), pages web traitant d'un même sujet, protéines ayant une même fonction biologique. Intuitivement, sur un réseau social par exemple, on peut facilement être convaincu que les individus se regroupent naturellement en communautés correspondant à des groupes d'amis, de collègues de travail, familiaux, etc. La notion de communauté dans un graphe est cependant difficile à définir formellement. C'est pourquoi toutes les approches récentes ont

[1] *Clustering est le terme Anglais qui signifie partitionnement, regroupement ou segmentation, etc.*

utilisé une notion intuitive des communautés. Une communauté est alors vue comme un ensemble de sommets dont la densité de connexions internes est plus forte que la densité de connexions vers l'extérieur.

B- Dans la théorie des graphes

Diverses définitions ont été adoptées pour le terme communauté, la théorie des graphes répond à des contraintes telles que : "Comment exprimer mathématiquement l'appartenance d'un nœud à une communauté plutôt qu'à une autre ? " par les définitions suivantes :

- *Une clique* est un sous-graphe complet maximal, ou chaque nœud est lié avec tous les autres. Il est évident qu'une *clique* possède les propriétés requises pour une communauté, mais elle impose aussi des restrictions importantes, surtout pour les réseaux peu denses.
- *k-core* est un sous-graphe de dimension supérieure à k, où chaque nœud doit avoir des liaisons vers au moins k autres nœuds. Même si les contraintes d'un *k-core* sont moins dures, il désavantage les nœuds de faible degré.

Les notions de *clique* et de *k-core* sont basées seulement sur l'analyse d'un sous-graphe donné. Une troisième définition, qui prend en compte la totalité du graphe est celle d'un *LS set*.

- *Un LS set* est un groupe de nœuds qui présente la propriété que chaque sous-groupe de ce groupe possède plus de liaisons vers l'intérieur du groupe que vers le reste du graphe.

C- Autres définitions

À part ces définitions, assez restrictives, qui peuvent être utilisées aussi dans la détection de communautés, il existe trois autres essais d'exprimer mathématiquement le concept. Deux de ces définitions, celle de communauté au sens *fort* et celle de communauté au sens *faible*, sont données par Radicchi et al [43].

Communauté au sens fort

La définition *forte* de la communauté tient compte de chaque nœud du sous-ensemble. Ainsi, un sous-graphe V est considéré comme une communauté si :

$$K_i^{in}(V) > K_i^{out}(V), \qquad \forall\, i \in V$$

Ou $K_i^{in}(V)$ représente le nombre de liens du nœud i avec les nœuds de l'ensemble V et $K_i^{out}(V)$ est le nombre de liens du nœud i à l'extérieur de la communauté. Cette

définition impose une contrainte sur chaque nœud, ce qui n'est pas le cas pour une communauté dans le sens faible.

Communauté au sens faible :

Concrètement, dans une communauté forte, chaque nœud a plus de connexions au sein de cette communauté qu'avec le reste du réseau et dans une communauté *faible* le nombre total de liens internes est supérieur au nombre total des liens vers l'extérieur.

$$\sum_{i \in v} K_i^{in}(V) > \sum_{i \in v} K_i^{out}(V)$$

Plus récemment, les auteurs dans [25] ont proposé une nouvelle définition d'une communauté, encore moins contraignante. Dans leur définition, un nœud i fait partie de la communauté V_a s'il a au moins autant de liens vers cette communauté que vers n'importe quelle autre communauté :

$$K_i^{in}(V_a) \geq K_i^{in}(V_b), \forall b \neq a, \forall i \in V$$

Il existe peu d'algorithmes qui utilisent directement ces définitions. Dans la plupart des cas, on ne regarde même pas si le partitionnement final les vérifie, mais on préfère comparer les propriétés de ce partitionnement avec un modèle de référence qui provient d'une topologie similaire, mais sans une structure modulaire (le "null model").

Le fait qu'il n'existe pas une définition généralement reconnue pour la notion de communauté rend plus difficile la construction d'un algorithme accepté par tout le monde, mais, en même temps, permet de diversifier et particulariser les approches proposées pour détecter ce type de structures.

1.5.2 Qualité de partition : La Modularité

Étant donné un graphe, l'objectif est de le décomposer en un ensemble de communautés de sorte que la densité des connexions internes est plus forte que la densité des connexions externes. Une décomposition en communautés est ainsi une partition de l'ensemble des sommets et il existe de nombreuses fonctions pour juger de la qualité d'une partition d'un graphe donné, parmi elles, nous citons *la modularité* introduite par Girvan et Newman [38].

Soit un graphe $G = (V, E)$ ayant $n = |V|$ sommets et $m = |E|$ liens, ainsi qu'une partition $P = \{C_1 \dots C_p\}$ en communautés. Notons $\sum_c e_c$, la fraction de liens situés à l'intérieur des communautés, où e_c est le nombre de liens dans c. Selon la définition intuitive d'une communauté, pour avoir une bonne partition, la fraction de liens situés à l'intérieur des communautés doit être élevée c'est-à-dire que la valeur de $\sum_c e_c$doit être élevée. Or, on voit clairement que la valeur maximale de $\sum_c e_c$ est trouvée si l'ensemble du réseau est considéré comme une seule communauté, c'est-à-dire si $P = \{V\}$, car dans ce cas, tous les liens se trouvent dans cette communauté et $\sum_c e_c = 1$.

Pour y remédier, Girvan et Newman ont proposé une approche simple qui est devenue largement acceptée. Elle est basée sur l'idée intuitive que les réseaux aléatoires ne possèdent pas de structure communautaire. Ainsi, si l'on a trouvé une partition qui a du sens, on souhaite non seulement que $\sum_c e_c$ soit élevée et dans le même temps que la même partition sur un graphe aléatoire donne une faible valeur pour $\sum_c e_c$.

La fraction des liens au sein des communautés dans un réseau réel peut-être tout simplement comptée, et la valeur attendue pour un réseau aléatoire peut être calculée à partir du degré des sommets de la manière suivante :

Pour une partition P, si un lien est choisi au hasard, la probabilité a_c, qu'une extrémité de celui-ci mène à la communauté c, est le nombre de liens ayant une extrémité dans la communauté c divisé par le nombre total de liens m du réseau. Nous voyons clairement que la probabilité qu'un lien ait une extrémité dans la communauté c est simplement la proportion de demi-liens dans cette communauté, soit la somme des degrés des sommets de la communauté divisée par deux fois le nombre de liens :

$$a_c = \frac{\sum_{i \in c} d(i)}{2m}$$

La probabilité que les deux extrémités d'un lien soient dans la communauté c est donc a_c^2. De cela, l'expression générale de la modularité est donnée par l'équation (1.1) comme suit :

$$Q(P) = \sum_i (e_c - a_c^2) \tag{1.1}$$

La modularité est toujours comprise entre -1 et +1 et la définition de la modularité implique qu'une partition où tous les sommets sont regroupés dans la même communauté a une modularité nulle. Cela implique aussi qu'il est toujours possible de trouver une partition de modularité positive ou nulle, quel que soit le graphe considéré, bien qu'il existe des partitions de modularité négative. Une bonne modularité est donc toujours positive et la qualité augmente avec la modularité. Une autre façon de voir cette fonction de qualité est la suivante : Soit A la matrice d'adjacence du graphe G dont les éléments A_{ij} sont les poids des liens entre les sommets i et j, et valent donc 0 ou 1 dans le cas d'un graphe non pondéré. L'équation (1.1) peut alors être reformulée par l'équation (1,2) avec $d(i)=\sum_j A_{i,j}$ est le degré du sommet i et $\delta(C_i, C_j)$ est la fonction de Kronecker qui vaut 1 si $C_i = C_j$ et 0 sinon.

$$Q(P) = \frac{1}{2m} \sum_{i,j \in v} \left[A_{i,j} - \frac{d(i)d(j)}{2m} \right] \delta(C_i, C_j) \qquad (1.2)$$

C'est la formule (1.2) qui est généralement utilisée pour calculer la qualité d'une partition d'un graphe en communautés. Quelle que soit la manière dont on exprime la modularité, sa signification reste la même : la différence entre le nombre de liens à l'intérieur d'une communauté et le nombre de liens attendus à l'intérieur de cette communauté si les liens apparaissent aléatoirement dans le graphe tout en respectant la distribution des degrés des nœuds. Il a été montré que maximiser la modularité d'un réseau est un problème NP-difficile [9] et on ne peut donc chercher que des solutions approchées à l'aide de méthodes heuristiques. De nombreuses méthodes d'optimisation de la modularité ont été proposées au cours des dernières années avec deux objectifs principaux : l'amélioration de la qualité et la réduction de la complexité du calcul de l'algorithme.

Limites de l'optimisation de la modularité

Plusieurs algorithmes ont été proposés pour l'optimisation de la modularité et les auteurs [20] voient que ces algorithmes souffrent d'un problème de limite de résolution dans le sens où les petites communautés peuvent disparaître à cause des regroupements avec d'autres communautés similaires ou bien elles sont absorbées par des communautés relativement beaucoup plus grandes. Dans les graphes non pondérés par exemple la maximisation de la modularité ne permet pas de distinguer des communautés ayant un nombre de liens inférieur à $\sqrt{m/2}$ avec m le

nombre de nœuds (figure 1.1). D'autre part, on montre dans [30] que la maximisation de la modularité n'a pas seulement tendance à fusionner les petits groupes, mais aussi à éclater des grandes communautés, et il semble impossible d'éviter simultanément les deux problèmes.

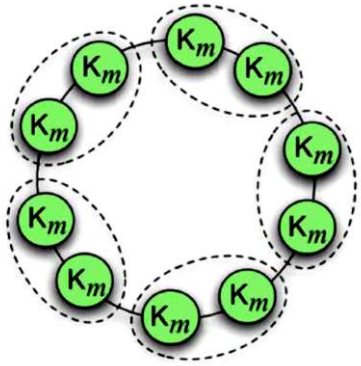

Figure 1. 1 Illustration de la limite de résolution[2]. [20]

Des remèdes variés ont été proposés, comme celui de Fortunato, qui consiste à réexaminer les grandes communautés comme des graphes autonomes pour vérifier si elles ne contiennent pas des sous-communautés non détectées.
Dans [50] les auteurs montrent que les algorithmes de maximisation de la modularité sont très sensibles à des perturbations minimales appliquées au graphe étudié.
De nombreuses approches ont été développées pour la détection de communautés par maximisation de la modularité malgré ces sérieux problèmes. Cependant, la modularité reste la fonction qualificative, la plus utilisée et son optimisation est encore un défi très intéressant.

1.6 Conclusion

Dans ce chapitre, nous avons introduit quelques notions de base de la théorie des graphes pour faciliter la lecture et la compréhension de ce document. Nous avons également discuté des différentes définitions de communauté pouvant être utilisées pour la détection de communautés statiques ou dynamiques. Le chapitre suivant porte sur les notions de réseaux dynamiques et de détection de communautés.

[2] K_m: *Une clique d'ordre m. A partir d'un anneau de cinq 2-cliques reliées deux à deux par un seul lien, la partition naturelle associe une communauté à chaque clique avec une modularité d'environ 0.65. Tandis que la modularité de cliques regroupées deux à deux sera 0.675, ce qui constitue une partition erronée.*

Ch2

Réseaux dynamiques et détection de communautés

2.1 Introduction

La création et la disparition de relations dans les réseaux sociaux impliquent que les réseaux évoluent dans le temps (réseaux dynamiques) contrairement aux réseaux statiques qui sont des réseaux constitués d'un ensemble de nœuds et d'un ensemble de liens entre ces nœuds, sans notion de temps, ou d'ordre, pour les caractériser. Les réseaux dynamiques sont modélisés par des graphes dynamiques[3] et la détection de communautés sur ces graphes reste un problème. En effet, il n'existe pas de définition de communautés dynamiques faisant le consensus. On peut les définir comme une succession de communautés statiques, mais aussi, plus directement, suivant la façon de modéliser les graphes dynamiques. Ce chapitre vise à élucider la notion de dynamicité (dynamique) dans les réseaux. Pour ce faire, nous définissons les réseaux dynamiques ainsi que les communautés dynamiques et leurs évolutions au fil du temps.

2.2 Les réseaux dynamiques

Un réseau dynamique est un réseau en évolution dans lequel les changements se produisent au fil du temps. Ces changements se produisent lorsque de nouveaux

[3] *Comme son nom l'indique, un graphe dynamique est un graphe variable avec le temps. Autrement dit, un graphe dynamique est une collection de graphes, chacun pour un temps donné.*

membres rejoignent le réseau, que des membres existants le quittent ou que des membres existants établissent une nouvelle relation ou mettent fin à une relation existante dans le temps. Ces changements semblent avoir peu d'effet sur la structure du réseau dans son ensemble. Cependant, ils peuvent conduire à des transformations significatives de la structure communautaire. Cela soulève un besoin naturel de ré-identification de la structure communautaire au fil du temps. Comme exemple, on pourrait citer les réseaux sociaux qui changent avec le temps. Ceci est dû à l'arrivée des individus, et à la disparition ou l'apparition de nouvelles relations entre eux. Ces réseaux sont modélisés par des graphes à chaque instant et la figure 2.1 montre un exemple de cette modélisation.

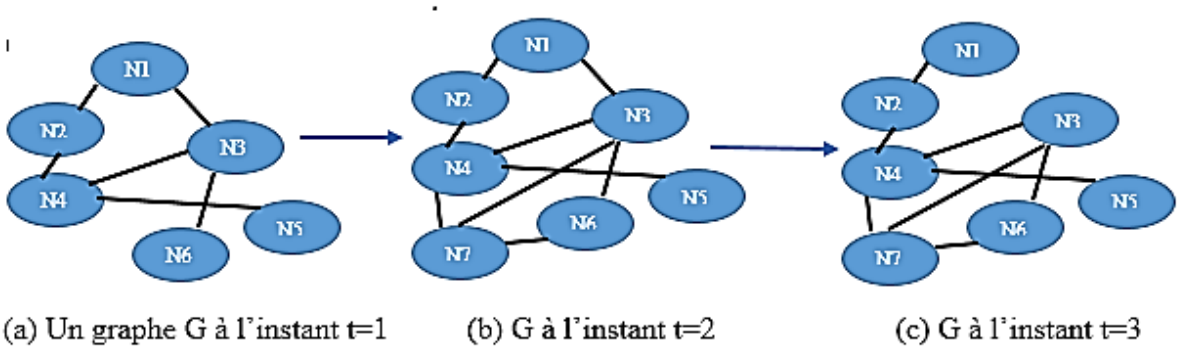

Figure 2. 1 Modélisation d'un réseau social sous forme d'un graphe dynamique sur trois instants.

Dans la figure 2.1, nous remarquons que le graphe dynamique G est une succession de graphes statiques G à t=1, t=2, t=3. On peut constater que dans la représentation dynamique de G les nœuds et les liens changent. Par exemple de t1 à t2, on peut noter l'apparition du nœud N7 et l'apparition des relations (N7, N3), (N7, N4), (N7, N6). De même, de t=2 à t=3, on remarque la disparition de la relation (N1, N3). Ces graphes statiques sont appelés des instantanés, car ils représentent l'image du réseau social dynamique à un instant donné.

2.3 Les communautés dynamiques

2.3.1 Définition

Les communautés dynamiques peuvent changer ou évoluer avec le temps. L'évolution de ces communautés peut être définie de deux manières différentes :

- En tant que séquence d'événements (modifications) qui ne se succèdent par pas de temps. En d'autres termes, l'évolution est décrite par les transformations identifiées des communautés d'un instantané à l'autre.
- En tant que communauté statique initiale et séquence de changements sur cette communauté, à savoir l'intégration des nœuds et leur exclusion.

2.3.2 Opérations d'évolution de communautés

Dans les réseaux dynamiques, les communautés sont amenées à évoluer et à changer à cause de l'apparition ou la disparition de nœuds ou de liens. Différentes opérations peuvent apparaitre et une description de chacune peut être trouvée ci-dessous :

-*La croissance et la contraction de communautés* (growth and contraction) : dans les réseaux réels, la croissance d'une communauté correspondant à l'ajout de nouveaux nœuds et la contraction au retrait de nœuds d'une communauté existante.
-*La naissance et la mort de communautés* (birth and death) : des nouvelles communautés peuvent apparaître, et d'anciennes communautés peuvent disparaître avec l'évolution de réseau.
- *La fusion et la division de communautés* (merging and splitting) : deux communautés - ou plus - peuvent en effet, se fusionner en une seule au cours du temps. De manière semblable, une communauté peut se diviser en deux ou plus en communautés, plus petites que celle dont elles sont issues.
C'est Palla et al dans [41], qui ont été les premiers à définir ces opérations. Ils en ont proposé une représentation schématique illustrée dans la figure 2.2.

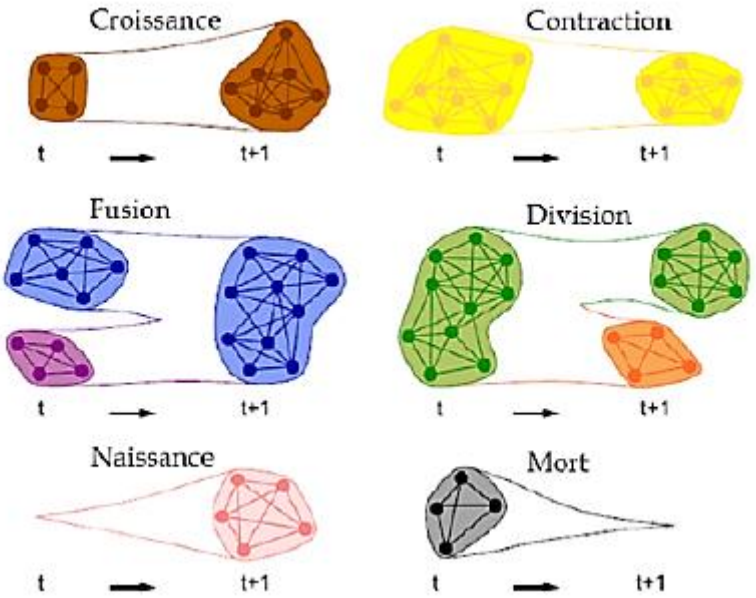

Figure 2. 2 Représentation schématique de quelques opérations possibles sur les communautés dynamiques. [41]

D'autres auteurs comme [10] ont identifié une autre opération : la résurgence, lorsqu'une communauté disparaît pendant un certain temps et réapparaît plus tard sous forme identique ou très proche (Figure 2.3).

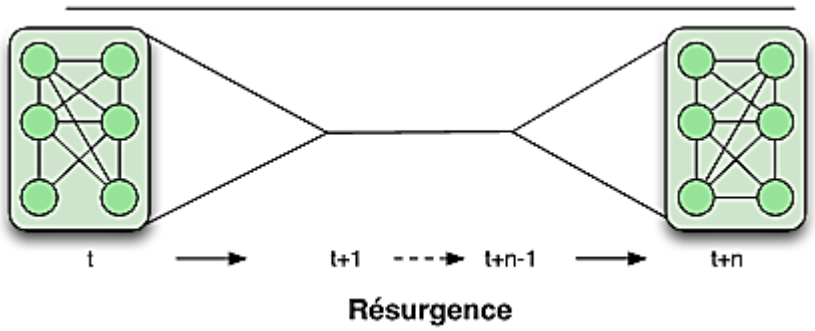

Figure 2. 3 Représentation de l'opération de résurgence. [10]

2.3.3 Détection de communautés dynamiques

L'absence d'une définition unique de la communauté est l'un des principaux problèmes de la détection de communautés. Dans les études traitant ce problème, nous avons sélectionné trois catégories d'approches traitant la détection de communautés et le suivi de la structure communautaire. La première catégorie identifiée est constituée de méthodes utilisant les algorithmes statiques pour le suivi des communautés [24], [48], [8], [52], [58], [41]. La deuxième englobe les méthodes par

détection directe [11], [56] et la dernière contient les méthodes incrémentales [49], [15], [40], [54]. Les détails de chaque méthode sont donnés dans le chapitre 3 à la section 3.2.3. Quel que soit le type de réseau étudié, pondéré ou non pondéré, orienté ou non orienté et quel que soit le type de communautés produites disjointes ou chevauchantes [62], le but de ces travaux est de trouver une solution pour pallier au problème de détection de communautés et de suivi la structure communautaire dans les réseaux dynamiques.

2.4 Conclusion

Dans ce chapitre sur les réseaux dynamiques et la détection de communautés, nous avons donné une idée sur ce qu'est un réseau dynamique. Nous avons défini ensuite ce qu'est une communauté dynamique. Nous avons au passage également mis l'accent sur les différentes opérations (changements) d'une communauté dynamique dans laquelle nous avons cité les plus basiques. Enfin, nous avons donné une idée sur la détection de communautés dynamiques et cité également quelques catégorisations d'approches traitant de la détection de communautés et de suivi de la structure communautaire. Nous en fournissons de plus amples détails et d'explications à cet effet au chapitre 3.

Ch3

Etat de l'art

3.1 Introduction

La détection de communautés dans les réseaux est un sujet relativement récent. Elle consiste à décomposer un réseau en un nombre inconnu de groupes de nœuds partageant l'idée de connexions internes fortes et de connexions externes faibles. Plusieurs critères ont été utilisés à cet effet, ce qui a conduit à la multiplication de méthodes dans le domaine. À ce titre, plusieurs travaux sur la détection de communautés dans des réseaux [46], [27], [19], [2] ont vu le jour.

Dans ce chapitre, nous décrivons les principales approches existantes. Pour mettre en exergue les caractéristiques des méthodes, nous avons scindé ces dernières en trois catégories : la première catégorie concerne les méthodes de partitionnement de graphe qui sont à l'origine des premières solutions répondant au problème de détection de communautés. Les deux autres catégories sont regroupées selon le critère "type de communauté" statique ou dynamique. Elles concernent les méthodes de détection de communautés statiques et les méthodes de détection de communautés dynamiques.

La catégorie la plus récente concerne les méthodes de détection de communautés dynamiques. De fait, les méthodes statiques n'étaient capables que de trouver un découpage sur un graphe défini pour un moment donné. Or, beaucoup de graphes de terrain ont en fait la propriété de changer, d'évoluer au cours du temps, des nœuds et des liens pouvant apparaître ou disparaître. Trouver des communautés

dans de tels réseaux demande de prendre en considération leurs différentes étapes d'évolution, de manière à donner des communautés cohérentes non pas à un moment donné, mais sur une période donnée, avec leurs éventuelles modifications au cours du temps. L'étude des réseaux dynamiques est un domaine qui suscite aujourd'hui de plus en plus d'intérêt, de par les nouvelles opportunités — et les nouveaux challenges — qu'il propose.

Si la méthode proposée dans cette thèse s'inscrit dans cette dernière catégorie, il est utile, et même nécessaire de faire un état de l'art des techniques existantes pour les catégories précédentes, parce que la plupart des méthodes récentes sont basées sur des méthodes plus anciennes.

3.2 Les méthodes de détection de communautés

3.2.1 Le partitionnement de graphe

Le partitionnement de graphe se rapporte aux méthodes supervisées (nombre et taille des communautés connues), il consiste à partager l'ensemble des nœuds d'un graphe en k groupes, k étant préalablement fixé de manière à minimiser le nombre d'arêtes entre les groupes. Dans le partitionnement de graphe, on cherche toujours à minimiser le nombre d'arêtes coupées entre les différentes parties (minimiser le coût de coupe d'une partition).

-*Le coût de coupe* d'une partition est : une partition $P_k = \{S_1, \ldots\ldots S_k\}$ de S en k parties exprimée par l'équation (3.1) comme suit :

$$coupe\ (S_1, S_2) = \sum_{s_1 \in S_1, s_2 \in S_2} poids(s_1, s_2) \qquad (3.1)$$

D'où : $coupe\ (P_k) = \sum_{i<j}(S_i, S_j)$

En outre, il existe des partitionnements de graphes qui demandent de trouver une partition qui minimise des fonctions objectives qui sont *le ratio de coupe* et *la coupe normalisée*.

-Le ratio de coupe représente, pour chaque sous-ensemble, le poids total de ses arêtes coupées sur son poids total. Il est minimal quand le poids des sous-ensembles est maximal et quand le coût de coupe est minimal.

$$ratio(P_k) = \sum_{i=1}^{k} \frac{coupe(S_i, S - S_i)}{poids(S_i)} \quad (3.2)$$

-*La coupe normalisée* correspond, pour chaque sous-ensemble, le poids total de ses arêtes coupées sur la somme de son poids et du poids total de ses arêtes coupées.

$$norm(P_k) = \sum_{i=1}^{k} \frac{coupe(S_i, S - S_i)}{coupe(S_i, S)} \quad (3.3)$$

Il existe aussi des partitions équilibrées (les parties sont de taille similaire lorsque la balance ≤ 1.50). On introduit alors dans ce cas une mesure de cet équilibre, appelée *balance de partitionnement* :

$$\text{balance}(P_k) = \frac{max_i poids(S_i)}{poids_{moy}} \quad (3.4)$$

Avec : $poids_{moy} = \frac{poids(S)}{K}$

En pratique, la plupart des approches de partitionnement de graphes procède par une division du graphe en deux sous-graphes, puis par un partitionnement récursif des deux sous-graphes ainsi obtenus. Les méthodes ayant connu plus de succès sont la méthode de bissection spectrale [3] et la méthode de Kernighan-Lin [28].

3.2.1.1 La méthode de bissection spectrale

La bissection spectrale [3] cherche à établir une coupe minimale en nombre d'arêtes qui sépare un graphe en deux groupes égaux en taille. Cette coupe peut s'écrire en utilisant la matrice Laplacienne. On utilise la plus petite valeur propre non nulle de la matrice de la place et le vecteur propre associé. Les nœuds qui donnent des composantes négatives dans ce vecteur sont considérés comme faisant partie d'un sous-graphe et les nœuds qui donnent des composantes positives sont inclus dans un autre sous-graphe.

3.2.1.2 La méthode de Kernighan-Lin

L'idée principale de cet algorithme [28] est de trouver deux sous-ensembles de sommets de même taille, chacun dans une partie de la bissection, tels que leur échange diminue le coût de coupe de la bissection. L'algorithme commence d'une bissection existante, échange successivement deux sous-ensembles de la bissection jusqu'à ce qu'il ne soit plus possible de diminuer le coût de coupe. La dernière bis-

section obtenue est donc la bissection de coût de coupe minimal trouvé par l'algorithme et pour l'obtenir il est préférable de partir d'une bissection de coût de coupe faible. La complexité temporelle est en $O(n^2)$ avec n correspondant au nombre de nœuds.

3.2.2 Détection de communautés statiques

Dans les réseaux de terrain de grande taille qui représentent les données complexes (réseaux sociaux, réseaux d'information, etc.), le nombre de groupes que l'on cherche à obtenir ne peut pas être connu à l'avance. Dans ces réseaux, le nombre de communautés existant est en fait, en lui-même, un résultat important. Un nouveau problème ainsi posé est de décomposer le réseau en un ensemble de sous-graphes interconnectés, chacun constituant ce que l'on appelle une *communauté* sans exiger la connaissance à priori du nombre de communautés.

Un nombre très important d'algorithmes ont été proposés [19], [21]. Comme nous ne pouvons pas décrire ici toutes les méthodes, nous en présenterons une sélection, parmi les plus connues en adoptant l'organisation hiérarchique suivante:

-Les méthodes de clustering hiérarchiques

-Méthodes hiérarchiques ascendantes (agglomerative clustering)

-Méthodes hiérarchiques descendantes (separative clustering)

-Autres méthodes

-Méthodes basées sur la propagation d'étiquettes

-Méthodes utilisant les marches aléatoires

-Méthodes basées sur les cliques

3.2.2.1 Les méthodes de clustering hiérarchiques

Les méthodes hiérarchiques cherchent à construire une hiérarchie de partitions qui sont présentées sous forme d'un arbre appelée *dendrogramme*. Ce dendrogramme donne une idée du nombre de clusters existant effectivement dans l'ensemble des objets. Une fois l'arbre construit, l'utilisateur doit trouver le meilleur endroit pour couper l'arbre (c'est à dire là où les clusters sont plus éloignés). Sur la figure 3.1, la méthode propose trois clusters : vert, rouge et bleue.

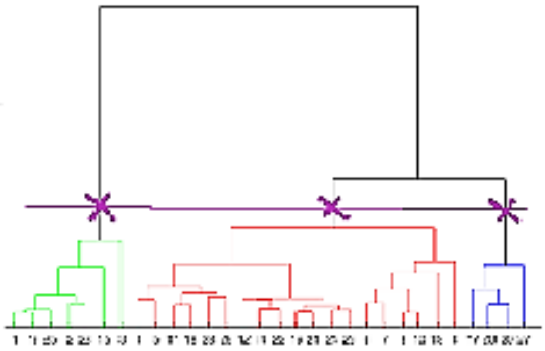

Figure 3. 1 Arbre hiérarchique dit dendrogramme

Les approches de clustering hiérarchique sont de deux types : descendants et ascendants.

Les méthodes hiérarchiques ascendantes

Les approches hiérarchiques ascendantes supposent au départ que chaque nœud forme une communauté. Les communautés les plus proches sont fusionnées pour former une nouvelle communauté jusqu'au moment où tous les nœuds sont dans la même communauté. Le calcul de similarité entre deux communautés nécessite l'utilisation d'une mesure de similarité entre chaque paire de nœuds (la distance euclidienne, coefficient de jaccard, etc.), ainsi qu'un critère ou indice d'agrégation dans lequel il en existe plusieurs : Le lien complet (*complete linkage*) ou agrégation par le diamètre voit que la distance entre deux communautés est la grande distance entre les sommets de celle-ci. A l'opposé, on peut considérer la distance minimum (*single linkage)*. De manière intermédiaire, dans le critère lien moyen (*average linkage),* la distance entre deux communautés peut être considérée comme la distance moyenne entre chaque paire de leurs sommets. Dans cette catégorie, nous citons l'approche de Newman et celle de Louvain.

A- La méthode de Newman

Girvan et Newman [38] ont introduit dans leur article une approche basée sur l'optimisation de la modularité. Initialement, chaque nœud est une communauté. Pour toutes les paires de communautés voisines, la modification de la modularité en cas de fusion est calculée et les deux communautés qui apportent le gain le plus important sont réunies dans une seule. Le calcul de la modularité est effectué de

manière itérative jusqu'au moment où aucun gain n'est plus possible. La complexité est de l'ordre de $O(mn)$ et afin d'améliorer cette complexité Clauset et al [14] ont proposé une structure de donnée adaptée.

B- La méthode de Louvain

La méthode de Louvain [6] implante une méthode d'optimisation de la modularité. Au début, chaque nœud est mis dans une communauté différente. L'algorithme applique ensuite une itération de succession de deux phases :

Phase d'affectation des nœuds : pour évaluer le gain de la modularité, on regarde si le placement d'un nœud dans la communauté d'un de ses voisins apporte un gain en modularité. Si c'est le cas, ce nœud va se déplacer dans cette communauté, sinon il reste dans l'ancienne communauté.

Phase de compression : la deuxième phase de l'algorithme commence en remplaçant chaque communauté par un seul nœud. Deux nœuds c_x, c_y dans le nouveau graphe sont liés par un lien s'il existe, un lien entre un nœud de la communauté représentée par c_x et un nœud de la communauté représentée par c_y. Le poids de lien entre deux communautés est égal à la somme des poids des liens reliant des nœuds de deux communautés. L'algorithme (cf. Algorithme 3.1) s'arrête s'il a y plus de possibilité de réaffectation de nœuds ou si un maximum de modularité soit atteint.

Algorithme 3. 1 Pseudocode de l'algorithme de Louvain

```
1 : G le graphe initial.
2 : Répéter
3:      Placer chaque sommet de G dans une unique communauté ;
4:      Sauvegarder la modularité de cette décomposition ;
5:      Tant que il y a des sommets déplacés faire
6:          Pour tout sommet n de G faire
7:                  Chercher c la communauté voisine maximisant le gain de modularité ;
8:                  Si c induit un gain strictement positif alors
9:                      déplacer n de sa communauté dans c ;
10 :                Fin Si
11 :        Fin Pour
12 :     Fin Tant que
13:      Si la modularité atteinte est supérieure à la modularité initiale alors
14:          fin ←faux ; //un booléen
15:          Afficher la décomposition trouvée ;
16:          Transformer G en le graphe entre les communautés ;
17:      Sinon
18:           fin ←vrai ;
19 :     Fin Si
16:jusqu'à fin
```

La figure 3.2 montre un exemple de fonctionnement de la méthode de Louvain.

Cette méthode a connu une large diffusion, grâce à sa faible complexité qui est en $O(n\ log\ n)$. Il s'agit sans conteste de l'une des méthodes les plus connues et la plus utilisée dans de nombreuses applications, capable de traiter des graphes ayant plusieurs milliards de liens.

Cette méthode sera utilisée dans le chapitre expérimentation de ce travail pour comparer les résultats qu'elle trouve avec celles de notre algorithme.

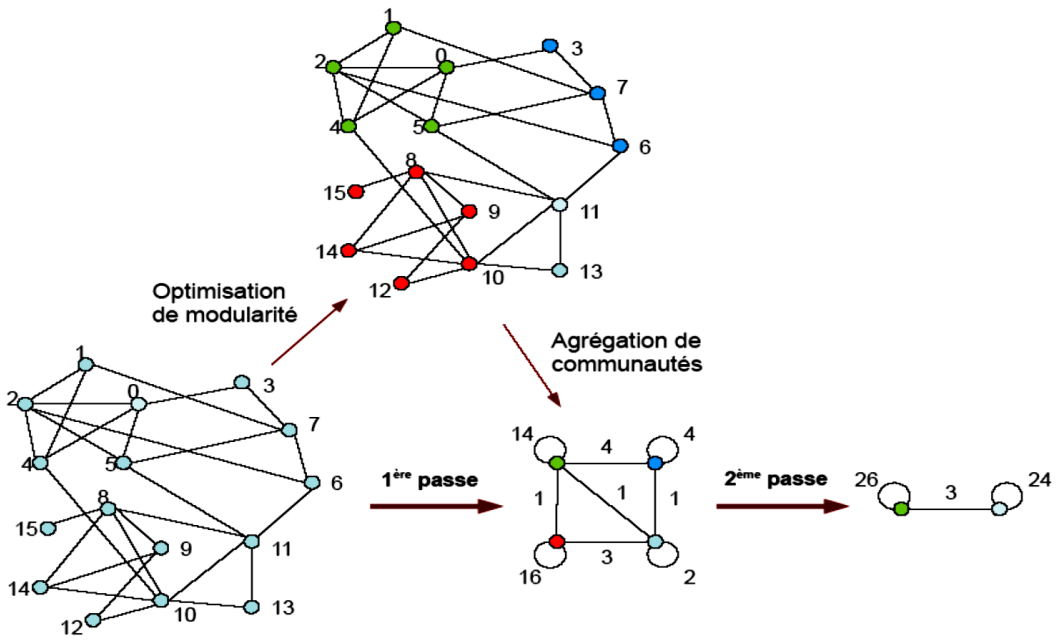

Figure 3. 2 Exemple de fonctionnement de la méthode de Louvain. [6]

Les méthodes hiérarchiques descendantes

Inversement par rapport aux méthodes hiérarchiques ascendantes, les méthodes hiérarchiques descendantes consistent à diviser le graphe (réseau) en plusieurs communautés en éliminant itérativement les arêtes (liens) entre les nœuds. On commence à retirer les arêtes par la seule communauté qui regroupe tous les nœuds. Le processus continue jusqu'à l'obtention des nœuds singletons. Le choix, à chaque fois, de l'arête à retirer ainsi que des nœuds à diviser, représente l'opération la plus importante des méthodes descendantes. Dans cette catégorie, nous rapportons ci-dessous les deux approches dédiées à la détection de communautés.

A- Méthodes basée sur la centralité d'intermédiarité[4] (Edge-Betweenness)

Dans l'article de Girvan-Newman [21] (cf. Algorithme 3.2) l'heuristique appliquée pour le choix de l'arête (lien) à supprimer dans le graphe à chaque itération consiste à choisir le lien dont la centralité d'intermédiarité est maximale, sachant que la centralité d'intermédiarité d'une arête est donnée par la fraction du nombre de plus courts chemins passant par l'arête et reliant n'importe quels couples de nœuds dans le graphe sur le nombre total de plus courts chemins dans le graphe.

[4] *Centralité d'intermédiarité est une mesure de centralité d'un nœud d'un graphe. Elle est égale au nombre de fois que ce nœud est sur le chemin le plus court entre deux autres nœuds quelconques du graphe*

Algorithme 3. 2 Algorithme de Girvan et Newman

Entrée : un graphe non orienté G= (V, E) d'ordre N.
Sortie : P une partition en communautés.
Début
1 : Initialisation : $T = (v_1, \dots, v_N)$, G'=G ;
2 : Calculer la centralité d'intermédiarité pour chaque arête e_i du graphe G'.
// $g_{jk}(e_i)$: *Le nombre de plus courts chemins allant de j à k passant par l'arête* (e_i)
// g_{jk} *: Le nombre total de plus courts chemins allant de j à k.*

$$C^{int}(e_i) = \sum_{j=1}^{N} \sum_{k=1}^{N} \frac{g_{jk}(e_i)}{g_{jk}};$$

3 : Retirer du graphe G' l'arête e_m ayant la plus grande centralité d'intermédiarité.

$$E' = E' \setminus \{e_m \in E' \,|e_m = argmax_{e_{j \in E'}} C^{int}(e_j)\};$$

4 : Identifier l'ensemble C= $(C_i, \dots, C_l)$ de toutes les composantes connexes du graphe G' ;
5 : **Si** C∉ T **alors** rajouter C à T ;
6 : **Si** |E'|≠ 0 **alors** aller à l'étape 2 ;
7 : Retourner la partition P ayant la plus grande modularité

$$P = argmax\ T_{i \epsilon T}\ \{modularité(G, T_i)\};$$

Fin

B- Méthodes basées sur le clustering d'arêtes

Radicchi et al dans leur article [43] proposent un coefficient de clustering d'arête qui est définie comme étant le nombre de triangles construits par cette arête, divisée par le nombre maximum de triangles possibles. Cet algorithme retire donc, à chaque étape, l'arête de plus faible clustering. L'avantage de cet algorithme est que le calcul de ce coefficient requiert des calculs locaux seulement (seul le coefficient de clustering des arêtes voisines sera recalculé), ce qui lui permet d'être plus rapide que l'algorithme de centralité d'intermédiarité. L'algorithme est moins coûteux en temps, mais ne donne pas de résultats assez satisfaisants et sa complexité algorithmique est de $O(n^2)$.

3.2.2.2 Autres méthodes

A- Méthodes basées sur la propagation d'étiquettes (Label Propagation)

Raghavan et Albert proposent [44] un algorithme LP pour détecter des communautés. L'idée maîtresse de l'algorithme est d'initialiser chaque nœud avec un label

unique, par la suite, chaque nœud remplace son label par celui utilisé par la majorité de ses voisins (ou un nœud choisi au hasard en cas d'égalités) de manière successive. Après un processus itératif, la même étiquette peut être associée aux membres connectés pour former une communauté. De même Gregory et al. [22] proposent COPRA appliqué aux communautés avec recouvrement. Chaque nœud maintient une liste des labels les plus courants dans son entourage et non pas le label le plus courant chez ses voisins. Un paramètre de l'algorithme fixe le nombre maximum de labels qu'un nœud peut retenir pour qu'un label ne s'étende pas à l'infini.

B- Méthodes utilisant les marches aléatoires

Le principe des marches aléatoires est basé sur l'idée qu'un promeneur (marcheur) est affecté à un nœud du graphe, et peut à chaque étape se déplacer vers un des autres nœuds voisins. Rosvall et Bergstrom [45] ont proposé InfoMap qui utilise des méthodes venant de la théorie de l'information, et cherche à trouver un encodage optimal de graphe. Pour ce faire, chaque nœud est décrit par :

- Un identifiant de la communauté à laquelle il appartient.
- Un identifiant unique au sein de cette communauté.

On peut décrire le déplacement d'un marcheur en commençant à donner l'identifiant de la communauté auquel il appartient puis l'identifiant du nœud sur lequel il est. Si le marcheur se déplace à un nœud au sein de sa communauté, alors on peut décrire le nouveau nœud qu'il atteint avec son identifiant. Dans le cas contraire, il faut donner l'identifiant de la nouvelle communauté puis l'identifiant du nœud au sein de cette nouvelle communauté. Finalement, l'algorithme cherchant à minimiser la longueur de la description du chemin parcouru par un marcheur aléatoire et rendant les déplacements de ce dernier les plus rares possible, donne un bon découpage en communautés.

Par conséquent, si le graphe étudié est un graphe aléatoire alors le marcheur aléatoire va avoir tendance à rester à l'intérieur des communautés, hors, un marcheur aléatoire ne restera pas "coincé" dans une communauté. InfoMap sera donc capable de dire que le graphe n'est pas divisible en communautés pertinentes, contrairement aux méthodes basées sur une optimisation de la modularité qui trouvent toujours des communautés quel que soit le graphe étudié. Le fonctionnement d'InfoMap est illustré par la figure 3.3.

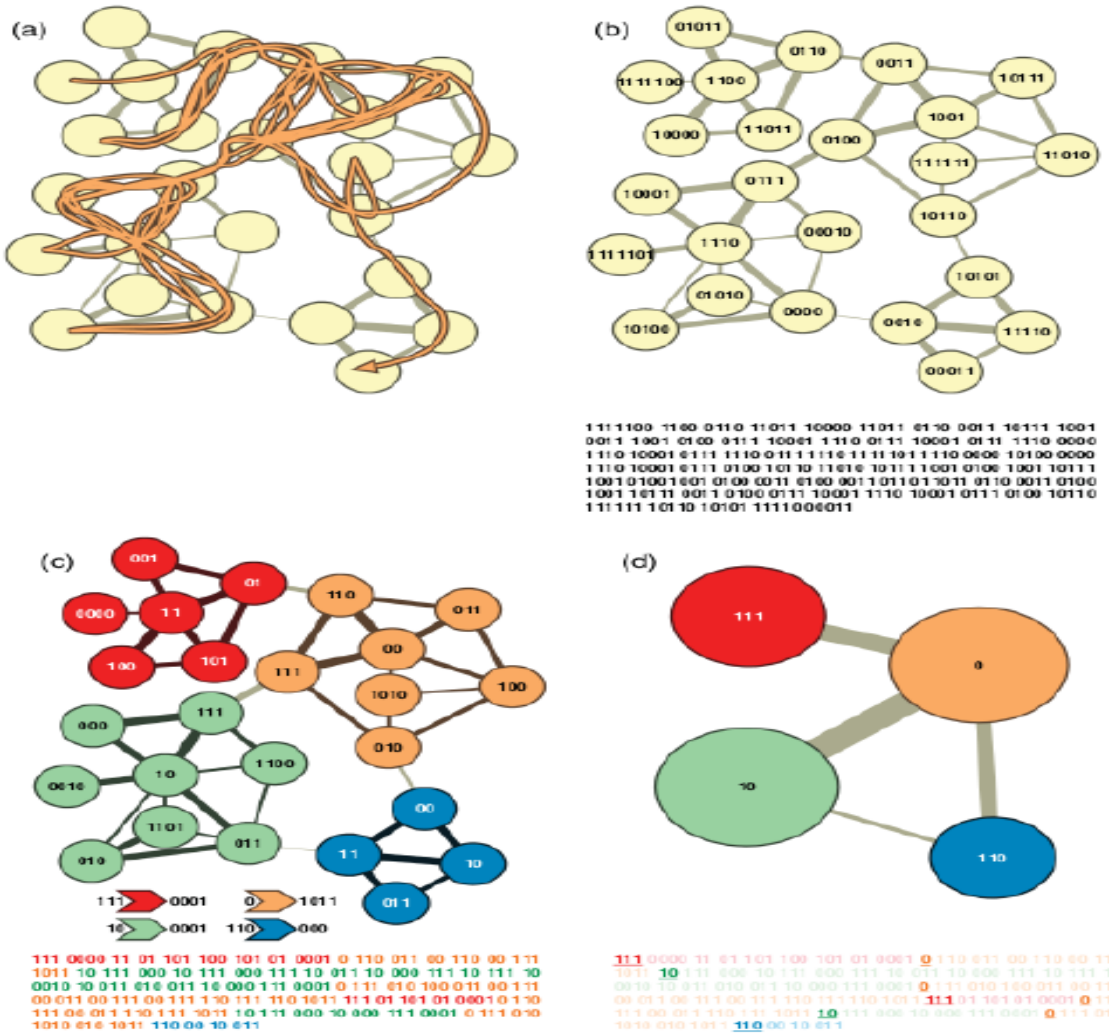

Figure 3. 3 Illustration de fonctionnement d'InfoMap[5]. [45]

[5] *-Image A : Représente le parcours d'un marcheur aléatoire que l'on souhaite encoder.*
-Image B : Représente l'encodage : chaque nœud est identifié par une suite binaire et le parcours, (en noir en-dessus) qui utilise la séquence des identifiants des nœuds.
-Image C : Optimisation de l'encodage en identifiant les groupes et les nœuds.
-Image D : Visualisation des groupes sous forme d'un graphe.

C- Méthodes basées sur les cliques

L'idée de ces approches est que, à partir de k-cliques, on construit petit à petit des communautés. Une communauté est définie comme une chaîne de k-cliques adjacentes. Une k-clique est un sous-ensemble de k nœuds tous adjacents les uns aux autres, et deux k-cliques sont adjacentes si elles partagent k-1 nœuds. Mais il y a des cas où un nœud peut appartenir à plusieurs k-cliques non forcément adjacentes. L'algorithme de Palla et al présenté dans [42] nommé CFinder repose sur trois principales étapes pour découvrir les communautés :

1) Calculer l'ensemble de cliques de taille k dans le graphe cible G. k est un paramètre de l'algorithme.

2) Construire un graphe de cliques où chaque clique est représentée par un nœud. Deux nœuds sont connectés par un lien si les deux cliques associées partagent k-1 nœuds dans le graphe G.

3) Les communautés dans le graphe G sont alors les composantes connexes identifiées dans le graphe de cliques construit à l'étape 2.

Une limite de cet algorithme est qu'il nécessite un paramétrage : la valeur de k (la taille des communautés à considérer).

3.2.3 Détection de communautés dynamiques

Toutes les méthodes que nous venons de présenter à la section précédente s'appliquent à des réseaux statiques. Or, dans la réalité, les réseaux sont dynamiques et évoluent au cours du temps.

Dans cette partie, nous allons présenter les méthodes de détection de communautés dynamiques classées selon la hiérarchie suivante :

- Méthodes utilisant les algorithmes statiques pour le suivi
  - Méthodes basées sur la métrique de matching
  - Méthodes à base des nœuds cœurs
  - Méthodes basées sur le graphe union
- Méthodes par détection directe des communautés dynamiques
- Méthodes incrémentales

-Méthodes basées sur la fonction de coût

-Méthodes basées sur l'optimisation de la modularité

-Méthodes basées sur la propagation d'étiquettes

-Autres méthodes

3.2.3.1 Méthodes utilisant les algorithmes statiques pour le suivi

Il est naturel de commencer à utiliser les solutions de détection de communautés statiques pour résoudre les problèmes de communautés dynamiques. C'est pourquoi il existe déjà plusieurs tentatives d'utilisation des algorithmes statiques. L'idée générale de ces approches est de diviser le réseau dynamique en une série d'instantanés qui sont tous des graphes statiques ensuite la détection se fait en deux temps : une première étape consiste à appliquer un algorithme statique sur chacun de ces instantanés, ce qui permet d'obtenir une série de partitions, une pour chaque instantané. Ensuite, pour analyser la dynamique, il faut trouver, pour chaque communauté de l'instant t, ce qu'elle est devenue à l'instant t+1. Elle peut ne pas avoir changé, avoir disparu, s'être séparé en plusieurs groupes ou avoir fusionné avec d'autres communautés, donc il faut trouver une correspondance (association) entre les communautés existant dans des instantanés consécutifs.

La figure 3.4 est un exemple d'une communauté qui s'évalue à trois instantanés. L'exemple montre comment ce que fait l'association entre les communautés dynamiques et leurs chemins d'évolution. Nous observons 4 communautés dynamiques au total durant les trois instants indiqués par les couleurs suivantes : C1 (bleu foncé), C2 (rouge), C3 (verte), C4 (bleu clair). L'évolution de ces communautés est exprimée par :

Evolution(C1) ← {C1(t), C1(t + 1), C1(t + 2)}

Evolution(C2) ← {C2(t + 1), C2(t + 2)}

Evolution(C3) ← {C3(t), C3(t + 1), C3(t + 2)}

Evolution(C4) ← {C4(t + 2)}

Durant cette évolution, on observe deux communautés qui apparaissent : la communauté C2 qui est la branche de C1 et la communauté C2 qui émerge à l'instant t+2.

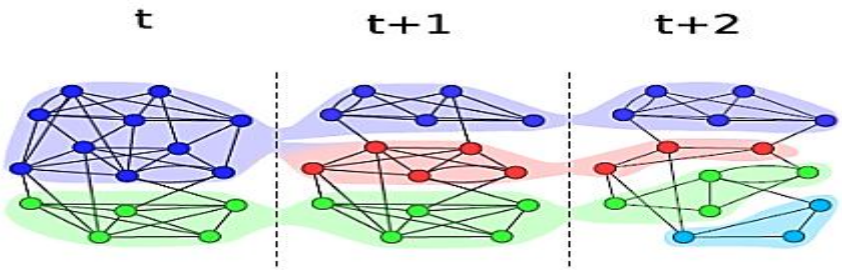

Figure 3. 4 Exemple de trois instantanés d'un réseau avec une association entre les communautés des différentes étapes. [51]

A- Méthodes basées sur la métrique de matching

La première méthode la plus classique repose sur des considérations ensemblistes sur la taille des intersections.

Hopcroft et al ont défini dans leur article [24] une méthode qui utilise la taille de l'intersection entre deux communautés successives à l'instant t et t+1. Les auteurs observent les changements entre ces deux instantanés en calculant une valeur d'appariement (matching) entre les communautés. Cet appariement est défini par la formule (3.5) comme suit :

$$match(c_1, c_2) = \min\left(\frac{|c_1 \cap c_2|}{|c_1|}, \frac{|c_1 \cap c_2|}{|c_2|}\right) \qquad (3.5)$$

Sachant que :

- la valeur de match est comprise entre 0 et 1.

-Le match vaut 1 si les communautés sont égales, 0 si elles sont différentes.

-Plus les communautés sont proches, plus l'intersection est grande.

-c_1 est la communauté à l'instant t.

-c_2 est la communauté correspondante à c_1 à l'instant t+1.

Un gros problème d'instabilité de détection existe et Hopcroft et al, pour le résoudre, décident de restreindre les communautés qui sont grandement modifiées entre deux pas de temps, et de ne considérer que les communautés stables définies comme des communautés qui même si l'on introduit un changement mineur du réseau, restent identiques. Bien que cette restriction élimine de nombreuses communautés intéressantes, elle a permis une première analyse de communautés dynamiques.

Dans [48], les auteurs définissent match (C_1) comme la communauté à t + 1 dont l'intersection est la plus grande qu'une limite donnée. Si une telle communauté

n'existe pas, car les intersections sont toutes trop petites, alors match (C_1) = ∅. Ils utilisent ensuite cet ensemble de règles pour définir les événements régissant la vie des communautés :

– C_1 ∈ Pt (Pt une partition à l'instant t) devient C_2 ∈ Pt+1 si C_2 = match (C_1) et ∀C_3 ∈ Pt ≠ C_1; C_2 ≠ match (C_3).

– C_1 ∈ Pt se divise en plusieurs communautés $C_a, C_b, \dots C_k$ ∈ Pt+1 si $\forall_{i \in a,b,\dots,k}$; C_i ∩ C_1 est suffisamment grand et ($C_a \cup C_b \cup \dots \dots \cup C_k$) ∩ C_1 est suffisamment grand.

–C_1 ∈ Pt s'est regroupée avec d'autres communautés pour former C_2 ∈ Pt+1 si C_2 = match (C_1) et ∀Z ∈ Pt ≠ C_1; C_2 = match(Z).

– C_1 ∈ Pt disparaître, si aucune des règles précédentes ne s'applique pas.

– C_2 ∈ Pt+1 apparaître, si ∀C_1 ∈ Pt ; C_2 ≠match (C_1).

Ces règles sont assez faciles à comprendre, mais ne sont pas vraiment satisfaisantes. Aussi, une intersection est "suffisamment grande" quand la taille de chaque communauté représente n%, mais comment fixer n ? Est-ce que deux communautés sont similaires quand elles partagent 50% des nœuds, 70% ou 90% ? Donc trouver un ensemble minimal et consensuel de règles semble impossible en pratique et les paramètres qu'ils impliquent ne peuvent pas être fixés sans connaissances à priori sur l'évolution des communautés. La figure 3.5 montre un exemple de la métrique de matching.

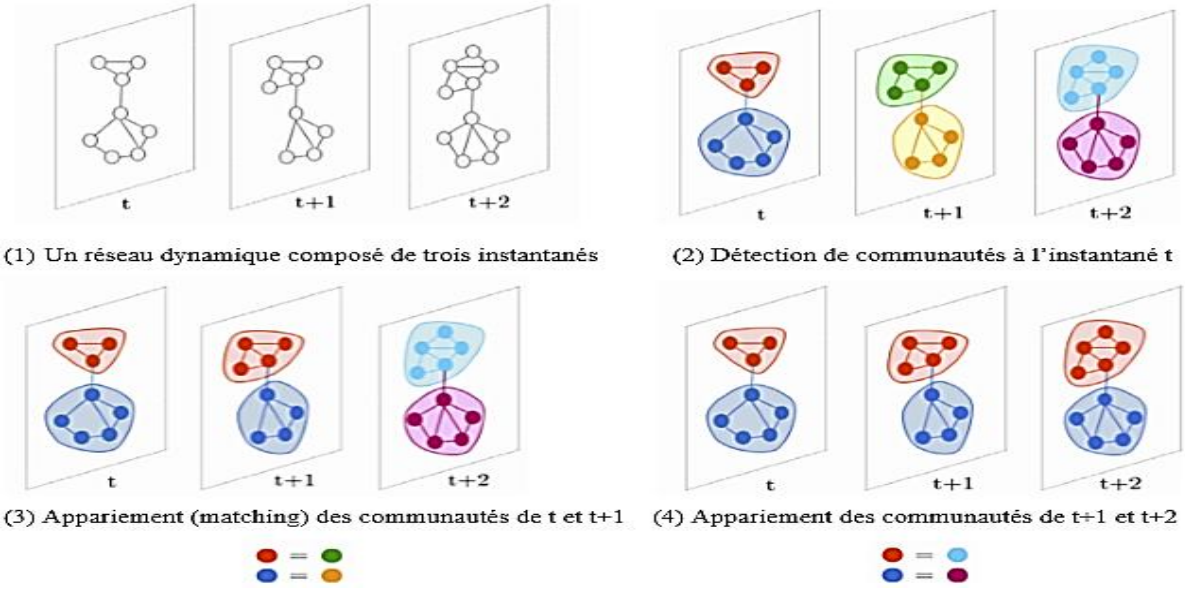

Figure 3. 5 Illustration de la métrique de matching (appariement). [16]

Dans [8] les auteurs ont mis au point une nouvelle technique appelée CED (Community Evolution Detection) afin de suivre et détecter l'évolution de la communauté sur les réseaux sociaux. La méthode mise en correspondance (matching) des communautés à partir d'instantanés consécutifs lors de l'identification de l'événement (naissance, mort, croissance, contraction, fusion, division). Elle dépend des nœuds clés (nœuds essentiels dans chaque communauté) et de la métrique QuantityInsertion. Cette métrique permet d'évaluer l'inclusion d'une communauté dans une autre. En d'autres termes, combien de membres de la communauté $C_i^{(t)}$ sont dans la communauté $C_j^{(t+1)}$. Par conséquent, la quantité $QI(C1\,,C2)$ de la communauté $C1$ dans la communauté $C2$ est calculée comme montré par l'équation (3.6) suivante :

$$QI\left(C_i^{(t)}, C_j^{(t+1)}\right) = \frac{\left|C_i^{(t)} \cap C_j^{(t+1)}\right|}{\left|C_i^{(t)}\right|} \tag{3.6}$$

B- Méthodes à base des nœuds cœurs

Les auteurs Wang et al [52] utilisent le même principe que Hopcroft et al, mais au lieu de calculer le match entre les communautés, ils proposent de les identifier à l'aide des nœuds cœurs. Ces nœuds cœurs sont considérés comme le centre de chaque communauté et suivre l'évolution de celle-ci revient à observer le comportement de ces nœuds cœurs. Sélectionner les nœuds cœurs dans chaque communauté est un problème à cause des auteurs qui les définissent de différentes manières. Wang et al [52] les voient comme les nœuds v qui vérifient $\sum_{n\,\epsilon\,voisins} degree(v) - degree(n) > 0$, tandis que dans [4] ils correspondent aux nœuds dont le degré est supérieur à une valeur k donnée.

Pour rendre la détection de communautés simples, Chen et al [13] définissent les communautés comme étant des cliques maximales du réseau et pour suivre les communautés d'un instant à l'autre, ils identifient des nœuds cœurs qui permettent de réduire le nombre de communautés à considérer. La faiblesse de cette méthode est qu'elle définit les communautés comme étant des cliques donnant souvent des communautés non-pertinentes et conduisant à un nombre important de communautés sur les grands graphes.

Les auteurs dans [58] utilisent aussi la notion des nœuds cœurs appelés les nœuds cœurs chaînés pour suivre les communautés, mais d'une manière un peu

différente. Les communautés sont initialisées par la construction des nœuds cœurs chaînés, sachant que ces nœuds sont recherchés et séparés en communautés qui leur correspondent par un algorithme de détection statique. Pour détecter les communautés dynamiques et leurs évolutions, les auteurs proposent de juger l'évolution des communautés par le suivi de changement de la situation des nœuds cœurs dans chaque communauté.

a) Pour ce faire Yin et al utilisent l'évolution de galaxie dans l'univers comme référence, qui essaie de fixer des étoiles ayant un rôle important pour attirer les étoiles ayant une grande masse pour entrer dans la galaxie. Ce sont les nœuds cœurs qui jouent le rôle de ces étoiles. Deux nœuds cœurs connectés peuvent appartenir à une seule communauté ou à des communautés différentes. Les nœuds cœurs qui se connectent ensemble et qui contiennent des branches sont appelés "des nœuds cœurs chaînés".

b) Après la construction des communautés initiales, l'algorithme passe à la détection des communautés statiques, il prend en considération les nœuds non-cœurs et établit une matrice de relation de gravité entre ses nœuds puis test si ces nœuds sont attirés par une communauté sous l'action de gravité.

c) Pour les réseaux dynamiques, les nœuds et les liens peuvent changer d'un instant à l'autre, donc les communautés évoluent entre deux instants adjacents et les auteurs essaient d'analyser cette évolution par le suivi des nœuds cœurs.

Bien que les approches à base des nœuds cœurs peuvent résoudre le désaccord causé par les nœuds secondaires. Mais aussi, on peut perdre des changements structurels importants qui sont liés à des nœuds secondaires lors de suivi des communautés.

C- Méthodes basées sur le graphe union

Les auteurs de [41], créent un graphe union U (figure 3.6) contenant les liens et les nœuds des communautés à l'instant t et t+1. Les auteurs garantissent que les communautés à l'instant t et t+1 seront intégralement contenues dans les communautés de l'union, sachant que chaque communauté de t et de t+1 sera contenue dans une et une seule communauté de U.

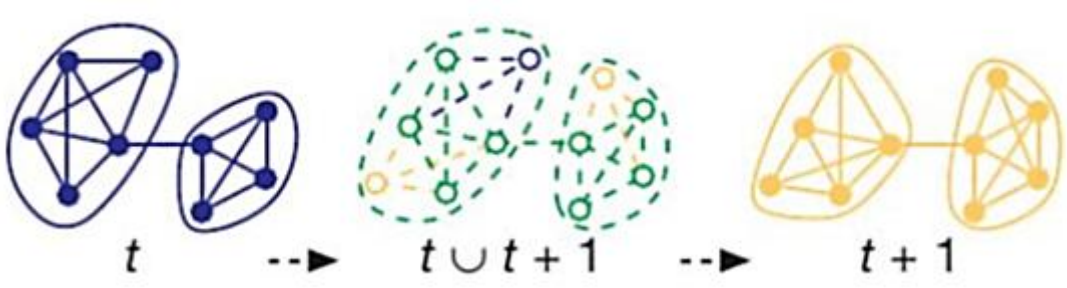

Figure 3. 6 Regroupement des graphes à deux instants pour trouver l'évolution des communautés entre t et t + 1. [41]

Lorsque des liens sont ajoutés au réseau, les communautés ne peuvent que grossir, se regrouper ou rester inchangées. Cette propriété assure que les communautés détectées sur l'union des instantanés ne contiendront que des communautés entières parmi celles de t ou t+1. Par conséquent, quand on détecte des communautés sur le graphe union, on peut clairement décider de mettre en relation des communautés des instants t et t+1. Il est possible que les communautés sur le graphe union ne contiennent que des parties de communautés à t ou t+1, ce qui ne garantit pas la détection de toutes les communautés intéressantes dans les réseaux complexes voir la figure 3.7.

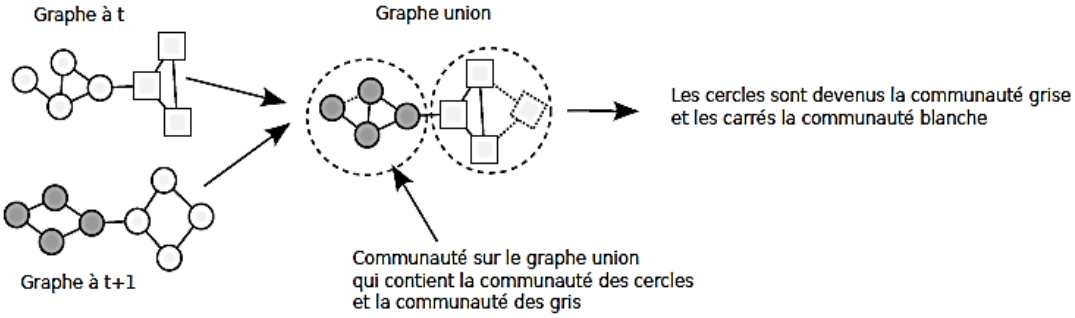

Figure 3. 7 Exemple d'un graphe union qui est constitué par le regroupement de deux graphes à l'instant t et t+1. [41]

Pour conclure nous dirons que les méthodes présentées dans cette partie possèdent deux phases :

1- Les communautés sont détectées sur chaque instantané indépendamment des autres instantanés.

2- Les relations entre les communautés à chaque instantané sont déduites successivement.

Pour réaliser ces deux phases [24], [48], [8] utilisent la métrique de matching comme une mesure de similarité, et [52], [4], [58] introduisent une méthode basée

sur les nœuds cœurs, et enfin [41] utilisent les méthodes basées sur le graphe union.

Le processus de cette approche convient aux réseaux ayant une structure de communauté hautement dynamique et non ambiguë. Il présente les avantages suivants :

- Premièrement, à l'étape de la détection de communauté dans des instantanés indépendants, les méthodes de détection de communauté traditionnelles peuvent être réutilisées.
- Il est également possible d'utiliser les mesures existantes pour assortir (matching) les communautés.

Un inconvénient majeur de cette approche est que les algorithmes de détection de communautés sont instables. Ce manque de stabilité résulte du fait que les algorithmes peuvent donner des résultats très différents pour deux réseaux presque similaires. Aussi une modification mineure du réseau conduit à une grande transformation des communautés, ce qui rend cette technique peu efficace. Certains travaux ont tenté de trouver des solutions pour résoudre ce problème en considérant la partie la plus stable de la communauté (nœuds cœurs) et en ignorant les nœuds qui changent fréquemment d'appartenance. Néanmoins, ce problème reste important et toutes les autres approches tentent de le surmonter.

3.2.3.2 Méthodes par détection directe des communautés dynamiques

Les approches de cette section utilisent directement la dynamique dans la décomposition et non d'étudier une succession de réseaux statiques puisque les solutions statiques n'étaient pas satisfaisantes. L'idée de ces approches est de redéfinir les communautés comme des objets dynamiques.

Au lieu d'extraire des communautés pour chaque pas de temps et de les faire correspondre, [11] a utilisé deux paramètres d'optimisation : la qualité des instantanés pour mesurer la qualité du clustering sur le pas de temps actuel et la qualité de l'historique pour calculer la similarité entre le clustering actuel et le précédent. Les deux paramètres d'optimisation sont regroupés dans une fonction de qualité Q exprimée par l'équation (3.7) suivante :

$$Q = Q_{instant} + \alpha Q_{stabilité} \qquad (3.7)$$

Où $Q_{instant}$ est une fonction de qualité statique, $Q_{stabilité}$ est un terme qui évalue la distance entre la nouvelle partition et la précédente, et α un paramètre caractérisant l'importance de la stabilité. Ainsi, la structure de la communauté et son développement ont été contrôlés en même temps afin de permettre la régularité temporelle. Cette nouvelle fonction de qualité permet d'obtenir une série de partitions plus intéressantes et Chan et al [12] utilisent la même idée.

Xu et al [56] utilisent une idée proche de la précédente en tenant compte de la matrice d'adjacence W et non pas de la fonction de qualité qui est définie par : $W = \alpha W_t + (1-\alpha)W_{t+1}$ .Les auteurs donc cherchent à maximiser la qualité du graphe et trouver le meilleur α.

Les approches modèles ont été utilisées pour la détection directe des communautés dynamiques qui consistent à décrire un modèle aléatoire de génération de graphes ayant un certain nombre de paramètres ensuite à trouver les paramètres qui permettent de mieux rendre compte des données. Parmi ces paramètres, par exemple en ayant des probabilités différentes de relier des nœuds selon leurs communautés.

Dans l'article [57] Yang et al utilisent le modèle de block aléatoire DSBM "dynamic stochastic block model". Dans ce modèle, les auteurs essaient d'assigner pour chaque nœud i, une communauté k avec une probabilité π_k et pour chaque paire de nœuds i, j appartenant à des communautés k, m respectivement, des liens sont ajoutés selon une loi de probabilité. Cette approche semble intéressante mais n'est applicable que sur de petits réseaux.
Nous concluons en disant que dans les approches de détection directe des communautés dynamiques, les auteurs de [11], [12] utilisent des fonctions de qualité. L'utilisation de ces dernières nécessite des outils de validation (réseaux de tests avec une meilleure décomposition) ce qui n'existe pas dans le cas dynamique. Yang et al [57] faisant référence à un modèle aléatoire à de fortes bases théoriques, mais elle est difficile à appréhender par les utilisateurs à cause de ses technicités, ce qui ralentit leur choix.

3.2.3.3 Méthodes incrémentales

Les méthodes incrémentales utilisent l'information de l'instant précédent pour détecter les communautés à l'instant courant. Lorsque la structure de réseau

change, l'approche n'utilise pas uniquement l'information de la structure du graphe courant, mais aussi l'information de la structure de la communauté détectée à l'instant précédent. En plus, les approches incrémentales possèdent souvent naturellement une mémoire de ce qui s'est passé, et donc tiennent au moins partiellement compte de l'historique et de la dynamique. Le principe de ces approches est illustré dans la figure 3.8.

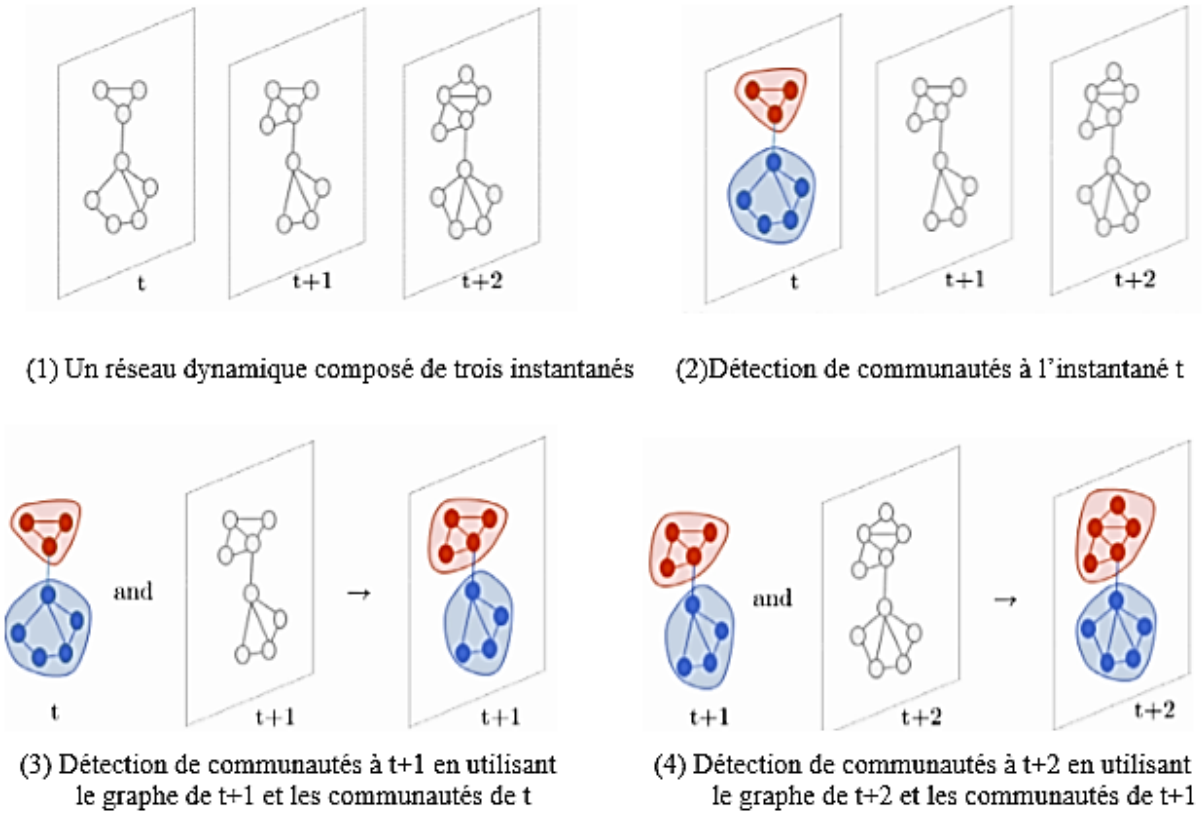

Figure 3. 8 Illustration de fonctionnement des approches incrémentales. [16]

A- Méthodes basées sur la fonction de coût

Dans les méthodes basées sur la fonction de coût, une alternative est faite pour trouver les communautés à chaque pas de temps en considérant le fait que la structure communautaire courante est similaire à la structure communautaire précédente.

Les auteurs dans [33] souhaitent analyser les communautés et leurs évolutions dans un processus unifié. Pour atteindre cet objectif, ils proposent d'utiliser la structure communautaire au temps t-1 pour régulariser la structure communautaire courante à t. Pour incorporer une telle réglementation, ils introduisent une

fonction de coût ($Cost$) qui est une combinaison de snapshot cost et past history cost. Cette fonction est exprimée par l'équation (3.8) suivante :

$$Cost = \alpha CS + (1 - \alpha)CT \quad (3.8)$$

Dans cette fonction de coût, CS mesure la correspondance entre la structure communautaire et les interactions graphiques au temps t. Le coût historique CT qualifie la cohérence de la structure communautaire avec celle à l'instant t-1. La partition la plus valable est celle qui a un grand CS et un petit CT. Le paramètre α est défini par l'utilisateur pour contrôler le niveau d'emphase sur chaque partie du coût total. Ce dernier est une limitation majeure puisqu'il est au préalable inconnu.

Tantipathananandh et Berger-Wolf proposent une nouvelle fonction de coût [49] qui consiste en trois parties :

α-cost : Le coût des nœuds qui changent leurs communautés entre deux instants.

ß-cost : Le coût de deux nœuds appartenant à une seule communauté, mais qui ne sont pas connectés.

γ-cost : Le coût de deux nœuds appartenant à des communautés différentes mais qui se connectent ensemble.

Les auteurs par la suite utilisent un algorithme qui minimise ses trois fonctions de coût.

B- Méthodes basées sur l'optimisation de la modularité

Shang et al [47] proposent un algorithme de détection en temps réel pour suivre la structure communautaire dans des réseaux dynamiques. Les auteurs adoptent deux étapes : premièrement, ils appliquent l'algorithme de Blondel et al (GBL) [6] pour générer la structure communautaire initiale du réseau ensuite ils appliquent leur incrémentale stratégie de mise à jour pour suivre les communautés dynamiques. Les auteurs classifient quatre types d'arêtes à ajouter et effectuent des modifications aux communautés de manière à augmenter la modularité si c'est possible. Les quatre types d'arêtes sont :

(1) Une arête à l'intérieur de la communauté (ICE).

(2) Une arête qui combine deux communautés (CCE).

(3) Une nouvelle arête moitié (HNE).

(4) Une nouvelle arête entre deux nouveaux nœuds (NE).

Et les opérations (modifications) associées aux arêtes comprennent :

(opt1) : La structure communautaire reste inchangée.

(opt2) : La combinaison de deux communautés en une seule communauté

(opt3) : L'ajout d'une arête entre un nouveau nœud et une communauté existante.

(opt4) : La création d'une nouvelle communauté avec des nouveaux nœuds.

Les auteurs gardent l'opération qui donne un gain de modularité selon des hypothèses sur l'évolution du réseau, par exemple s'ils s'intéressent à l'ajout d'une nouvelle communauté, les opérations 3 et 4 le réalisent, mais pour le choix d'entre eux Shang et al calculent la modularité des deux opérations et choisissent celle qui donne un gain de modularité. L'algorithme incrémental de Shang et al est donné par l'algorithme 3.3.

Algorithme 3. 3 Algorithme de Shang et al

Initialisation : Exécution de l'algorithme GBL pour générer la structure communautaire initiale.

Pour *i* := 1 **au** nombre d'arêtes à ajouter **faire**
  Switch Type d' (arête *i*)
  Case : ICE
    La structure communautaire reste inchangée.
  Case : CCE
    Si le gain (opt2)>gain (opt1)
      Combiner les communautés avec opt2.
      Sinon la structure communautaire reste inchangée.
    Fin si
  Case : HNE
    Mettre à jour la structure avec opt3.
  Case : NE
  Mettre à jour la structure avec l'opt4.
Fin

Les auteurs dans [15] utilisent une méthode qui est une modification de celle de Louvain où les arêtes sont ajoutées/supprimées dynamiquement. Ils accomplissent [47] par l'opération de suppression d'arêtes. Dans chaque itération, l'algorithme maintient toutes les communautés inchangées qui n'étaient pas affectées par des modifications apportées au réseau. Ce maintien est réalisé par la réutilisation de la structure communautaire obtenue à l'itération précédente. L'algorithme proposé par Cordeiro et al optimise la modularité localement (maximise le gain de la modularité, uniquement pour les communautés qui ont des arêtes à additionner/retirer).

Les auteurs classifient quatre types d'arêtes à ajouter :

-Une arête à l'intérieur d'une communauté.

-Une arête entre deux communautés.

-Une arête entre un nœud et une communauté.

-Une arête entre deux nouveaux nœuds.

De même, ils attribuent quatre types d'arêtes à supprimer :

- À l'intérieur d'une communauté.

- Entre deux communautés.

- Entre une communauté et un nœud isolé.

- Entre deux nœuds isolés.

La figure 3.9 est un exemple qui montre l'évolution d'un réseau sur trois instants. On remarque un lien qui est ajouté (1 - 4) à l'instant T+1 et un autre nouveau lien (5 - 7) à l'instant T+2. Pour mettre à jour les communautés détectées, les auteurs utilisent deux types de réseaux R1 et R2 :

R1 : maintient le réseau original.

R2 : stocke l'ensemble des communautés dans le réseau.

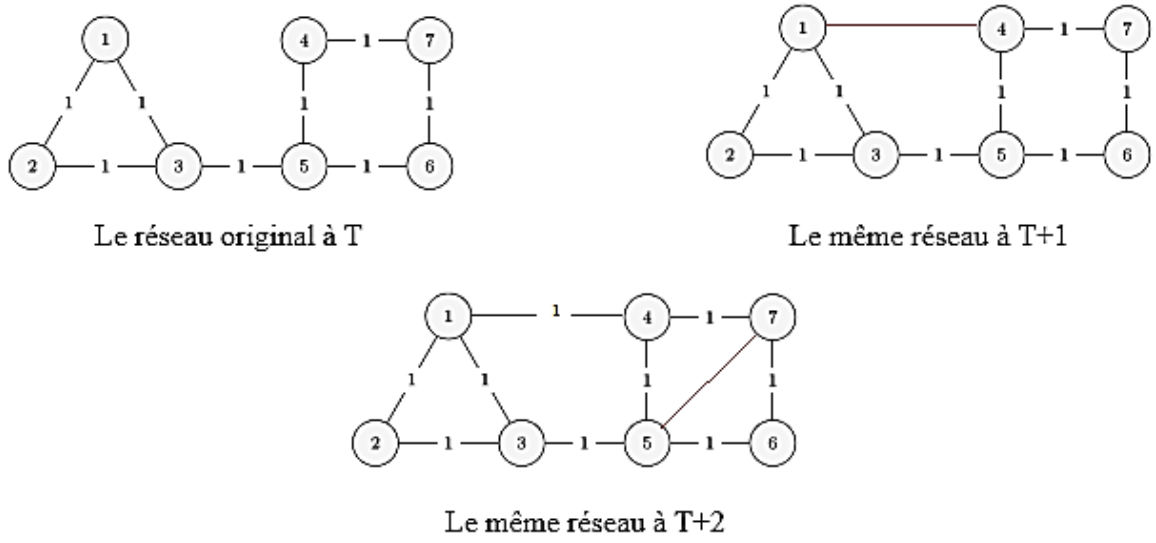

Figure 3. 9 Evolution d'un réseau sur trois instants.

L'algorithme commence à détecter les communautés à l'instant T, en utilisant l'algorithme de Louvain. Le réseau évolue à T+1 par l'ajout d'arête et l'algorithme prend en considération les deux réseaux R1 et R2 pour effectuer une optimisation locale de la modularité aux communautés touchées par le changement.

L'opération d'ajout d'arête (1 - 4) est réalisée après la première itération de Louvain. Les étapes de l'algorithme sont illustrées dans la figure 3.10.

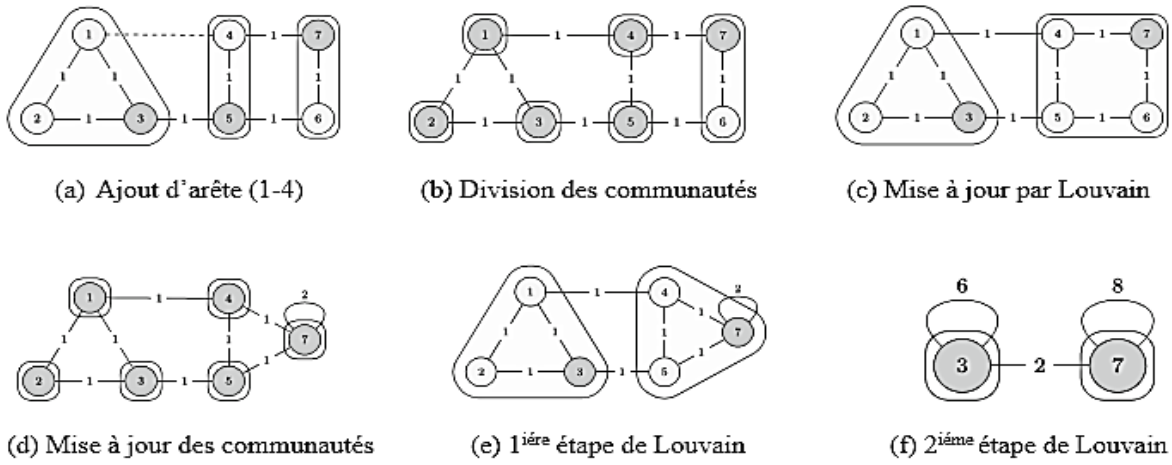

Figure 3. 10 Ajout d'arête (1-4) à l'instant T+1 au réseau original. [15]

-La figure 3.10 (a) : l'ajout d'arête au réseau R1 se fait après la première itération de Louvain.

-La figure 3.10 (b) : division des communautés {(1,2,3) ;(4,5) } affectées par le changement, chaque nœud représente une communauté. La communauté (6,7) reste inchangée.

-La figure 3.10 (c) : mise à jour de R1 avec les nouvelles communautés.

-La figure 3.10 (d) : mise à jour de R2.

-La figure 3.10 (e) : la première étape de Louvain est appliquée et la communauté (4,5) est absorbée par la communauté (6,7) à cause du gain de la modularité.

-La figure 3.10 (f) : représente le R2 final après l'application de la deuxième étape de Louvain.

À l'instant T+2, le processus se répète, l'algorithme prend en considération la structure communautaire à T+1 pour mettre à jour le réseau et détecter les communautés à T+2. La figure 3.11 montre l'exécution de l'algorithme à T+2.

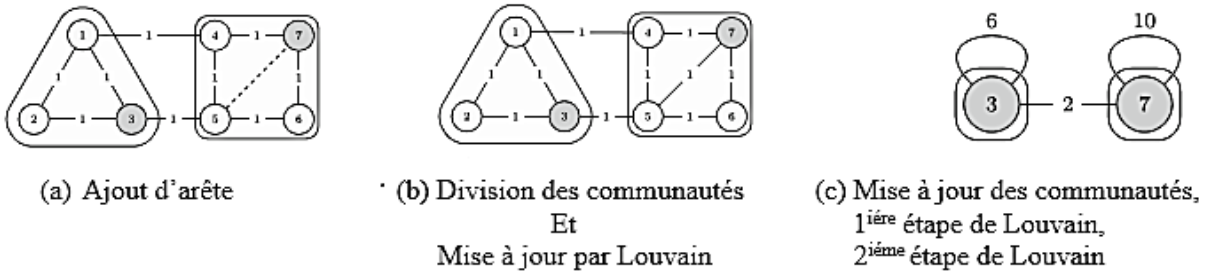

Figure 3. 11 Ajout d'arête (5-7) à l'instant T+2 au réseau de T+1. [15]

Dans [40], un algorithme QCA "the Quick Community Adaptation" a été présenté pour détecter la structure communautaire dans les réseaux sociaux dynamiques. L'algorithme commence par une structure communautaire de base et pour traiter les mises à jour du réseau après les changements qui interviennent (ajout / suppression des nœuds et des liens), il utilise la structure communautaire du réseau identifiée à l'instant précédent. Les mêmes auteurs [39] ont proposé AFOCS un nouvel algorithme de détection de communautés pour les réseaux dynamiques. Cet algorithme partageant les mêmes principes que QCA n'est modifié que pour gérer les communautés chevauchantes.

C- Méthodes basées sur la propagation d'étiquettes

La propagation des étiquettes a également trouvé sa place dans les méthodes incrémentales. LabelRankT proposé dans [54] basé sur le LabelRank généralisé [53] dans lequel chaque nœud n'a besoin que d'informations locales lors du traitement de propagation d'étiquettes. L'idée principale de l'algorithme est d'ajuster la détection quand la structure de réseau change et d'utiliser ce qui est obtenu à l'instant précèdent pour déduire la dynamique à l'instant courant. L'information de la structure locale est encodée dans le nœud distributeur de label et l'évolution de la communauté est capturée par cette distribution. Chaque nœud i diffuse son label P_i (identifiant unique) à ces voisins sur chaque itération et calcule le nouveau label P_i' simultanément en utilisant une équation mathématique. Si un nœud ne change pas (absence de changement des liens) à l'instant T, alors il garde le label qui a lui été affecté à l'instant T-1 ; sinon il reçoit le label de ses voisins. L'algorithme 3.4 montre le fonctionnement de LabelRankT avec une complexité O(m).

Algorithme 3. 4 LabelRankT

Entrée : instantanés du réseau G $([0, 1, \ldots, T])$;

Pour t=1 à T **faire**

a) Suivre le changement des nœuds dans $G(t)$ due ou changement des liens durant t-1 ;

b) Initialiser (le label à l'instant t)P^t ;

//Pour un nœud i qui n'a pas changé durant t-1, $P_i^t = P_i^{t-1}$.

//Pour les nœuds qui changent, réinitialiser le label distributeur.

c) Mettre à jour de manière itérative uniquement la distribution d'étiquettes de nœuds modifiés et les affecter aux communautés correspondantes;

Fin pour ;

Les figures (3.12, 13,14) montrent un exemple de différents événements à trois instants consécutifs. Les nœuds/liens sont ajoutés ou supprimés et les communautés sont fusionnés, divisées ou étendues par LabelRankT.

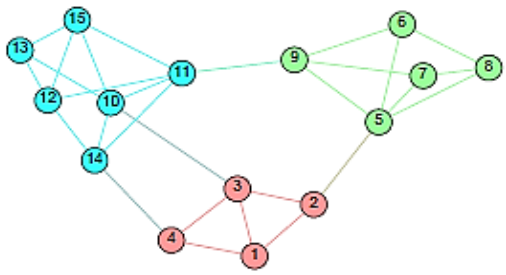

Figure 3. 12 Exemple d'un réseau à t = 0, avec le nombre de nœuds = 15. Les couleurs représentent les communautés découvertes par LabelRankT. [54]

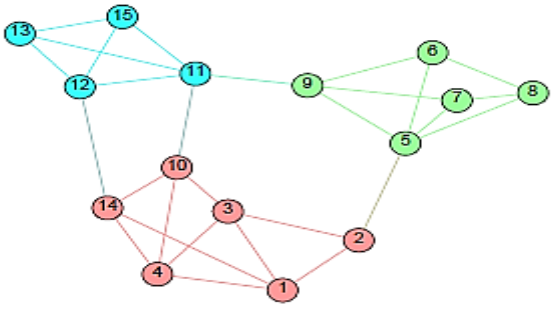

Figure 3. 13 [6]Le même réseau à t = 1, avec le nombre de nœuds = 15. [54]

[6] *Les nœuds 10 et 14 sont déplacés vers la communauté rouge.*
Trois liens sont supprimés (14-11 ,10-13,10-15) et trois autre liens sont ajoutés (13-11 ,14-1 ,10-4). Les couleurs représentent les communautés détectées par LabelRankT.

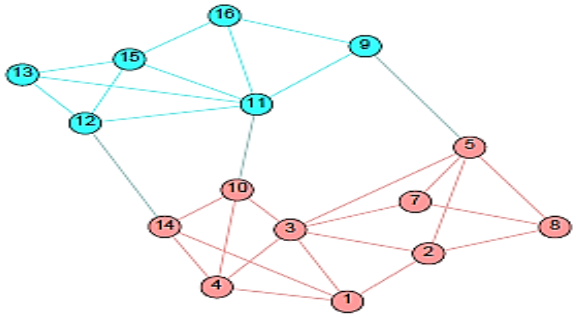

Figure 3. 14 [7]Le même réseau à t = 2, avec le nombre de nœuds =15. [54]

LabelRankT est plus rapide que les autres algorithmes de détection dynamiques comme FACETNET [32] et ILCD [10]. Il peut être intégré avec SLPA [55] afin de détecter les communautés chevauchantes. Cette méthode sera utilisée dans le chapitre expérimentation pour comparer ses résultats aux résultats de notre algorithme.

[23] propose un algorithme adaptatif de propagation d'étiquettes (ALPA) pour analyser les communautés dynamiques sans avoir besoin des informations préalables des communautés ou des paramètres définis par l'utilisateur. ALPA utilise la technique de propagation d'étiquettes de [44] (cf. section 3.2.2.2 A) pour ne propager les étiquettes que dans une partie du réseau. Il gère une liste de nœuds actifs contenant tous les nœuds actifs et termine son exécution lorsque la liste est vide. Un nœud actif est celui dont le libellé n'est pas majoritaire parmi ses voisins et qui le modifie éventuellement s'il tentait une mise à jour. Cependant l'algorithme est non déterministe, son exécution donne différente partitions pour le même réseau contrairement à LabelRankT qui est déterministe.

D- Autres méthodes

Li et al [31] proposent CDBIA "A dynamic community detection method based on incremental Analysis" basé sur le point suivant : la plupart des communautés ont tendance à évoluer graduellement dans le temps, et non pas apparaître ou disparaître rapidement. Les auteurs analysent les opérations qui effectuent des changements au réseau et les nœuds affectés par ces transformations peuvent changer

[7]*Le nœud 6 est retiré et le nœud 16 est ajouté. Les membres de la communauté verte sont déplacés vers les communautés bleue et rouge. Les couleurs représentent les communautés détectées par LabelRankT.*

de communauté par rapport à l'étape précédente. Les opérations qui peuvent intervenir sur le réseau sont :

1) Ajout d'arête.

2) Suppression d'arête.

3) Déplacement d'un nœud vers une autre communauté.

4) Ajout et suppression d'un outlier.

Chaque opération est nommée "Incrément" et la méthode se base sur l'analyse de ces incréments. Par exemple si un réseau change d'un instant T à l'instant T+1 par l'ajout d'arête (a, b) (1) sachant que l'étiquette de la communauté contenant le nœud a est $label(a) = p$ et l'étiquette de la communauté contenant le nœud b est $label(b) = q$. Les auteurs testent si $p = q$ alors les nœuds se trouvent dans la même communauté et les degrés des nœuds augmentent au sein de leurs communautés. Dans le cas contraire, si la différence entre le degré d'un nœud au sein de sa communauté et le degré du même nœud au sein d'une autre communauté est supérieure à un paramètre de l'algorithme, alors le nœud se déplace vers cette communauté. Les auteurs prennent aussi en considération les autres nœuds influencés après l'ajout de l'arête (a , b) au réseau.

Dans leur travail, Cazabet et al [10] ont proposé l'algorithme iLCD (intrinsic Longitudinal Community Detection) qui prend en compte la dynamique des réseaux et les communautés chevauchantes. Cet algorithme met à jour la communauté en lui ajoutant un nouveau nœud. Le nouveau nœud rejoint ou non les communautés existantes selon deux conditions de seuil adaptatives, puis décide si un nouveau lien est capable de former une communauté minimale ou non, et fusionne enfin toutes les communautés très proches les unes des autres (c'est-à-dire qu'elles ont plus qu'un certain ratio de nœuds communs).

3.3 Conclusion

Ce chapitre nous a permis de présenter un aperçu des principales méthodes de détection de communautés dans les réseaux statiques et dynamiques. Dans les réseaux statiques, habituellement, les gens travaillent sur les graphes statiques pour détecter les communautés pour un moment donné. Or, en réalité, les réseaux sont dynamiques dans le cas des réseaux issus du web. Ils évoluent au cours du temps

par l'ajout ou la disparition de liens et de nœuds. La dynamique de ces réseaux et le suivi de leurs structures communautaires est un facteur prépondérant à prendre en considération. La plupart de ces méthodes souffrent particulièrement du problème de limite de résolution dans le sens où les petites communautés peuvent disparaître à cause des regroupements avec d'autres communautés similaires ou bien elles sont absorbées par des communautés relativement beaucoup plus grandes. Cela dit, notre but est donc de trouver une méthode d'identification et de suivi de la structure communautaire dans les réseaux dynamiques, qui soit à même d'atténuer le problème la limite de résolution.

Ch4

Approche pour le suivi des structures communautaires dans les réseaux dynamiques

4.1 Introduction

L'identification et le suivi de la structure communautaire dans les réseaux dynamiques consistent à obtenir des communautés fortement connectées les unes aux autres et faiblement liées au reste du réseau. En d'autres termes, la densité interne des communautés doit être supérieure à leur densité externe. Donc, rechercher la densité interne la plus élevée et la densité externe la plus faible revient à maximiser la somme de leurs différences dans toutes les communautés [19], [34] du réseau.

Le suivi de la structure communautaire nécessite un algorithme rapide et efficace qui utilise la structure communautaire précédente pour détecter la structure communautaire courante. Nous proposons ici une approche pour le suivi de la structure communautaire dans des réseaux évoluant uniquement par l'ajout de nœuds et de leurs liens. Les nœuds et les liens, dans notre cas, sont permanents, et ne peuvent donc être supprimés ultérieurement. Autrement dit, notre approche est une partie d'une approche globale pour des réseaux dynamiques. Notre approche est basée sur une optimisation de la densité à deux niveaux. Cette maximi-

sation permet d'identifier les grandes et les petites communautés puisque la densité concerne les communautés. En d'autres termes, cela permet d'atténuer la limite de résolution dont souffrent la plupart des méthodes de détection de communautés dans les réseaux.

Ce chapitre est consacré à la description de l'approche que nous proposons dans cette thèse. Il est organisé comme suit : Nous commençons par la présentation des points essentiels de l'approche. Ensuite nous introduisons quelques définitions que nous utiliserons tout au long de ce chapitre et nous expliquerons par la suite en détails les deux niveaux d'optimisation de la densité du réseau en s'appuyant sur des exemples illustratifs.

4.2 Présentation de l'approche

Les méthodes existantes de détection de communautés disjointes permettent d'obtenir des communautés dynamiques dans les réseaux dynamiques. Nous proposons une nouvelle approche pour l'identification et le suivi de la structure communautaire dans des réseaux dynamiques évoluant par l'ajout de nœuds et de leurs liens. Notre méthode suggère une optimisation à deux niveaux (cf. sous-section 4.3.3) de la densité du réseau. L'optimisation au premier niveau consiste à intégrer un nœud avec ses liens à la communauté qui maximise la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées. Le deuxième niveau d'optimisation concerne la maximisation de la densité du réseau dans la mesure du possible. Notre approche commence par une structure communautaire initiale et utilise la structure communautaire précédente pour détecter la structure communautaire courante.

4.3 Méthode basée sur la densité avec une double optimisation

Avant de décrire notre méthode, il convient de présenter au préalable certaines notions que nous utiliserons tout au long de cette thèse.

4.3.1 Définitions et préliminaires

Soit $G = (V, E)$ un graphe, non orienté et non pondéré, représentant un réseau social dont V est l'ensemble des nœuds et E l'ensemble des arêtes. L'ensemble des

communautés disjointes du graphe G est défini par $C = \{C_1, C_2, \dots, C_k\}$ où $C_i \in C$, représente une communauté de G.

Ci-après sont définies et décrites certaines notations utilisées dans les formules.

$mI(C_i)$ ……Nombre de liens internes de la communauté i.

$mE(C_i)(C_j)$……Nombre de liens externes de la communauté i vers la communauté j.

$mE(C_i)$ ……Nombre de liens externes de la communauté i vers le graphe G.

nC_i ……Nombre de nœuds dans la communauté i.

$nC_i(x)$……Nombre de voisins du sommet x dans la communauté i.

d_C ……Somme des degrés des nœuds de la communauté c .

nbC……Nombre de communautés dans G.

nbC_{inf}……Nombre de communautés infectées par le changement dans G .

Définition 1 (Communautés infectées)

Soit x un nœud$\in G'$, à intégrer dans G en y ajoutant les liens avec ses voisins potentiels dans G.

On définit ainsi les communautés infectées dans G par les communautés qui contiennent les voisins du nœud x existants dans G. On note ces communautés par l'ensemble :

$C_{inf} = \{C_{inf_1}, C_{inf_2}, \dots, C_{inf_{nbC_{inf}}}\}$ avec $C_{inf_i} \in C$.

Définition 2 (Densité d'un graphe)

La densité δ d'un graphe G est définie par la formule (4.1) suivante :

$$\delta(G) = \frac{M}{\frac{N * (N-1)}{2}} \qquad (4.1)$$

N et M étant respectivement les nombres de nœuds et d'arêtes (liens) de G. $\frac{N*(N-1)}{2}$ est le nombre de liens possibles dans G.

Définition 3 (Densité intra-cluster)

La densité interne de la communauté C_i notée par $\delta_{int}(C_i)$ est définie comme suit par l'équation (4.2) :

$$\delta_{int}(C_i) = \frac{mI(C_i)}{nC_i * \frac{nC_i - 1}{2}} \tag{4.2}$$

Le terme $mI(C_i)$ est le nombre de liens internes de la communauté i. Le terme $nC_i * \frac{nC_i - 1}{2}$ est le nombre de liens internes possibles dans la communauté i.

Définition 4 (Densité inter-cluster)

La densité externe de la communauté C_i notée par $\delta_{ext}(C_i)$ est définie par l'équation (4.3) comme suit :

$$\delta_{ext}(C_i) = \frac{mE(C_i)}{nC_i * (N - nC_i)} \tag{4.3}$$

Où $mE(C_i)$ est le nombre d'arêtes allant des sommets de C_i vers les autres sommets du graphe.

Le terme $nC_i * (N - nC_i)$ est le nombre de liens externes possibles.

Définition 5 (Modularité de Newman)

Fondamentalement, la modularité Q est calculée selon l'équation (4.4) comme suit :

$$Q = \sum_{c \in C} \left[\frac{mI(c)}{M} - \left(\frac{d_c}{2M} \right)^2 \right] \tag{4.4}$$

Avec $\frac{mI(c)}{M}$ qui est le nombre de liens à l'intérieur d'une communauté C. Le terme $\left(\frac{d_c}{2M} \right)^2$ est le nombre de liens attendus à l'intérieur de C si les liens apparaissent aléatoirement dans le graphe tout en respectant la distribution des degrés des nœuds.

Définition 6 (Densité de la modularité)

La densité de la modularité Q_{ds} pour un réseau non dirigé est définie par l'équation (4.5) suivante :

$$Q_{ds} = \sum_{c_i \in C} \left[\frac{mI(c_i)}{M} \delta_{int}(c_i) - \left(\frac{2mI(c_i) + mE(c_i)}{2M} \delta_{int}(c_i) \right)^2 - \sum_{\substack{c_i \in C \\ c_j \neq c_i}} \frac{mE(c_i)(c_j)}{2M} * \frac{mE(c_i)(c_j)}{nC_i * nC_j} \right] \quad (4.5)$$

Le terme $\frac{mE(c_i)(c_j)}{nC_i * nC_j}$ est la densité par paire entre la communauté c_i et c_j.

Corollaire

Pour qu'un ensemble de nœuds soit une communauté C, il faut que $\delta_{int}(C)$ soit plus grande que $\delta(G)$ et que $\delta_{ext}(C)$ soit plus petite que $\delta(G)$[19] [34].

Ainsi, chercher le plus grand $\delta_{int}(C)$ et le petit $\delta_{ext}(C)$, revient à maximiser ω sur tous les clusters de la partition. Ceci est exprimé par l'équation (4.6) comme suit :

$$\omega = \sum_{i=1}^{nbC} [\delta_{int}(C_i) - \delta_{ext}(C_i)] \quad (4.6)$$

4.3.2 Approche pour le suivi de la structure communautaire dans les réseaux sociaux incrémentiels[8]

Nous proposons une nouvelle approche pour le suivi de la structure communautaire dans les réseaux sociaux [7]. Le changement du réseau est dû à l'ajout d'un nœud x avec ses liens sachant que ces entités sont ajoutées simultanément. Notre approche suit l'évolution du réseau uniquement sur les communautés concernées par le changement. Dans cette optique, la méthode proposée consiste à démarrer par une structure communautaire initiale obtenue par l'application de l'algorithme de Louvain [6]. Elle est incrémentale puisqu'elle utilise la structure communautaire précédente pour détecter la structure communautaire courante.

L'apparition des événements à chaque instant transforme le réseau G en un réseau dynamique. L'ensemble des événements qui peuvent atteindre le réseau G sont :

- L'ajout d'un nœud isolé x.

[8] *Réseau Incrémentiel ou incrémental évoluant (progressant) uniquement par ajout de nœuds et de leurs liens.*

- L'ajout d'un nœud x dont les liens se retrouvent avec des nœuds qui sont dans une même communauté.
- L'ajout d'un nœud x dont les liens se retrouvent avec des nœuds qui sont dans plusieurs communautés.

L'événement le plus complexe est le cas de l'ajout d'un nœud x ayant des liens avec des nœuds appartenant à plusieurs communautés. Dans un tel cas, notre méthode suggère une optimisation à deux niveaux. Cette optimisation consiste en la maximisation de ω sur les communautés infectées. On définit ci-après l'optimisation à deux niveaux.

4.3.3 L'optimisation à deux niveaux

4.3.3.1 Premier niveau d'optimisation

Pour déterminer la meilleure communauté k à laquelle il faut ajouter le nœud x, on maximise ω uniquement sur l'ensemble des communautés infectées C_{inf} d'où l'équation (4.7) :

$$\omega_{Cinf} = Argmax_{k=1,\dots,nbC_{inf}} \left(\frac{\alpha + \beta}{nbC_{inf}} \right) \tag{4.7}$$

Tel que :

- $\alpha = \delta_{int}(C_{inf_k}) - \delta_{ext}(C_{inf_k})$ désigne la différence entre la densité interne de la communauté k et la densité externe vers les autres communautés infectées.
- $\beta = \sum_{j=1,j\neq k}^{nbC_{inf}} [\delta_{int}\left(C_{inf_j}\right) - \delta_{ext}\left(C_{inf_j}\right)]$ est la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées.

Le nœud x peut intégrer une communauté C_{inf_k} dont la densité interne est donnée par : $\delta_{int}(C_{inf_k}) = \frac{mI\left(C_{inf_k}\right) + nC_{inf_k}(x)}{(nC_{inf_k}+1) * (\frac{nC_{inf_k}}{2})}$

et sa densité externe vers les autres communautés infectées est :

$$\delta_{ext}(C_{inf_k}) = \frac{\sum_{p=1,p\neq k}^{nbC_{inf}} \left(mE\left(C_{inf_k}\right)\left(C_{inf_p}\right) + nC_{inf_p}(x) \right)}{\left(nC_{inf_k}+1\right) * \left(\left(\sum_{q=1}^{nbC_{inf}} \left(nC_{inf_q}+1 \right) \right) - \left(nC_{inf_k}+1 \right) \right)}$$

D'où $\alpha = \frac{mI\left(C_{inf_k}\right)+nC_{inf_k}(x)}{\left(nC_{inf_k}+1\right)*\left(\frac{nC_{inf_k}}{2}\right)} - \frac{\sum_{p=1,p\neq k}^{nbC_{inf}}\left(mE\left(C_{inf_k}\right)\left(C_{inf_p}\right)+nC_{inf_p}(x)\right)}{\left(nC_{inf_k}+1\right)*\left(\left(\sum_{q=1}^{nbC_{inf}}\left(nC_{inf_q}+1\right)\right)-\left(nC_{inf_k}+1\right)\right)}$

En ce qui concerne les calculs de β on a :

$$\delta_{int}\left(C_{inf_j}\right) = \frac{mI\left(C_{inf_j}\right)}{(nC_{inf_j}) * \frac{nC_{inf_j}-1}{2}}$$

et $$\delta_{ext}\left(C_{inf_j}\right) = \frac{\sum_{p=1,p\neq j}^{nbC_{inf}}\left(mE\left(C_{inf_j}\right)\left(C_{inf_p}\right)\right)+nC_{inf_j}(x)}{nC_{inf_j}*\left(\left(\sum_{q=1}^{nbC_{inf}}\left(nC_{inf_q}\right)+1\right)-nC_{inf_j}\right)}$$

On a donc la formule de β sur l'ensemble des communautés infectées qui s'exprime comme suit:

$$\beta = \sum_{j=1,j\neq k}^{nbC_{inf}}\left(\left(\frac{mI\left(C_{inf_j}\right)}{(nC_{inf_j}) * (nC_{inf_j}-1)/2}\right) - \left(\frac{\sum_{p=1,p\neq j}^{nbC_{inf}}\left(mE\left(C_{inf_j}\right)\left(C_{inf_p}\right)\right)+nC_{inf_j}(x)}{nC_{inf_j} * \left(\left(\sum_{q=1}^{nbC_{inf}}\left(nC_{inf_q}\right)+1\right)-nC_{inf_j}\right)}\right)\right)$$

Ainsi, le nœud x intègre la communauté qui vérifie l'équation (4.7).

Le processus d'optimisation de premier niveau et d'ajout du nœud x est exprimé par l'algorithme 4.1.

Algorithme 4. 1 Premier niveau d'optimisation

Entrée : nœud x et ses liens $\{(x,v),(x,w),\dots,(x,z)\}$, structure communautaire courante C_t , les communautés infectées C_{inf} .

Sortie : la structure communautaire à C_{t+1}.

$som\_den \leftarrow 0$; //*La somme des différences entre la densité interne et la densité externe de toutes les* C_{inf} .

Pour $C_{inf_k} \in C_{inf}$ **faire**

$som\_den \leftarrow som\_den + \left(\delta_{int}\left(C_{inf_k}\right) - \delta_{ext}\left(C_{inf_k}\right)\right)$;

$C_{inf'} \leftarrow C_{inf} \setminus C_{inf_k}$; //*L'ensemble des* C_{inf} *moins la communauté* C_{inf_k} .

Pour $C_{inf_j} \in C_{inf'}$ **faire**

$som\_den \leftarrow som\_den + \left(\delta_{int}\left(C_{inf_j}\right) - \delta_{ext}\left(C_{inf_j}\right)\right)$;

Fin pour ;

Calculer et sauvegarder som_den/nbC_{inf} ;

Fin pour ;

Intégrer le nœud x à la communauté qui vérifie :

$\omega_{Cinf} = Argmax_{k=1,\dots,nb\,C_{inf}}\left(som\_den/nbC_{inf}\right)$;

Retourner $C_{t+1} \leftarrow C_t \cup x \cup \{(x,v),(x,w),\dots,(x,z)\}$;

Exemple illustratif

On applique l'algorithme 4.1 à un réseau de 11 nœuds et 12 liens représenté dans la figure 4.1.

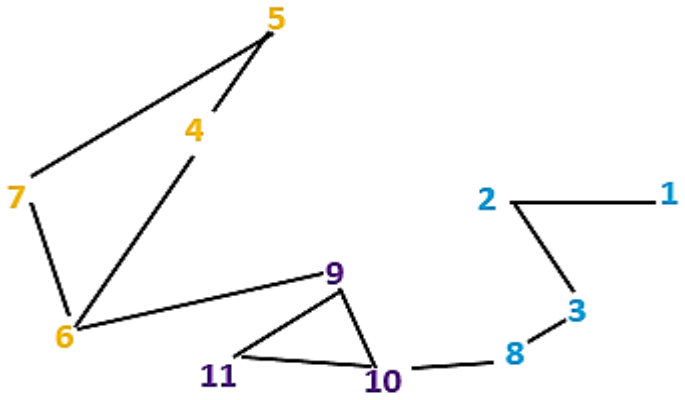

Figure 4. 1 Exemple de réseau

On voudrait y intégrer le nœud 12 avec les liens vers les nœuds 1, 2, 4, 5. Dans ce cas, on voit que les communautés infectées dans le réseau sont : la communauté

bleue (com_bleue) et la communauté jaune (com_jaune). Le nœud 12 peut intégrer soit la communauté bleue ou bien la communauté jaune :

Si on ajoute le nœud 12 à la communauté jaune alors :

- $\delta_{int}(com\_jaune) = \frac{6}{5*(4/2)} = 0{,}6$ et $\delta_{ext}(com\_jaune) = \frac{2}{5*(9-5)} = 0{,}1$.

La différence des densités est :

$$\delta_{int}(com\_jaune) - \delta_{ext}(com\_jaune) = 0{,}6 - 0{,}1 = 0{,}5 \qquad (A)$$

- $\delta_{int}(com\_bleue) = \frac{3}{4*(3/2)} = 0{,}5$ et $\delta_{ext}(com\_bleue) = \frac{2}{4*(9-4)} = 0{,}1$.

La différence des densités est :

$$\delta_{int}(com\_bleue) - \delta_{ext}(com\_bleue) = 0{,}5 - 0{,}1 = 0{,}4 \qquad (B)$$

D'où : $\omega_{Cinf} = \frac{A+B}{2} = \frac{0{,}5+0{,}4}{2} = 0{,}45$.

Si on intègre le nœud 12 dans la communauté bleue alors :

- $\delta_{int}(com\_bleue) = \frac{5}{5*(4/2)} = 0{,}5$ et $\delta_{ext}(com\_bleue) = \frac{2}{5*(9-5)} = 0{,}1$.

La différence des densités est :

$$\delta_{int}(com\_bleue) - \delta_{ext}(com\_bleue) = 0{,}5 - 0{,}1 = 0{,}4 \qquad (C)$$

- $\delta_{int}(com\_jaune) = \frac{4}{4*(3/2)} = \frac{2}{3}$ et $\delta_{ext}(com\_jaune) = \frac{2}{4*(9-4)} = 0{,}1$.

La différence des densités est :

$$\delta_{int}(com\_jaune) - \delta_{ext}(com\_jaune) = \frac{2}{3} - 0{,}1 \cong 0{,}56 \qquad (D)$$

D'où: $\omega_{Cinf} = \frac{C+D}{2} = \frac{0{,}4+0{,}56}{2} = 0{,}48$.

D'après les résultats du calcul, nous avons 0,48 > 0,45, donc le nœud 12 s'ajoute à la communauté bleue (voir la figure 4.2) qui a donné un ω_{Cinf} élevé.

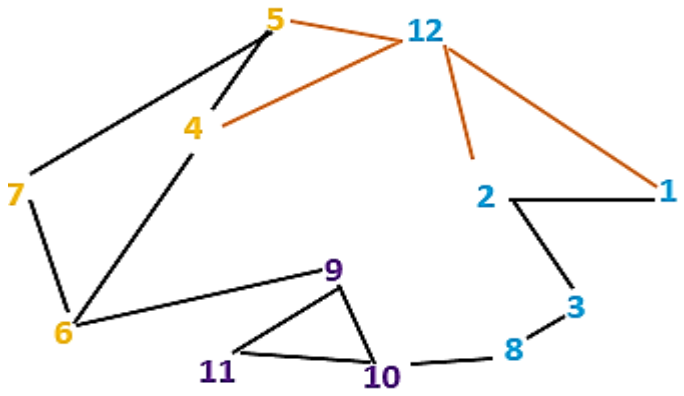

Figure 4. 2 Le réseau de la figure 4.1 après intégration du nœud 12 avec ses liens en rouge (premier niveau d'optimisation).

4.3.3.2 Deuxième niveau d'optimisation

Après l'intégration du nœud x avec ses liens à la communauté ayant donné un bon score de ω_{Cinf}, on passe à la deuxième étape, à savoir l'optimisation de deuxième niveau. Cette dernière consiste à vérifier si l'une des opérations suivantes : *fusion* ou *éclatement (naissance)* de communautés maximise mieux ω_{Cinf} .

A) Principe du test de fusion

La communauté C_{inf_k} contenant le nœud intégré x peut-être fusionnée avec C_{inf_j} , si le nombre de ses liens internes est inférieur ou égal au nombre de ses liens externes vers C_{inf_j} . Si c'est le cas, on calcule $\omega_{Cinf_{fusion}}$ qui est donné par la formule (4.8), sinon $\omega_{Cinf_{fusion}} = 0$, ce qui indique l'absence de fusion.

$$\omega_{Cinf_{fusion}} = Argmax_{j=1,\dots,nbC_{inf},j\neq k}\left((\alpha+\beta)/(nbC_{inf}-1)\right) \quad (4.8)$$

Tel que :

$\alpha = \delta_{int}(C_{inf_k}) - \delta_{ext}(C_{inf_k})$ avec $\delta_{int}(C_{inf_k})$ qui est la densité interne de C_{inf_k} si elle sera fusionnée avec C_{inf_j} et $\delta_{ext}(C_{inf_k})$ est la densité externe de C_{inf_k} vers C_{inf_p} avec $p \neq j \; et \; p \neq k$.

$\beta = \sum_{p=1,p\neq j,p\neq k}^{nbC_{inf}} \left(\delta_{int}\left(C_{inf_p}\right) - \delta_{ext}\left(C_{inf_p}\right)\right)$ est la différence entre la densité interne et la densité externe des autres C_{inf_p}.

$nbC_{inf} - 1$ est le nombre de communautés infectées décrémenté de 1 lorsqu'on fusionne deux communautés.

Si C_{inf_k} se fusionne avec C_{inf_j}, la densité interne de la communauté est exprimée par : $\delta_{int}(C_{inf_k}) = \dfrac{mI(C_{inf_k})+mI(C_{inf_j})+mE(C_{inf_k})(C_{inf_j})}{(nC_{inf_k}+nC_{inf_j})^{*\frac{\left((nC_{inf_k}+nC_{inf_j})-1\right)}{2}}}$

et sa densité externe est donnée par :

$$\delta_{ext}(C_{inf_k}) = \frac{\sum_{p=1,p\neq k,p\neq j}^{nbC_{inf}} \left(mE(C_{inf_k})\left(C_{inf_p}\right) + mE\left(C_{inf_j}\right)\left(C_{inf_p}\right)\right)}{\left(nC_{inf_k}+nC_{inf_j}\right) * \left(\left(\sum_{q=1}^{nbC_{inf}} nC_{inf_q}\right) - \left(nC_{inf_k}+nC_{inf_j}\right)\right)}$$

D'où $\alpha = \left(\frac{mI\left(C_{inf_k}\right)+mI\left(C_{inf_j}\right)+mE\left(C_{inf_k}\right)\left(C_{inf_j}\right)}{\left(nC_{inf_k}+nC_{inf_j}\right)*\frac{\left(\left(nC_{inf_k}+C_{inf_j}\right)-1\right)}{2}} \right) -$

$$\left(\frac{\sum_{p=1,p\neq k,p\neq j}^{nbC_{inf}}\left(mE\left(C_{inf_k}\right)\left(C_{inf_p}\right)+mE\left(C_{inf_j}\right)\left(C_{inf_p}\right)\right)}{\left(nC_{inf_k}+nC_{inf_j}\right)*\left(\left(\sum_{q=1}^{nbC_{inf}} nC_{inf_q}\right)-\left(nC_{inf_k}+nC_{inf_j}\right)\right)} \right)$$

De même pour β nous avons :

$$\delta_{int}\left(C_{inf_p}\right) = \frac{mI\left(C_{inf_p}\right)}{nC_{inf_p}*\frac{\left(nC_{inf_p}-1\right)}{2}} \quad \text{et} \quad \delta_{ext}\left(C_{inf_p}\right) = \frac{\sum_{q=1,q\neq p}^{nbC_{inf}} mE\left(C_{inf_p}\right)\left(C_{inf_q}\right)}{\left(nC_{inf_p}\right)*\left(\left(\sum_{q=1}^{nbC_{inf}} nC_{inf_q}\right)-nC_{inf_p}\right)}$$

D'où $\beta = \sum_{p=1,p\neq k,p\neq j}^{nbC_{inf}} \left(\left(\frac{mI\left(C_{inf_p}\right)}{nC_{inf_p}*\frac{\left(nC_{inf_p}-1\right)}{2}} \right) - \left(\frac{\sum_{q=1,q\neq p}^{nbC_{inf}} mE\left(C_{inf_p}\right)\left(C_{inf_q}\right)}{\left(nC_{inf_p}\right)*\left(\left(\sum_{q=1}^{nbC_{inf}} nC_{inf_q}\right)-nC_{inf_p}\right)} \right) \right)$

B) Principe du test d'éclatement (naissance)

En ce qui concerne le test d'éclatement (naissance) de communauté, on passe d'abord par une condition d'existence ou non des cliques >= 3 dans la communauté contenant le nouveau nœud (test éclatement) ou entre la communauté contenant le nouveau nœud et les autres communautés infectées (test naissance). Ces cliques si elles existent, peuvent former de nouvelles communautés C_{new}.

Si la condition est vérifiée, on retourne la valeur de $\omega_{Cinf_{nais-ecla}}$, sinon $\omega_{Cinf_{nais-ecla}} = 0$, ce qui indique l'absence d'éclatement ou de naissance d'une communauté. Le calcul de $\omega_{Cinf_{nais-ecla}}$ est donné par la formule (4.9).

$$\omega_{Cinf_{\text{nais}-ecla}} = Argmax_{k=1\ldots nbC_{inf}}\left(\frac{(\alpha+\beta)}{\left(nbC_{inf}+1\right)}\right) \tag{4.9}$$

Tel que

$\alpha = \delta_{int}(C_{new}) - \delta_{ext}(C_{new})$ avec $\delta_{int}(C_{new})$ est la densité interne de la clique qui peut former C_{new} et $\delta_{ext}(C_{new})$ est la densité externe de C_{new} vers les autres C_{inf_j} avec $j \neq C_{new}$.

$\beta = \delta_{int}\left(C_{inf_k}\right) - \delta_{ext}\left(C_{inf_k}\right) + \sum_{j=1,j\neq C_{new},j\neq k}^{nbC_{inf}}\left(\delta_{int}\left(C_{inf_j}\right) - \delta_{ext}\left(C_{inf_j}\right)\right)$ est la différence entre la densité interne et la densité externe des autres communautés infectées.

$nbCinf + 1$ est le nombre de communautés infectées incrémenté de 1, lorsqu'une communauté se forme.

Le processus du deuxième niveau d'optimisation est décrit par l'algorithme 4.2.

Algorithme 4. 2 Deuxième niveau d'optimisation

Entrée : $\omega_{Cinf_{nais-ecla}}$, $\omega_{Cinf_{fusion}}$, ω_{Cinf} , C_{t+1}: La structure communautaire à t+1.

Sortie : C_{t+1}' : La structure communautaire mise à jour à t+1.

$\omega_{Cinf_{fusion}} \leftarrow 0; \omega_{Cinf_{nais-ecla}} \leftarrow 0;$

// Test de fusion de communautés

Calculer $\omega_{Cinf_{fusion}}$;

// Test de naissance-éclatement de communautés

Calculer $\omega_{Cinf_{nais-ecla}}$;

$\omega_{Cinf\_new} \leftarrow max\left(\omega_{Cinf_{fusion}}, \omega_{Cinf_{nais-ecla}}\right);$

Si $\omega_{Cinf\_new} > \omega_{Cinf}$ **alors**

Mettre à jour la structure communautaire C_{t+1}' selon ω_{Cinf_new} en utilisant C_{t+1} ;

Fin Si;

Retourner C_{t+1}';

Exemple illustratif

Après l'intégration du nœud 12 à la communauté satisfaisante, nous passons au deuxième niveau d'optimisation. Nous testons si l'une des opérations fusion ou éclatement-naissance maximise plus ω_{Cinf}. Le $\omega_{Cinf_{fusion}} = 0$ puisque la condition de fusion n'est pas vérifiée sur les communautés infectées (le nombre de liens internes de la communauté bleue est supérieur au nombre de ses liens externes).

En ce qui concerne la condition de l'éclatement, elle est vérifiée. Dans la figure 4.2, il existe une clique {12, 1, 2} dans la communauté bleue qui peut former une communauté C_{new}.

Quant au test de naissance, il existe aussi une clique {12, 5, 4} entre les nœuds de la communauté bleue et les nœuds de la communauté jaune qui peut former une nouvelle communauté.

D'après la figure 4.2, il existe deux cliques {12, 1, 2}, {12, 5, 4} qui peuvent former une nouvelle communauté C_{new}. Le choix de la meilleure clique revient au choix du

maximum des calculs de somme des différences entre la densité interne et la densité externe sur toutes les communautés infectées.

Si C_{new}= {12, 1, 2} alors :

- $\delta_{int}(C_{new}) = \frac{3}{3*(2/2)} = 1$ et $\delta_{ext}(C_{new}) = \frac{3}{3*(9-3)} \cong 0{,}16$.

- $\delta_{int}(C_{new}) - \delta_{ext}(C_{new}) = 1 - 0{,}16 = 0{,}84$.

Les calculs de la communauté bleue représentant les nœuds (8, 3) sont donnés comme suit :

- $\delta_{int}(com\_bleue) = \frac{1}{2*(1/2)} = 1$ et $\delta_{ext}(com\_bleue) = \frac{1}{2*(9-2)} \cong 0{,}07$.

- $\delta_{int}(com\_bleue) - \delta_{ext}(com\_bleue) = 1 - 0{,}07 = 0{,}93$.

Concernant les calculs de la communauté jaune de la figure 4.2, ils sont réalisés comme suit :

- $\delta_{int}(com\_jaune) = \frac{4}{4*(3/2)} \cong 0{,}66$ et $\delta_{ext}(com\_jaune) = \frac{2}{4*(9-4)} = 0{,}1$.

- $\delta_{int}(com\_jaune) - \delta_{ext}(com\_jaune) = 0{,}66 - 0{,}1 = 0{,}56$.

D'où $\omega_{Cinf_{nais-ecla}} = \frac{0{,}84+0{,}93+0{,}56}{3} \cong 0{,}77$.

Si C_{new} = {12, 5, 4} alors :

- $\delta_{int}(C_{new}) = \frac{3}{3*(2/2)} = 1$ et $\delta_{ext}(C_{new}) = \frac{4}{3*(9-3)} \cong 0{,}22$.

- $\delta_{int}(C_{new}) - \delta_{ext}(C_{new}) = 1 - 0{,}22 = 0{,}78$.

Les calculs de la communauté bleue représentant les nœuds (1, 2, 3, 8) sont donnés comme suit :

- $\delta_{int}(com\_bleue) = \frac{3}{4*(3/2)} = 0.5$ et $\delta_{ext}(com\_bleue) = \frac{2}{4*(9-4)} = 0{,}1$.

- $\delta_{int}(com\_bleue) - \delta_{ext}(com\_bleue) = 0.5 - 0{,}1 = 0{,}4$.

Concernant les calculs de la communauté jaune (7, 6) de la figure 4.2, ils sont réalisés comme suit :

- $\delta_{int}(com\_jaune) = \frac{1}{2*(1/2)} = 1$ et $\delta_{ext}(com\_jaune) = \frac{2}{2*(9-2)} \cong 0{,}14$.

- $\delta_{int}(com\_jaune) - \delta_{ext}(com\_jaune) = 1 - 0{,}14 = 0{,}86$.

D'où $\omega_{Cinf_{nais-ecla}} = \frac{0{,}78+0{,}4+0{,}86}{3} = 0.68$.

Après avoir calculé $\omega_{Cinf_{nais-ecla}}$, nous remarquons que $\omega_{Cinf_{fusion}} < \omega_{Cinf}$, mais $\omega_{Cinf_{nais-ecla}} = (0{,}77 \text{ ou } 0{,}68) > \omega_{Cinf} = 0{,}483$, donc ω_{Cinf} sur les communautés infectées est maximisé plus dans le cas d'éclatement de la communauté bleue

ou de naissance d'une communauté entre les nœuds de la communauté bleue et jaune. Puisque nous avons retourné deux valeurs de $\omega_{Cinf_{nais-ecla}}$, une pour l'éclatement et l'autre pour la naissance de communauté, nous choisissons la valeur maximale qui est égale à 0 ,77 = $\omega_{Cinf_{new}}$. Nous mettons à jour la structure communautaire du réseau de la figure 4.2 avec l'éclatement de la communauté bleue. Le réseau de la figure 4.2 après optimisation au deuxième niveau est illustré sur la figure 4.3.

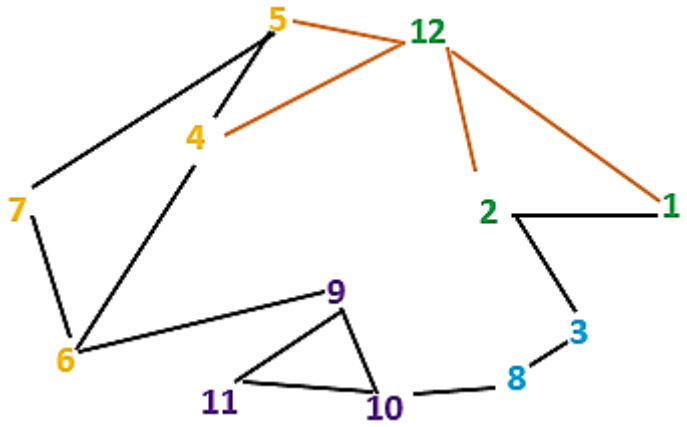

Figure 4. 3 Le réseau de la figure 4.2 après l'éclatement de la communauté bleue (deuxième niveau d'optimisation).

D'après l'exemple, nous remarquons l'existence de plusieurs alternatives dans la prise en compte des communautés lors de l'optimisation du deuxième niveau. Dans un tel cas, nous prenons l'alternative qui donne une meilleure densité de communauté et en cas d'égalité nous choisissons la communauté qui a la meilleure densité interne. De ce fait notre algorithme est bien déterministe[9].

Le processus général d'ajout d'un nœud avec ses liens simultanément est décrit dans l'algorithme 4.3.

[9] Un algorithme est dit déterministe si sa multiplicité d'exécution sur le même réseau produit la même structure communautaire finale

Algorithme 4. 3 Le processus général d'ajout d'un nœud x et ses liens

Si le nœud x est isolé dans G **alors** créer une nouvelle communauté pour x
  Sinon Si les voisins du nœud x se retrouvent dans la même communauté **alors**
    Ajouter le nœud x et ses liens directement dans cette communauté ;
    Sinon Si les voisins du x se retrouvent dans plusieurs communautés **alors**
      //Optimisation de premier niveau.
      Rechercher la meilleure communauté pour le nœud x ;
      Ajouter x et ses liens dans cette communauté;
      //Optimisation de deuxième niveau.
      Tester si la fusion ou l'éclatement (naissance) des communautés infectées maximise mieux ω ;
    Fin si ;
  Fin si ;
Fin si ;

Rappelons que notre algorithme est incrémental et sa complexité algorithmique est comme suit :

-Dans le premier niveau d'optimisation, elle est on $0(nbC_{inf})$ puisqu'on touche uniquement les communautés infectées par le changement.

-Dans le deuxième niveau d'optimisation, elle est en $0(1)$.

La complexité globale de l'algorithme est on $0(nbC_{inf})$ ce qui rend la méthode rapide.

4.4 Conclusion

Dans ce chapitre, nous avons présenté une approche basée sur la densité pour le suivi de la structure communautaire dans des réseaux dynamiques qui évoluent seulement avec l'ajout de nœuds et de leurs liens. La méthode maximise la densité de réseau à deux niveaux. Le premier niveau d'optimisation consiste en l'intégration du nœud dans la communauté maximisant la somme des différences entre la densité interne et la densité externe de toutes les communautés affectées par le changement (communautés infectées). Le deuxième niveau d'optimisation vise à améliorer davantage le score de la densité du réseau par le test des opérations sur

les communautés. Nous avons également utilisé la structure précédente pour identifier la structure communautaire courante. Dans le prochain chapitre, nous allons nous employer à valider notre approche en l'appliquant à des réseaux du monde réel et à évaluer les résultats retournés.

Ch5

Expérimentation et tests de validation

5.1 Introduction

Dans ce chapitre, nous présentons les résultats obtenus par la mise en œuvre et l'application de notre méthode. Cette dernière est l'ensemble des algorithmes proposés et décrits dans le chapitre 4. Nous avons, à cet effet, effectué des tests pour évaluer l'efficacité de notre approche. Les tests ont été menés sur trois fronts :

(1) Montrer la capacité de notre méthode à détecter efficacement les communautés en testant la modularité de Newman [38] et la densité de la modularité [60] et à évaluer sa performance en fonction de son temps de traitement ainsi que le nombre de communautés identifiées.

Dans ces premiers tests, nous avons appliqué notre algorithme sur une collection de données dont la structure communautaire est inconnue. La dynamique du réseau est due à l'importation d'un nœud et de ses liens dans deux réseaux émetteurs (cf. section 5.2 A) et son intégration à un réseau récepteur (cf. section 5.2 B).

(2) Montrer la qualité, la stabilité et la validité de la structure communautaire obtenue.

Dans les seconds tests, nous avons utilisé des données du monde réel dont la structure communautaire est connue. Pour constituer un réseau dynamique, nous avons

choisi un pourcentage de nœuds du réseau du monde réel aléatoirement pour former un réseau initial. Au réseau initial, nous avons ajouté le pourcentage des nœuds restants avec leurs liens afin de produire le réseau du monde réel.

(3) Montrer l'efficacité de l'algorithme d'atténuer le problème de limite de résolution.

Enfin, le dernier test concerne un jeu de données d'eprint arXiv dont la dynamique de réseau est prise dans KDD Cup 2003 (Knowledge Discover and Data mining)[10] [63].

Notre méthode (algorithmes) a été implémentée sous l'environnement Jetbrains Pycharm Community Edition[11] 2017 2. 3 avec le langage python, sur un Compaq avec Intel Celeron CPU 2 GHZ et 2 GO de RAM.

5.2 Tests d'évaluation des métriques

Cette section est divisée en deux tests d'évaluation. Le premier concerne l'évaluation des valeurs de modularité, de densité de la modularité, de temps d'exécution et de nombre de communautés identifiées. Le second test repose sur l'évaluation de la densité de la modularité des réseaux avec des structures communautaires initiales différentes. Celles-ci ont été réalisées sur deux classes de réseaux réels, à savoir les réseaux émetteurs et les réseaux récepteurs. Les réseaux sont choisis de KONECT[12], (la collection du réseau de Koblenz).

A- Les réseaux émetteurs

Les réseaux émetteurs sont les réseaux dans lesquels on choisit aléatoirement des individus à intégrer aux réseaux récepteurs. Les réseaux émetteurs utilisés durant cette expérimentation sont le réseau « Douban » et le réseau « arXiv-astro-ph ». Le nombre d'individus sélectionnés dans ces réseaux est égal à 10.

-Le réseau Douban est un réseau social de Douban qui correspond au site chinois de recommandation en ligne.

[10] *La compétition de la KDD Cup de 2003 portait sur l'exploitation de réseaux complexes ; il était basé sur un ensemble de données de l'e-print arXiv (arXiv.org), ainsi qu'un ensemble de tâches conçu pour capturer certains des défis inhérents à l'analyse de grands réseaux sociaux*

[11] https://www.jetbrains.com/pycharm/download/other.html

[12] *http://konect.uni-koblenz.de*

-Le réseau arXiv-astro-ph est un réseau de collaboration entre les auteurs d'articles scientifiques de la section Astrophysiques (Astro-ph) d'arXiv.

B- Les réseaux récepteurs

Les réseaux récepteurs sont les réseaux sur lesquels on opère l'ajout des individus avec les liens qu'ils possèdent dans les réseaux émetteurs s'ils existent dans le réseau récepteur. Les réseaux récepteurs considérés sont :

Le réseau « Jazz musicians »,

Le réseau « Hamsterster friendships » et

Le réseau « Facebook(NIPS) ».

-Le réseau Jazz musicians est un réseau social collecté en 2003, il représente les collaborations entre les musiciens de jazz. Il contient 198 nœuds avec 2 742 liens. Un lien indique que deux musiciens ont joué ensemble dans un groupe.

-Le réseau Hamsterster friendships est un réseau social qui représente 12 534 relations d'amitiés entre 1 858 utilisateurs du site web hamsterster.com.

-Le réseau Facebook(NIPS) est un réseau social non orienté qui représente 2 981 relations d'amitiés entre 2 888 utilisateurs de Facebook.

La structure communautaire initiale des réseaux récepteurs est obtenue par l'algorithme de Louvain [6]. L'ajout des 10 nœuds avec leurs connections se fait en 10 instants.

Rappelons que notre méthode est incrémentale, autrement dit, la structure communautaire courante est déduite de la structure communautaire précédente.

5.2.1 Evaluation de la modularité, la densité de la modularité, le temps d'exécution et le nombre de communautés

Pour mettre en évidence la fiabilité et la performance de notre méthode, nous avons réalisé plusieurs tests afin de comparer notre algorithme à celui de l'approche de Louvain décrit dans [6]. Ainsi, nous avons exécuté les deux algorithmes sur chacun des réseaux récepteurs et nous avons retourné les meilleurs résultats concernant la modularité, la densité de la modularité, le temps d'exécution et le nombre de communautés détectées.

-Au sujet du réseau *Jazz musicians*, après l'ajout des 10 nœuds et de leurs liens, le réseau obtenu sera composé de 208 nœuds avec 3 063 liens. La figure 5.1(a) montre que notre algorithme retourne des valeurs de modularité identiques ou un peu plus élevées avec un nombre de communautés élevé (Figure 5.1(b)). Concernant le temps d'exécution pour chaque instantané du réseau, notre algorithme atteint un maximum qui est égal à 6 s à t=3 (Figure 5.1(c)), tandis que Louvain consomme un temps égal à 4 fois le nôtre. Notre algorithme réduit donc le temps d'exécution car il ne considère que les communautés touchées par le changement, contrairement à Louvain qui fonctionne sur tout le réseau. Les valeurs de densité de la modularité (Figure 5.1(d)) dans les premiers instantanés sont plus élevées que Louvain, inférieures à t = 4 parce que notre algorithme produit plus de communautés (sept) que Louvain (quatre).

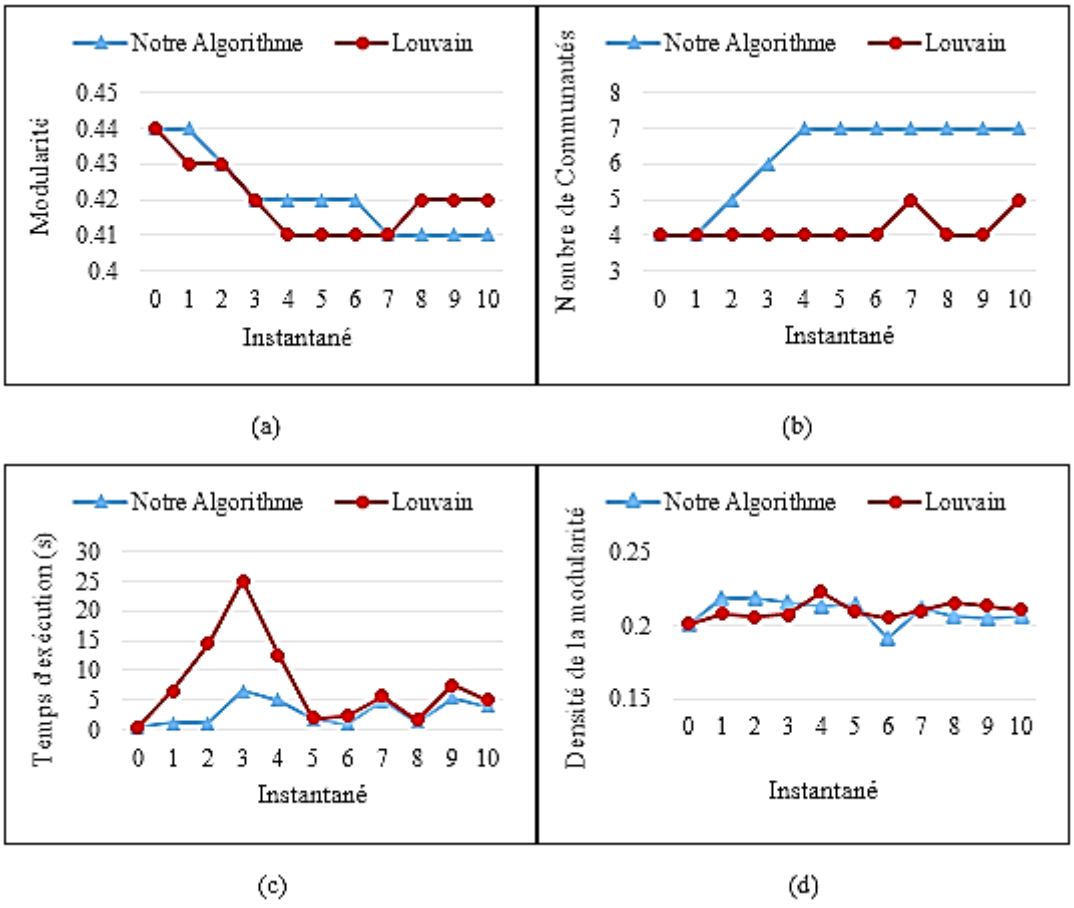

Figure 5. 1 Les résultats obtenus concernant la modularité, le nombre de communautés, le temps d'exécution et la densité de la modularité sur chaque instantané du réseau *Jazz musicians*.

-En ce qui concerne le réseau *Hamsterster friendships*, notre algorithme renvoie des valeurs de modularité supérieures à celles de Louvain sur trois instantanés et

relativement proches pour les autres instantanés restants (Figure 5.2(a)). Après l'ajout des 10 nœuds et de leurs liens, le réseau sera composé de 1 868 nœuds et 13 243 liens.

La Figure 5.2(b) montre que notre algorithme a la capacité de détecter et de découvrir plus de communautés que la méthode statique. La différence du nombre de communautés entre les deux algorithmes est expliquée par la combinaison des petites communautés par la méthode statique de Louvain afin de maximiser la modularité du réseau, contrairement à notre algorithme qui lui prend en considération le problème de la limite de résolution. Enfin, le temps nécessaire pour mettre à jour la structure communautaire du réseau par notre algorithme prend au maximum 77s (Figure 5.2 (c)) ; tandis que la méthode de Louvain a besoin d'un temps 6 fois plus élevé. Enfin, la figure 5.2(d) montre que notre densité de la modularité est plus élevée que Louvain sur tous les instantanés.

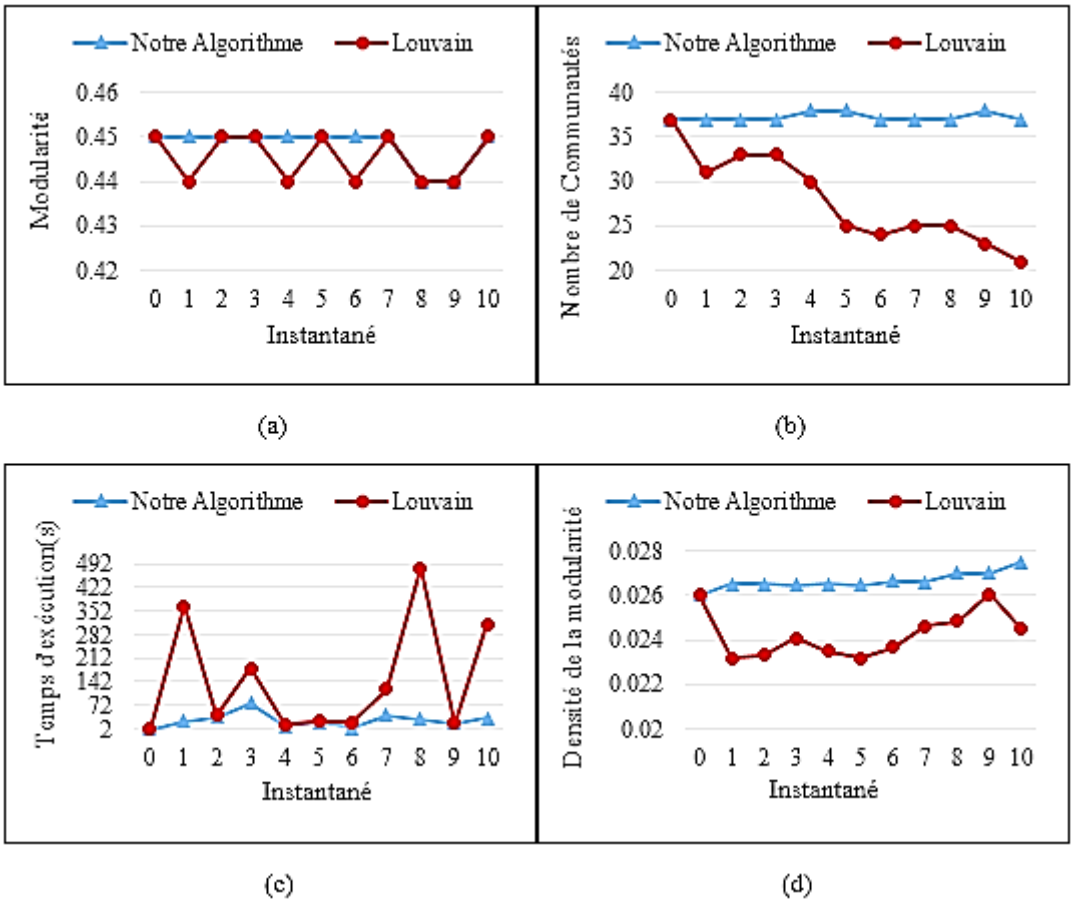

Figure 5. 2 Les résultats obtenus concernant la modularité, le nombre de communautés, le temps d'exécution et la densité de la modularité sur chaque instantané du réseau *Hamsterster friendships*.

-Pour le réseau *Facebook*, après avoir ajouté 10 nœuds avec leurs liens, il devient un réseau de 2 898 nœuds et 4 945 liens. La figure 5.3(a) montre que notre algorithme retourne une modularité égale à celle de Louvain presque sur tous les instantanés. Quant aux nombre de communautés détectées par notre algorithme (Figure 5.3(b)), il est plus grand que celui de la méthode statique qui souffre du problème de la limite de résolution de la modularité [20]. Le temps d'exécution sur chaque instantané du réseau des deux algorithmes est illustré dans la figure 5.3(c). Notre algorithme prend au moins 1s et au plus 73s, tandis que l'algorithme de Louvain nécessite au moins 4s et au plus 400s. Notre densité de la modularité surpasse celle de Louvain à tous les instantanés (Figure 5.3(d)).

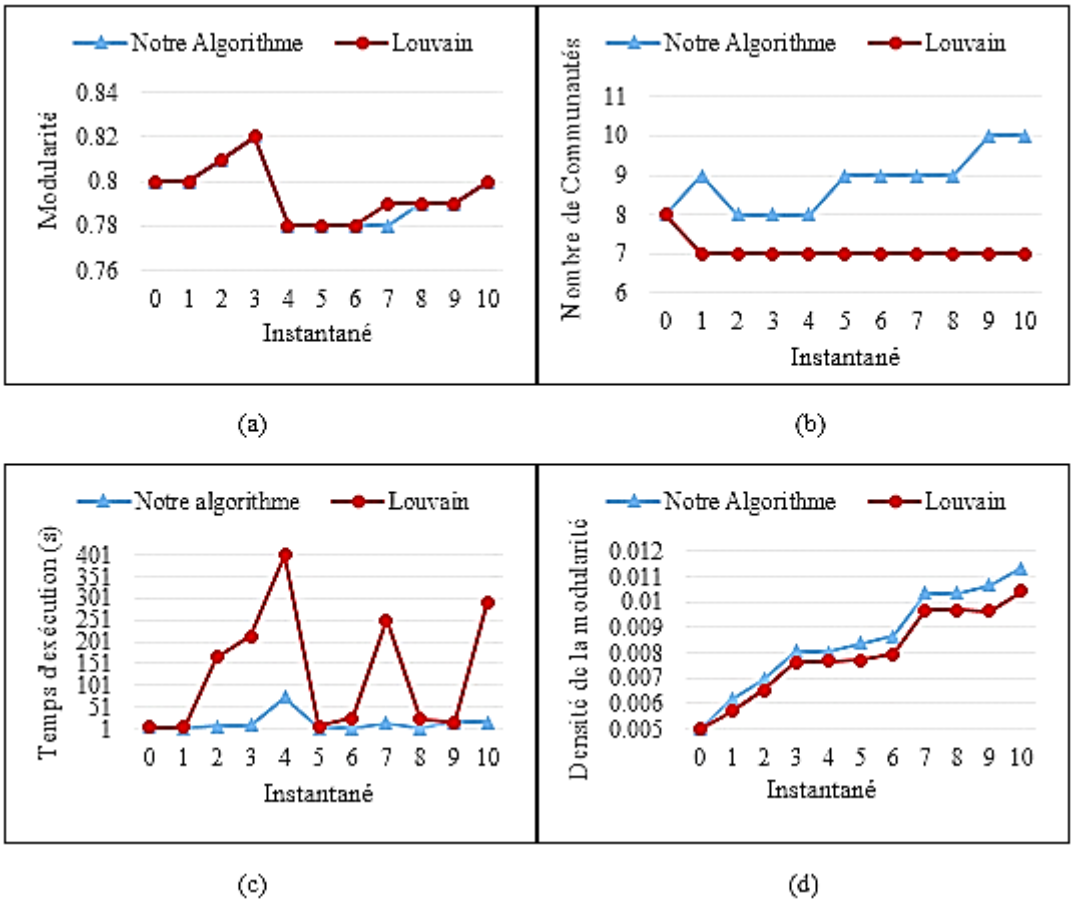

Figure 5. 3 Les résultats obtenus concernant la modularité, le nombre de communautés, le temps d'exécution et la densité de la modularité sur chaque instantané du réseau *Facebook*.

5.2.2 Evaluation de la densité de la modularité en fonction de la structure communautaire initiale

Pour prouver que l'exécution de notre algorithme avec différentes structures communautaires initiales n'affecte pas beaucoup la structure communautaire finale, nous avons choisi l'algorithme de Louvain [6] et celui de Clauset-Newman [14] pour former la structure communautaire initiale sur les trois réseaux vus précédemment (les réseaux récepteurs). Nous appliquons ensuite notre algorithme et testons les valeurs de densité de la modularité renvoyées. La figure 5.4 montre que même si notre algorithme commence par différentes structures communautaires initiales, il tend à trouver des densités de la modularité proche.

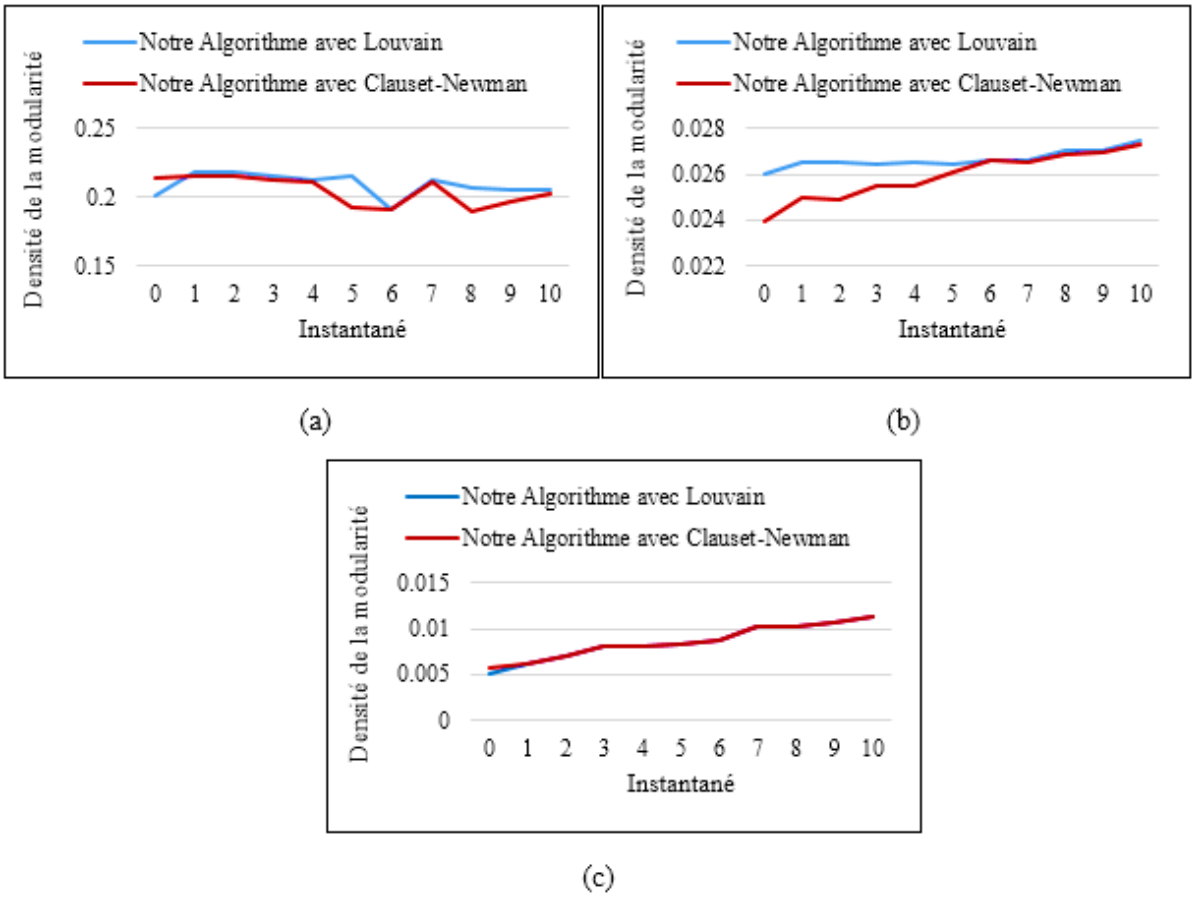

Figure 5. 4 Résultats concernant la densité de la modularité obtenus en utilisant deux structures communautaires initiales issues de deux algorithmes différents sur les réseaux *Jazz musicians* (a), *Hamsterster friendships* (b) et *Facebook* (c).

Grace à cette expérimentation menée sur les trois réseaux réels décrits ci-dessus, nous pouvons dire que :

- Notre algorithme est capable de détecter et de découvrir plus de communautés dans un temps raisonnable avec une meilleure modularité et densité de la modularité.

- Notre approche tend à trouver la même structure communautaire finale même si elle commence avec des structures communautaires initiales différentes.

5.3 Tests de qualité, stabilité et de validité de la structure communautaire

Cette partie d'expérimentation est destinée particulièrement aux tests concernant la structure communautaire. Les tests ont été réalisés sur trois sous-parties :

1) Tests de qualité de la structure communautaire dont la modularité [38] et la densité de la modularité [60] ont été utilisés comme des métriques d'évaluation.
2) Tests de stabilité de la structure communautaire en utilisant l'information mutuelle [29] comme mesure.
3) Tests de validité de la structure communautaire en évaluant les valeurs de l'information mutuelle normalisée [29] renvoyés.

Les réseaux utilisés durant cette partie d'expérimentation sont : « Zachary karate club », « American College football », « Polblogs » et « Hamsterster full ».

-**Zachary karate club** [59] est un réseau social de 78 relations d'amitiés entre 34 membres d'un club de karaté, dans une université aux Etats-Unis, au courant des années 1970.

-**American College football** [21] de (NCAA) National Collegiate Athletic Association est un réseau social avec des conférences d'équipe de football universitaires américaines. Le réseau représente le calendrier des matchs de la division I-A pour la saison 2000. Il est composé de 115 nœuds et de 613 liens regroupés en 12 équipes. Les nœuds du réseau représentent les équipes et les liens représentent les matchs entre les deux équipes qui ont joué ensemble.

-**Polblogs** [1] est un réseau de blogs politiques qui comprend 1 490 blogs sur l'élection présidentielle américaine de 2004 avec 16 718 liens. Les blogs ont été divisés manuellement en catégories conservatrice et libérale.

-**Hamsterster full** est un réseau social qui représente les relations d'amitiés et liens familiaux entre 2 426 utilisateurs de site web hamsterster.com. Le réseau est sélectionné de KONECT.

Afin de former un réseau dynamique, Y% des nœuds d'un réseau du monde réel sont choisis aléatoirement pour former le réseau initial. Les Z% des nœuds restants sont ajoutés à notre réseau sur S instants pour produire à la fin le réseau du monde réel. Sur chaque réseau sélectionné, nous ajoutons un nombre de nœuds avec une variété de nombre d'arêtes sur chaque instantané. Les résultats retournés sont comparés avec deux algorithmes à savoir :

-L'algorithme statique de Louvain [6] basé sur l'optimisation de la modularité.

-L'algorithme dynamique LabelRankT [54] basé sur la propagation d'étiquette.

5.3.1 Test de qualité de la structure communautaire

Afin d'évaluer la qualité de la structure communautaire, nous avons utilisé la métrique de modularité (Q) [38] et de la densité de la modularité (Q_{ds}) [60].

-70 % des nœuds du réseau *Zachary karate club* sont choisis aléatoirement pour former le réseau initial. Les 30 % des nœuds restants sont rajoutés au réseau Zachary, ce qui donne un total de 5 instantanés du réseau. Nous ajoutons deux nœuds sur chaque instantané.

Notre expérimentation montre que notre algorithme renvoie de meilleures valeurs de la modularité et de la densité de la modularité sur les instantanés du réseau Zachary (Tableau 5.1) par rapport à Louvain et LabelRankT.

Temps (T)	Notre Algorithme		Louvain		LabelRankT	
Métriques	Q	Q_{ds}	Q	Q_{ds}	Q	Q_{ds}
T=1	**0.461**	0.2893	0.440	0.2514	0.444	**0.2928**
T=2	**0.421**	**0.2639**	0.416	0.2627	0.412	0.2618
T=3	**0.424**	**0.2571**	0.405	0.2534	0.417	0.2477
T=4	0.423	**0.2435**	**0.427**	**0.2435**	0.410	0.2362
T=5	**0.419**	**0.2301**	0.418	**0.2301**	0.344	0.1928

Tableau 5. 1 Les résultats concernant les valeurs de la modularité et de la densité de la modularité obtenues sur chaque instantané du réseau *Zachary Karate Club* par notre algorithme, l'algorithme de Louvain et l'algorithme LabelRankT.

A la fin d'ajout des nœuds au réseau *Zachary*, nous avons obtenu un réseau de 34 nœuds et de 78 liens avec un nombre de communautés égal à 4. Le changement du réseau est illustré aux figures (5.5, 6, 7, 8, 9) aux instants T=1, 2, 3, 4, 5.

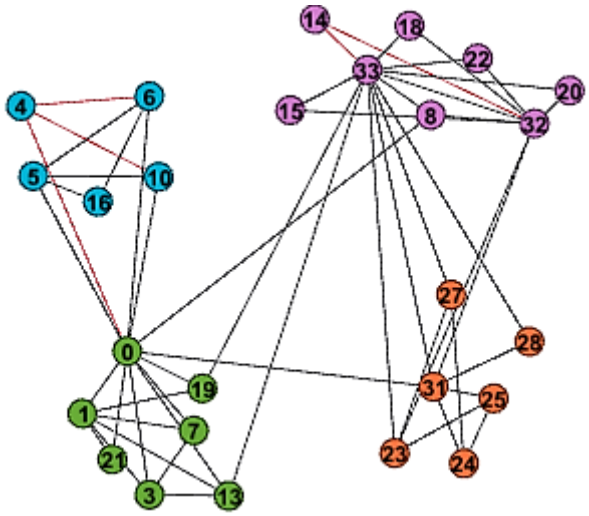

Figure 5. 5 [13] Capture d'écran du réseau dynamique *Zachary karate club* à T=1 obtenue par notre algorithme.

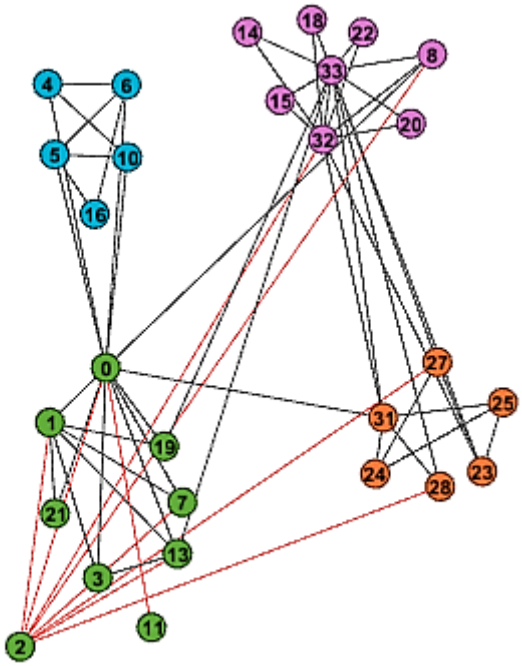

Figure 5. 6 [14] Capture d'écran du réseau dynamique *Zachary karate club* à T=2 obtenue par notre algorithme.

[13] *Ajout des nœuds 14 et 4 avec leurs liens rouges à l'instant T=1 au réseau initial Zachary karate club constitué de 24 nœuds et 48 liens*

[14] *Ajout des nœuds 2 et 11 avec leurs liens rouges à l'instant T=2 au réseau Zachary karate club de T=1*

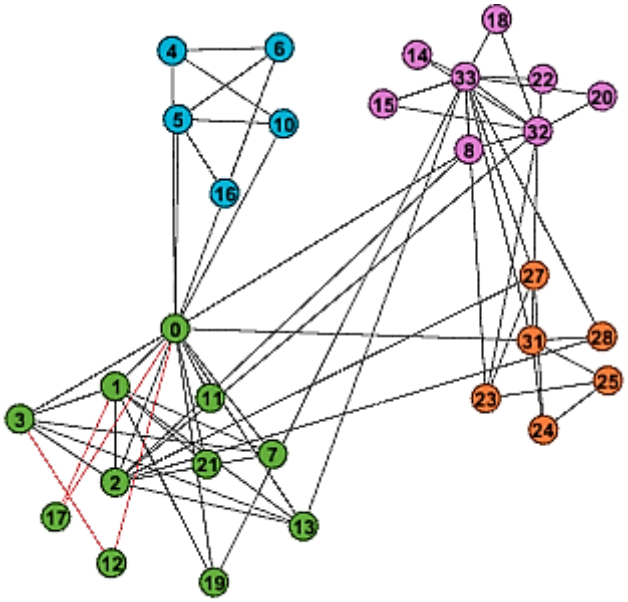

Figure 5. 7 [15] Capture d'écran du réseau dynamique *Zachary karate club* à T=3 obtenue par notre algorithme.

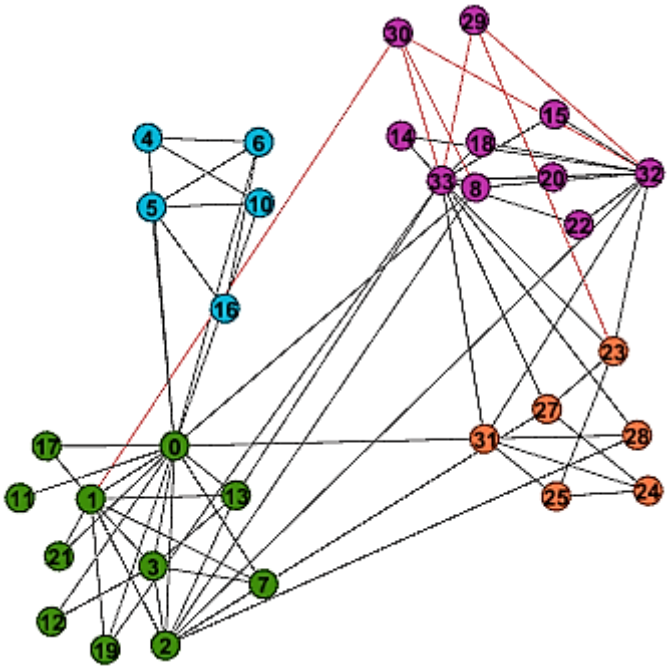

Figure 5. 8 [16] Capture d'écran du réseau dynamique *Zachary karate club* à T=4 obtenue par notre algorithme.

[15] *Ajout des nœuds 12 et 17 avec leurs liens rouges à l'instant T=3 au réseau Zachary karate club de T=2*

[16] *Ajout des nœuds 30 et 29 avec leurs liens rouges à l'instant T=4 au réseau Zachary karate club de T=3*

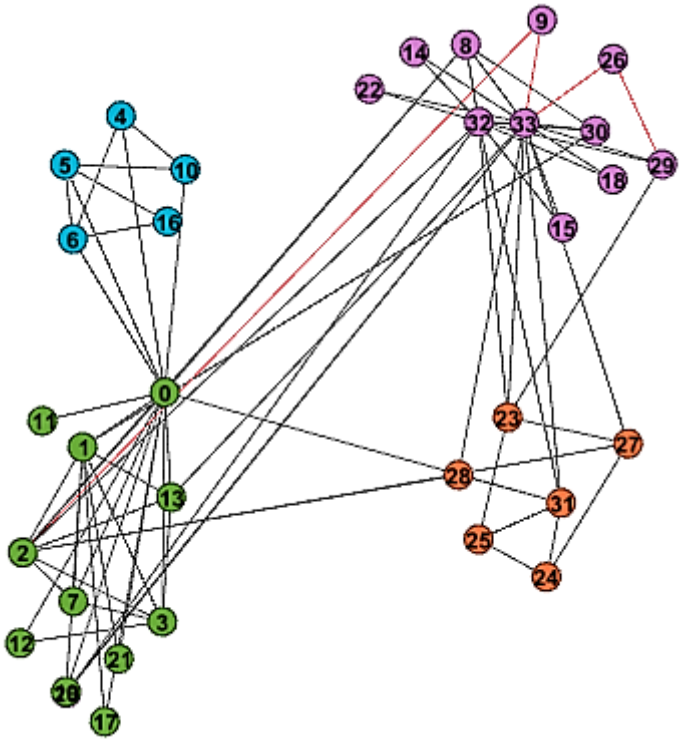

Figure 5. 9 [17] Capture d'écran du réseau dynamique *Zachary karate club* à T=5 obtenue par notre algorithme.

-Relativement au réseau *American College football*, 50 % des nœuds du réseau ont été choisis aléatoirement pour former notre réseau initial. Les 50 % des nœuds restants sont ajoutés au réseau initial, 5 nœuds à chaque instantané ce qui forme 12 instantanés de réseau.

Notre algorithme a la capacité de mettre à jour la structure du réseau dynamique avec une meilleure modularité et densité de la modularité par rapport à Louvain et LabelRankT (Tableau 5.2).

[17] *Ajout des nœuds 26 et 9 avec leurs liens rouges à l'instant T=5 au réseau Zachary karate club de T=4*

Temps (T)	Notre Algorithme		Louvain		LabelRankT	
Métriques	Q	Q_{ds}	Q	Q_{ds}	Q	Q_{ds}
T=1	**0.642**	**0.4409**	0.640	0.4022	0.548	0.0907
T=2	0.634	**0.4332**	**0.641**	0.3954	0.550	0.0907
T=3	**0.625**	**0.4237**	0.622	0.3826	0.585	0.0987
T=4	**0.634**	**0.4519**	0.629	0.3466	0.563	0.0945
T=5	**0.638**	**0.4467**	0.633	0.4038	0.525	0.0834
T=6	0.627	0.4304	**0.628**	**0.4394**	0.577	0.0842
T=7	**0.608**	0.4211	0.601	**0.4365**	0.537	0.0802
T=8	**0.609**	**0.4139**	0.600	0.3894	0.468	0.0742
T=9	**0.605**	**0.4262**	0.602	0.3907	0.496	0.0801
T=10	0.601	**0.4555**	**0.613**	0.3769	0.435	0.0750
T=11	0.585	**0.4484**	**0.603**	0.3942	0.431	0.0643
T=12	0.564	0.4393	**0.604**	**0.4495**	0.467	0.0789

Tableau 5. 2 Les résultats concernant les valeurs de la modularité et de la densité de la modularité obtenues sur chaque instantané du réseau *American College football* par notre algorithme, l'algorithme de Louvain et l'algorithme LabelRankT.

Les figures (5.10, 11, 12, 13) montrent quelques captures d'écran sur le réseau dynamique *American College football* aux instants T=1, 2, 5, 12.

La figure 5.10 illustre le réseau *American College football* à l'instant T=1 après l'ajout de 5 nœuds et de leurs liens au réseau initial football avec un nombre de communauté égale à 8.

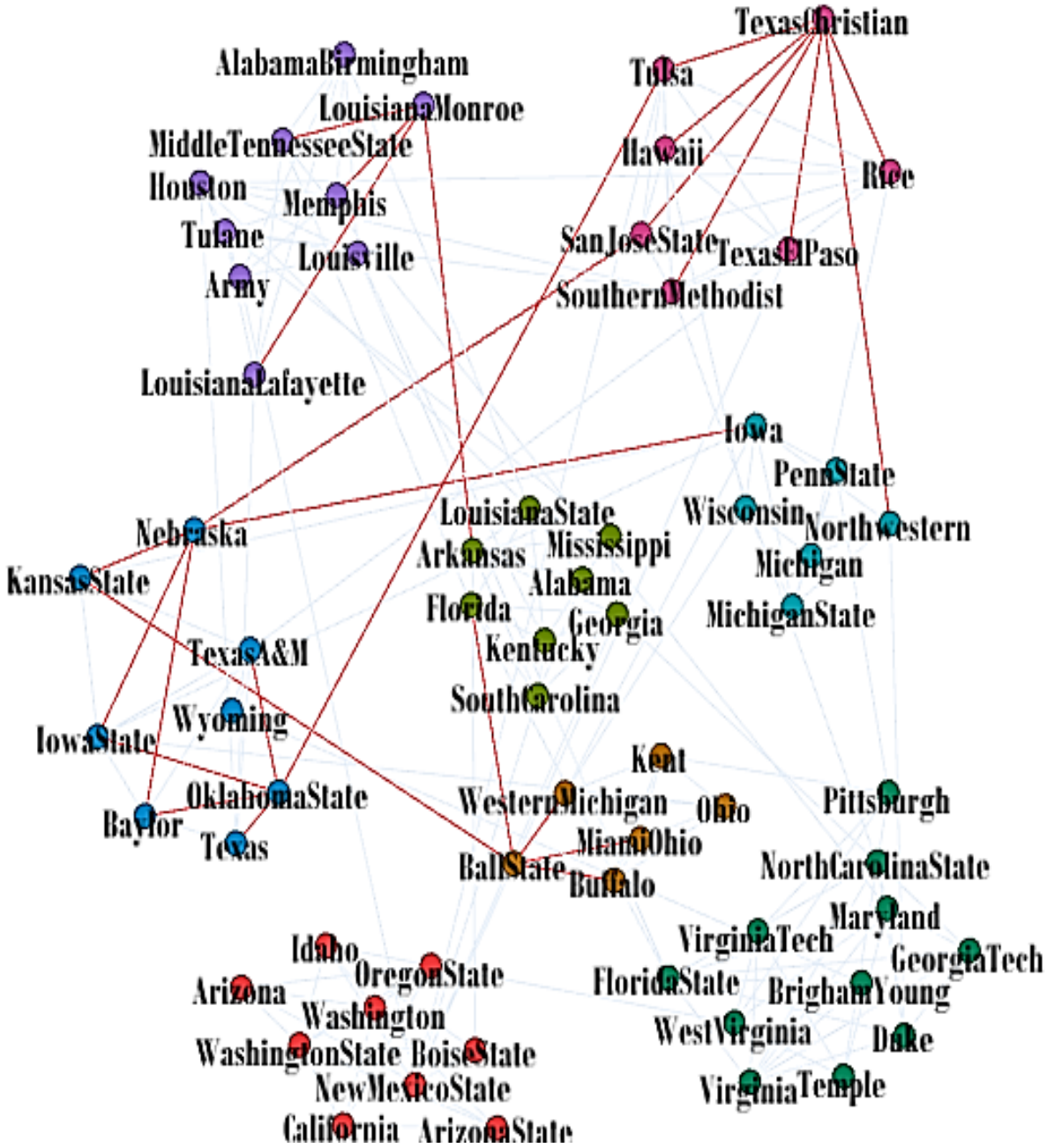

Figure 5. 10 [18] Capture d'écran du réseau dynamique *American College football* à T=1 obtenue par notre algorithme.

La figure 5.11 illustre le réseau *American College football* à l'instant T=2 après l'ajout de 5 nœuds et de leurs liens au réseau football de T=1. On remarque la naissance de la communauté verte constituée de (*Wyoming, Nevada las Vegas, San Diego State, Brigham Yong*) ce qui forme 9 communautés.

[18] *Les liens rouges représentent les liens ajoutés à l'instant T=1*

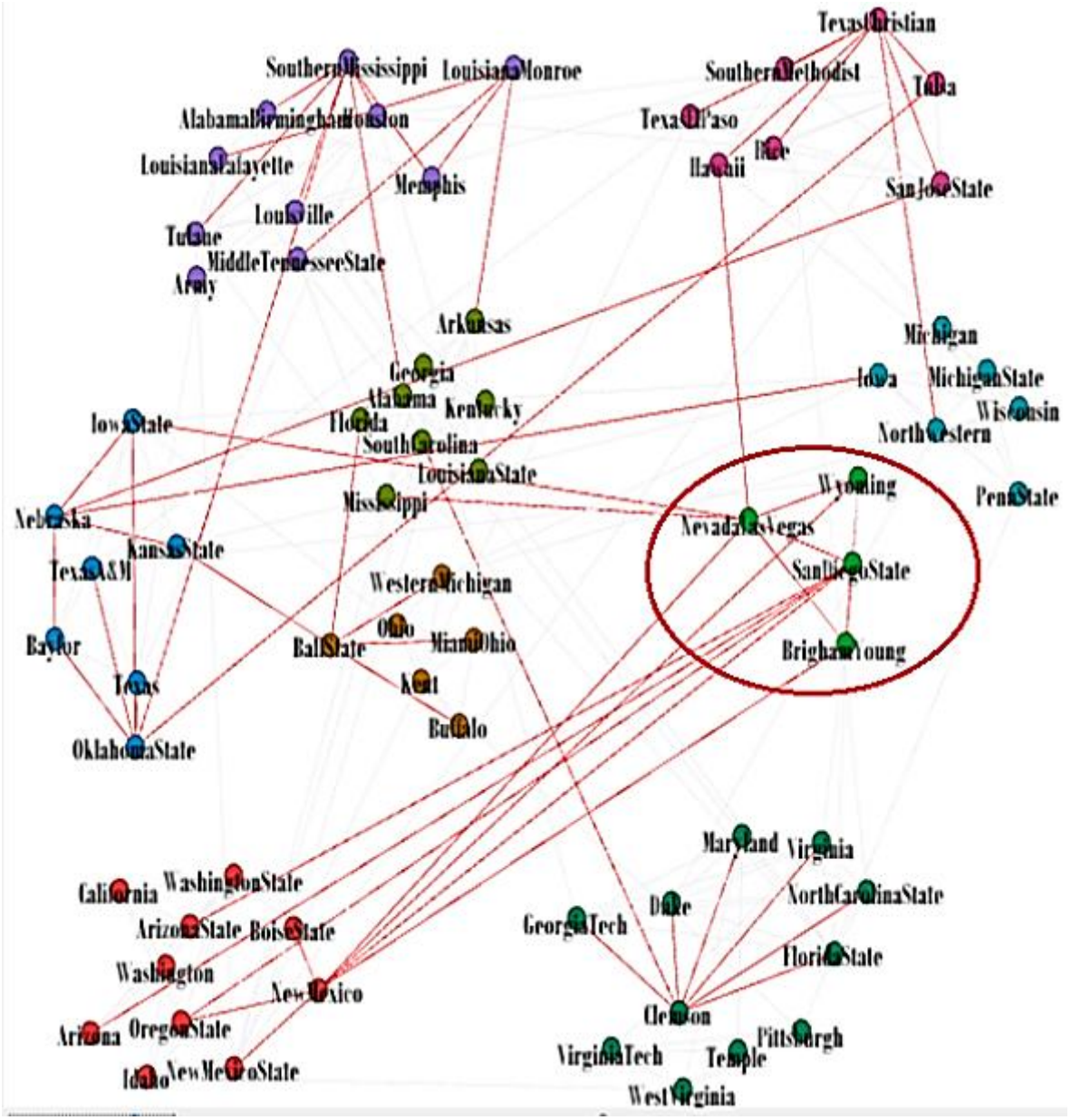

Figure 5. 11 [19] Capture d'écran du réseau dynamique *American College football* à T=2 obtenue par notre algorithme.

Quant à la figure 5.12, après l'ajout des nœuds et de leurs liens à T=5, nous avons obtenu un réseau de 10 communautés. La dixième communauté qui est la communauté orange comprend les nœuds (California, Washington, Washington State, Oregon, Arizona State, Oregon State, Arizona). Les autres communautés ont été renforcées par l'ajout des nœuds et de leurs liens.

[19] *Les liens rouges représentent les liens ajoutés à T=1 et à T=2*

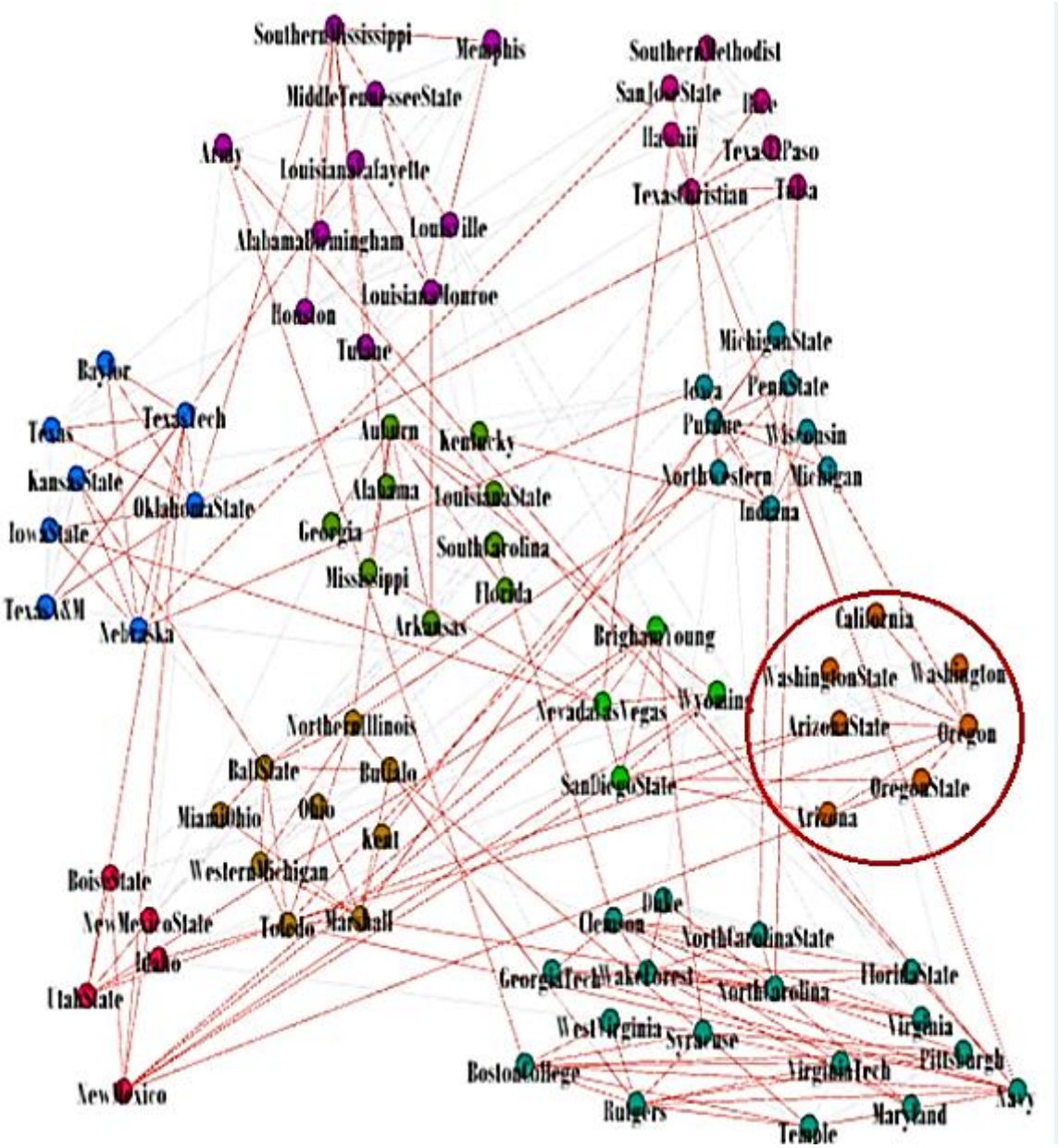

Figure 5. 12 [20] Capture d'écran du réseau dynamique *American College football* à T=5 obtenue par notre algorithme.

Enfin, la figure 5.13 montre le réseau final *American College football* de 115 nœuds et 613 liens avec un nombre de communauté égale à 11. Dix communautés ont été renforcées par l'ajout des nœuds et de leurs liens de T=6 jusqu'à T=12 et une communauté est née (la communauté jeune).

[20] *Les liens rouges représentent les liens ajoutés à T=1, T=2, T=3, T=4 et T=5*

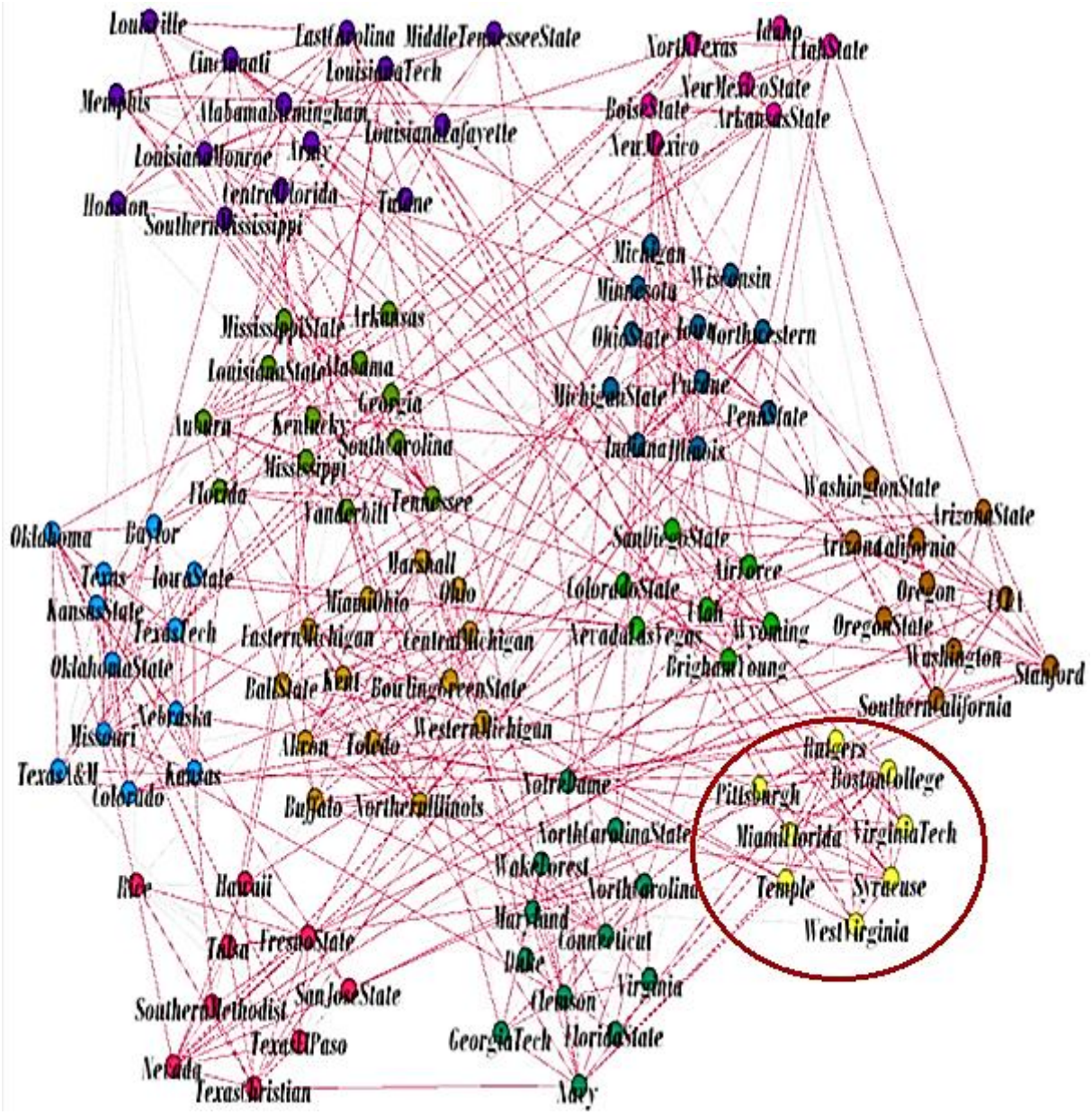

Figure 5. 13 [21]Capture d'écran du réseau dynamique *American College Football* à T=12 obtenue par notre algorithme.

-Quant au réseau *Polblogs*, 62 % des nœuds du réseau forment notre réseau initial. L'évolution du réseau initial est simulée via une série de 14 instantanés pour l'ajout des nœuds restants (38 %) avec leurs liens, 40 nœuds à chaque instantané. Il est apparu que dans la plupart des instantanés, notre algorithme retourne de meilleures valeurs de modularité et de densité de la modularité en comparaison avec Louvain et LabelRankT (Tableau 5.3).

[21] *Les liens rouges représentent les liens ajoutés à T=1 jusqu'à T=12*

Temps (T)	Notre Algorithme		Louvain		LabelRankT	
Métriques	Q	Q_{ds}	Q	Q_{ds}	Q	Q_{ds}
T=1	**0.4222**	0.0572	0.4205	**0.0576**	0.4152	0.0468
T=2	**0.4230**	**0.0564**	0.4223	0.0555	0.4165	0.0450
T=3	**0.4233**	0.0544	0.4227	**0.0547**	0.4168	0.4385
T=4	0.4231	0.0531	**0.4236**	**0.0537**	0.4202	0.0418
T=5	0.4233	**0.0535**	**0.4240**	0.0531	0.4221	0.0406
T=6	**0.4245**	0.0516	0.4243	**0.0522**	0.4243	0.0389
T=7	0.4251	**0.0539**	**0.4253**	0.0535	0.4239	0.0377
T=8	**0.4256**	**0.0540**	0.4254	0.0537	0.4249	0.0359
T=9	**0.4262**	**0.0530**	0.4261	**0.0530**	0.4261	0.0348
T=10	0.4251	**0.0522**	0.4248	**0.0522**	**0.4263**	0.0332
T=11	0.4260	**0.0520**	0.4258	0.0490	**0.4263**	0.0325
T=12	0.4238	**0.0491**	0.4242	0.0476	**0.4263**	0.0311
T=13	0.4243	**0.0408**	**0.4249**	0.0406	0.4246	0.0305
T=14	**0.4271**	**0.0412**	0.4268	0.0409	0.4261	0.0297

Tableau 5. 3 Les résultats concernant les valeurs de la modularité et de la densité de la modularité obtenues sur chaque instantané du réseau *Polblogs* par notre algorithme, l'algorithme de Louvain et l'algorithme LabelRankT.

-Au sujet du réseau *Hamsterster full*, 66.7 % des nœuds du réseau ont été sélectionnés d'une manière aléatoire. Les 33.3 % des nœuds restants sont ajoutés sur 9 instantanés (100 nœuds à chaque instantané) pour produire à la fin un réseau *Hamsterster full* de 2 426 nœuds et de 16 631 liens.

Le tableau (Tableau 5.4) montre que les valeurs de la modularité et de la densité de la modularité retournées par notre méthode sont meilleures par rapport à celles obtenues par LabelRankT. D'autre part, sur certains autres instantanés, la modularité de Louvain est un peu plus élevée.

Temps (T)	Notre Algorithme		Louvain		LabelRankT	
Métriques	Q	Q_{ds}	Q	Q_{ds}	Q	Q_{ds}
T=1	**0.588**	0.0901	0.587	0.0899	0.462	**0.1435**
T=2	**0.583**	**0.1432**	0.580	0.0901	0.438	0.1332
T=3	0.569	**0.1356**	**0.579**	0.1231	0.439	0.1092
T=4	0.561	0.1330	**0.562**	**0.1342**	0.470	0.0877
T=5	**0.560**	**0.1101**	0.558	0.0954	0.462	0.0794
T=6	**0.566**	**0.0934**	**0.566**	0.0911	0.460	0.0819
T=7	0.551	0.0789	**0.562**	**0.0865**	0.451	0.0647
T=8	**0.564**	**0.0910**	0.563	0.0786	0.470	0.0552
T=9	0.560	**0.0973**	**0.561**	0.0754	0.472	0.0512

Tableau 5. 4 Les résultats concernant les valeurs de la modularité et de la densité de la modularité obtenues sur chaque instantané du réseau *Hamsterster full* par notre algorithme, l'algorithme de Louvain et l'algorithme LabelRankT.

5.3.2 Test de stabilité de communautés

La stabilité est un moyen d'analyser l'évolution des communautés. Cette mesure est basée sur la similarité entre différentes communautés sur deux instantanés consécutifs. Afin de mesurer cette stabilité, nous adoptons une métrique bien connue en théorie de l'information appelée l'Information Mutuelle [29] qui compte le nombre de bits partagés par deux variables aléatoires. Etant données deux partitions X et Y, l'information mutuelle entre X et Y est définie par la formule (5.1) suivante :

$$IM(X,Y) = \sum\sum p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \qquad (5.1)$$

$IM(X,Y) = H(X) - H(X|Y)$, quand $H(X) = -\sum_x p(x)\log p(x)$ est l'entropie de Shannon de x et $H(X|Y) = -\sum_{x,y} p(x,y)\log p(x|y)$ est l'entropie conditionnelle de X par rapport à la partition Y .

Dans notre mesure de la stabilité, nous voulons définir une mesure décrivant comment les communautés détectées partagent les mêmes nœuds entre deux instantanés consécutifs. Des valeurs plus élevées d'informations mutuelles signifient que les communautés entre deux instantanés consécutifs partagent les mêmes nœuds tandis que des valeurs plus basses signifient que les communautés détectées ont changé entre les instantanés.

Dans cette partie de l'expérimentation, l'Information Mutuelle est calculée entre chaque partition et sa précédente sur les quatre réseaux vus précédemment (sous-section 5.3) et les résultats obtenus sont comparés avec la méthode de Louvain et LabelRankT.

La figure 5.14 représente les résultats de la stabilité des communautés sur les quatre réseaux considérés. Notre méthode retourne de meilleures valeurs pour l'Information Mutuelle sur chaque instantané de réseau. Cela signifie que les communautés détectées par notre méthode sont plus stables entre les instantanés que celles calculées avec Louvain et LabelRankT.

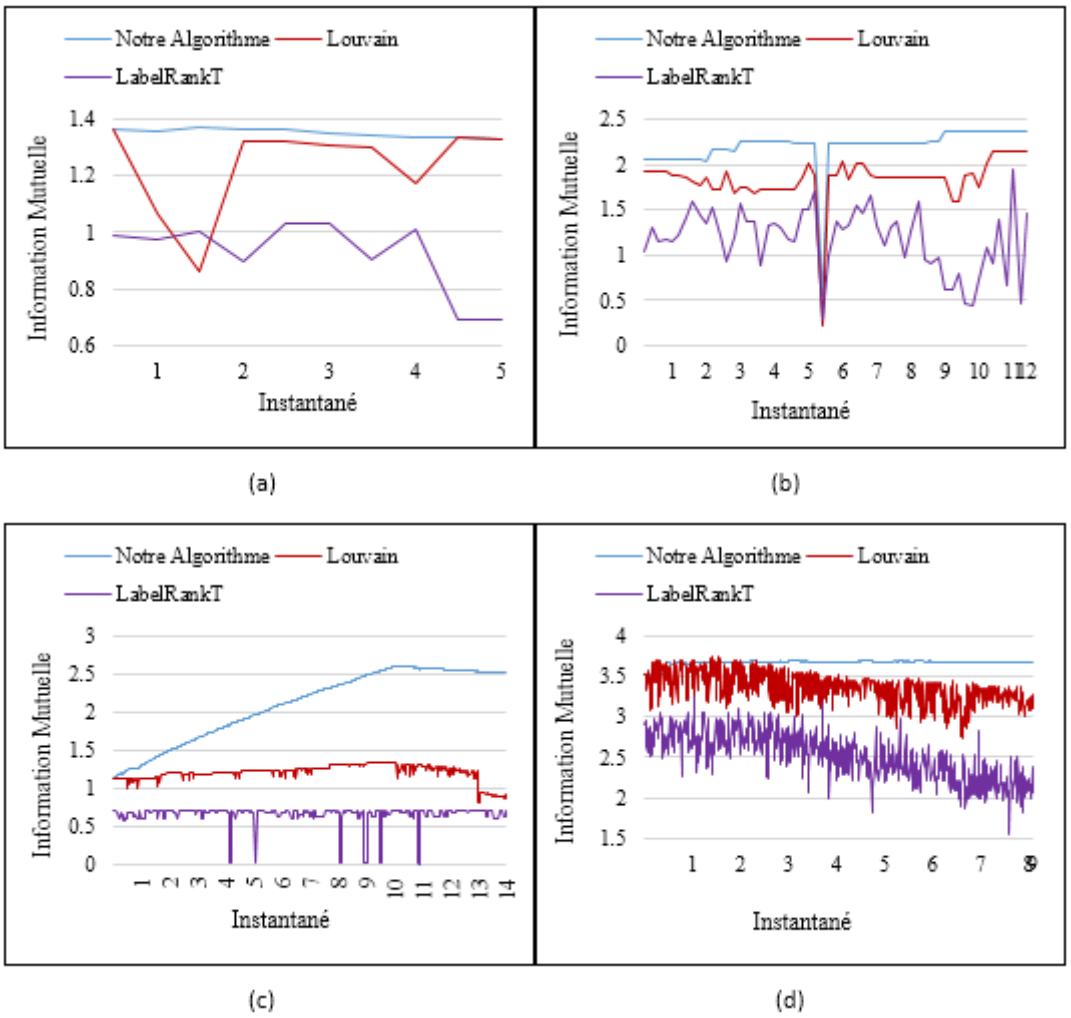

Figure 5. 14 Les résultats obtenus concernant la stabilité des communautés pour les réseaux *Zachary karate club* (a), *American College football* (b), *Polblogs* (c) et *Hamsterster full* (d), les courbes représentent l'Information Mutuelle sur chaque instantané des quatre réseaux considérés.

5.3.3 Test de validité de la structure communautaire

Afin d'évaluer la validité de la structure communautaire, nous utilisons l'Information Mutuelle Normalisée (IMN). IMN est utilisée pour mesurer la similarité entre deux partitions, à savoir, la partition obtenue par notre méthode et la partition cible. L'Information Mutuelle Normalisée [29] prend une valeur comprise entre 0 et 1, égale à 1 si les partitions sont identiques et égale à 0 si les partitions sont complètement différentes.

L'Information Mutuelle Normalisée est définie par la formule (5.2) suivante :

$$IMN = \frac{2IM(X,Y)}{H(X) + H(Y)} \tag{5.2}$$

Durant cette expérimentation nous n'avons pas pu calculer l'IMN sur le réseau *Hamsterster full* en raison du manque d'informations appropriées sur la structure communautaire réelle. On constate que l'IMN retournée par notre algorithme sur le réseau karaté est identique avec Louvain et moins élevée en comparaison avec LabelRankT, mais elle se rapproche de 1.
D'autre part, notre algorithme retourne une meilleure IMN sur les deux réseaux *American College football* et *Polblogs* en comparaison avec les résultats de la méthode statique de Louvain et de la méthode dynamique LabelRankT (Tableau 5.5).

Réseaux	Zachary Karate club	American College football	PolBlogs
Métriques	IMN	IMN	IMN
Notre Algorithme	0.618	**0.888**	**0.678**
Louvain	0.618	0.885	0.628
LabelRankT	**0 .732**	0.345	0.677

Tableau 5. 5 Les résultats concernant la modularité, la densité de modularité et l'IMN obtenus par notre algorithme, l'algorithme de Louvain et l'algorithme LabelRankT sur les trois réseaux réels considérés.

Les résultats expérimentaux montrent que notre algorithme produit des structures communautaires de qualité. De même, il identifie des communautés stables (deux communautés entre deux instantanés consécutives partagent les mêmes nœuds). Enfin, l'approche tend à trouver des structures communautaires valables vu que celles-ci se rapprochent aux structures communautaires réelles.

5.4 Évaluation dans le réseau de citations HEP-TH

Cette section est consacrée au test de réseau incrémental HEP-TH ou les nœuds et les liens sont ajoutés au réseau et ne peuvent pas être supprimés ultérieurement. Le but de ce test est de montrer que notre approche atténue le problème de la limite de résolution. Pour ce faire, nous avons choisi la métrique densité de la modularité [60]. Cette mesure a été définie dans la sous-section 4.3.1 du chapitre

précédent pour résoudre le problème de la limite résolution engendré par l'optimisation de la modularité. Nous avons comparé les résultats de notre algorithme aux résultats de deux autres algorithmes dynamiques à savoir :

-L'algorithme « LabelRankT » basé sur la propagation d'étiquettes précédemment utilisé dans la sous-section 5.3.1.

-L'algorithme « Dynamic Louvain » [15] qui est une modification de l'algorithme de Louvain [6] basé sur l'optimisation locale de la modularité.

Le réseau utilisé qui est le réseau de citations HEP-TH (high energy physics theory) [61] d'e-print arXiv couvre toutes les citations dans un jeu de données de 27 770 articles et de 352 807 liens[22] de janvier 1992 à mai 2003. Dans notre expérience, les nœuds et leurs liens ont été ajoutés en fonction de la date de soumission. Les liens de citation des 9 premières années de 1992 à 2000 ont été pris en compte pour constituer la structure communautaire de base de notre algorithme. Pour simuler l'évolution du réseau, 29 instantanés (mois) sont créés de janvier 2001 à mai 2003.

Nous comparons les résultats de densité de la modularité obtenus par notre algorithme à chaque instantané de réseau avec la méthode LabelRankT [54] ainsi qu'avec la méthode Dynamic Louvain [15]. La figure 5.15 montre que les valeurs de densité de la modularité renvoyées par notre méthode sont plus élevées sur quelques instantanés que celles de la méthode Dynamic Louvain et proches sur d'autres instantanés. En comparaison avec la méthode LabelRankT, elles sont performantes sur tous les instantanés. Cela peut s'expliquer par le fait que notre méthode maximise la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées, ce qui permet d'atténué le problème de la limite de résolution.

[22] *https://www.cs.cornell.edu/projects/kddcup/datasets.html*

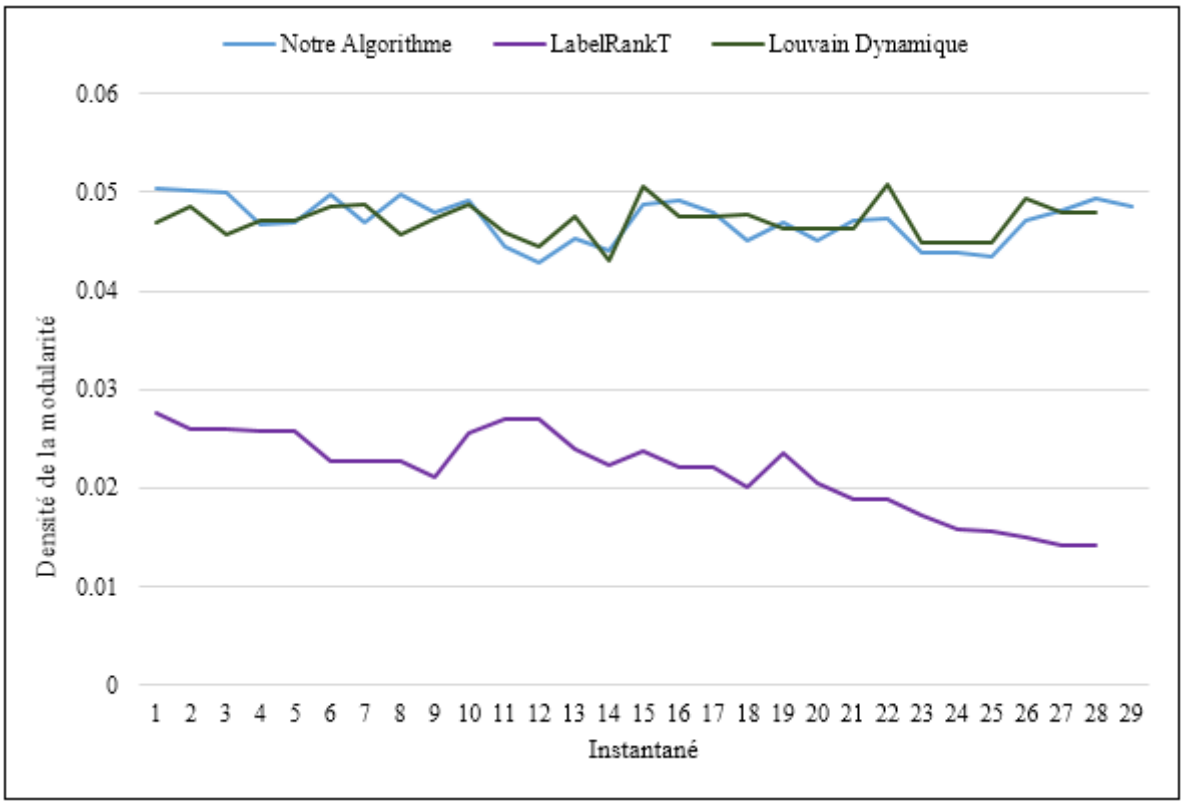

Figure 5. 15 Résultats obtenus concernant la densité de modularité par notre méthode, LabelRankT et Dynamic Louvain

5.5 Conclusion

Dans ce chapitre, nous avons présenté les résultats expérimentaux de notre approche. Nous avons testé nos contributions sur différentes collections de données du monde réel. Les résultats ont été comparés avec les algorithmes de détection de communauté statique et dynamique. Les tests montrent que notre algorithme est meilleur que les algorithmes statiques en termes de densité de la modularité, temps d'exécution, modularité et le nombre de communautés découvertes. Il surpasse les algorithmes dynamiques en termes de stabilité des communautés et de densité de la modularité. Ceci montre l'efficacité et la capacité de notre méthode à détecter des communautés dynamiques et à suivre leur évolution par l'utilisation de la structure communautaire précédente.

Conclusion générale

Méthode proposée vs. Certaines méthodes rapportées dans l'état de l'art

Dans cette section nous rappelons les points essentiels de notre travail pour le situer par rapport à certains travaux existants. En effet, l'étude que nous avons menée a montré que les méthodes utilisées pour les réseaux statiques, qu'elles soient agglomératives [6] [14], séparatives [43] [21], basées sur la propagation d'étiquettes [44] [22] ou sur d'autres critères [45], [42], produisent de bonnes structures communautaires, mais ne peuvent être appliquées fréquemment sur des réseaux dynamiques particulièrement à cause de la difficulté à prendre en considération le réseau tout entier. Par rapport à ces méthodes, notre proposition se démarque par un certain nombre de points importants. Tout d'abord, il faut noter que notre méthode, tout comme celles citées ci-dessus, produit de bonnes structures communautaires. D'autre part, il faut également souligner qu'elle se distingue par le fait qu'elle s'applique aisément et fréquemment sur des réseaux dynamiques évoluant par l'ajout de nœuds et d'arêtes moyennant des complexités algorithmiques relativement faibles.

Parmi les méthodes dédiées à la détection de communautés dynamiques rapportées dans l'état de l'art, les premières méthodes utilisent des algorithmes statiques [24] et [52] qu'elles adaptent aux réseaux dynamiques. Ces méthodes se sont avérées vite inefficaces et inadaptées à cause de leurs complexités temporelles particulièrement très élevées.

Notre méthode est typiquement incrémentale comme nous l'avons déjà mentionné dans la section 4.2 du Chapitre 4 où est décrite notre méthode.

Pour rappel, les méthodes dites incrémentales s'appuient sur le principe de l'utilisation de la structure communautaire précédente pour détecter la structure communautaire suivante. Ces méthodes se distinguent par leur critère. En effet, dans [33] le nombre de communautés à détecter est fixé à l'avance. Dans [47] on met à

jour la structure communautaire par l'ajout de différents types d'arêtes. Les auteurs dans [15] réalisent la même stratégie que [47], en supprimant différents types d'arêtes, mais de manière différente. Ils optimisent la modularité du réseau à chaque changement qui intervient. Rappelons que l'optimisation de la modularité souffre du problème de la limite de résolution dont le sens où les petites communautés peuvent disparaître à cause des regroupements avec d'autres communautés similaires ou bien elles sont absorbées par des communautés relativement beaucoup plus grandes.

La particularité de l'incrémentalité de notre méthode réside dans le fait qu'on n'a pas une connaissance préalable du nombre de communautés à détecter. Par ailleurs, l'approche propose d'optimiser la densité du réseau au lieu de la modularité. L'optimisation consiste en la maximisation de la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées, ce qui permet d'identifier un nombre de communautés et de résoudre le problème de limite de résolution.

Synthèse et perspectives

Dans cette thèse, nous avons proposé une nouvelle approche basée sur la densité avec une double optimisation pour le suivi de la structure communautaire dans des réseaux dynamiques. Notre approche est dédiée aux réseaux évoluant par ajout de nœud et de leurs liens. L'ajout d'un nouveau nœud et de ses liens se fait simultanément, ce qui diminue le cout de calcul. De plus, notre algorithme est incrémental dans le sens où il s'appuie sur le principe de l'utilisation de la structure communautaire précédente pour détecter la structure communautaire courante. Notre approche est caractérisée par l'utilisation d'une optimisation à deux niveaux de la densité du réseau.

Le premier niveau d'optimisation est l'œuvre d'un algorithme d'optimisation locale permettant d'intégrer un nœud avec ses liens à la communauté qui maximise la somme des différences entre la densité interne et la densité externe de toutes les communautés infectées.

Le second niveau d'optimisation vise à améliorer au mieux le score de la densité du réseau, ceci est réalisé par le test des opérations (fusion, éclatement ou naissance) sur les communautés touchées par le changement.

Pour valider notre approche, nous avons testé notre méthode (algorithmes) sur trois collections de données du monde réel.

La première collection comprend des données dont la structure communautaire est inconnue. L'expérimentation a montré que notre méthode donne de meilleurs résultats en termes de densité de la modularité, modularité, temps de traitement et de nombre de communautés découvertes par rapport à la méthode statique de Louvain. Nous avons aussi testé notre algorithme, en commençant par des structures communautaires initiales différentes. Les résultats retournés en termes de densité de la modularité indiquent que même si notre algorithme commence par des structures communautaires initiales différentes, il tend à trouver la même structure communautaire finale.

La seconde collection concerne des données dont la structure communautaire est connue. Les résultats expérimentaux montrent que notre méthode se comporte mieux que la méthode dynamique LabelRankT en qualité, validité de la structure communautaire obtenue et en stabilité de communauté dans laquelle les attributions de communauté de nœuds restent inchangées entre les instantanés.

Enfin, la dernière collection correspond au réseau incrémental HEP-TH. Les valeurs de densité de la modularité retournées par notre algorithme sont meilleures de celles de la méthode LabelRankt et Dynamic Louvain. Ces résultats nous renforcent dans l'idée que l'optimisation locale de la densité atténue le problème de limite de résolution par l'identification des grandes et petites communautés.

Les travaux et résultats présentés dans cette thèse offrent naturellement de nombreuses perspectives. La première, en l'occurrence découle logiquement du fait que nous n'avons traité que des graphes non-orientés et non-pondérés, quoiqu'il existe d'autres types de graphes qui peuvent simuler d'autres situations. Nous souhaitons dans un premier temps étendre l'approche proposée aux graphes pondérés et/ou orientés.

La deuxième perspective envisageable serait d'élargir la méthode de suivi au traitement de communautés chevauchantes, du fait que nous n'avons traité que les communautés disjointes.

Enfin, une autre perspective, qui semble être, une continuité logique de ce travail, serait d'étendre l'approche aux réseaux entièrement dynamiques. Ainsi, les nœuds et les liens dans un réseau dynamique peuvent être supprimés et non pas seulement ajoutés.

Bibliographie

[1] Adamic, L.A., & Glance, N. (2005) 'The political blogosphere and the 2004 US Election', In Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem.

[2] Aynaud, T., Fleury, E., Guillaume, J. L., & Wang, Q. (2013) 'Communities in evolving networks: definitions, detection, and analysis techniques', In *Dynamics On and Of Complex Networks,* Vol. 2, pp. 159-200.

[3] Barnes E.R. (1982) 'An Algorithm for Partitioning the Nodes of a Graph', *SIAM Journal on Algebraic and Discrete Methods*, Vol. 3, No. 4, pp.541-550.

[4] Beiró, M. G., Busch, J. R., & Alvarez-Hamelin, J. I. (2010) 'Visualizing communities in dynamic networks', In *LAWDN-Latin-American Workshop on Dynamic Networks*.

[5] Blondel, V. D., & Senellart, P. P. (2002) 'Automatic extraction of synonyms in a dictionary', *In the SIAM Workshop on Text Mining*, Vol.1.

[6] Blondel, V. D., Guillaume, J., Lambiotte, R., & Lefebvre E. (2008) 'Fast unfolding of communities in large networks', *Journal of Statistical Mechanics: Theory and Experiment*, Vol. 2008, No. 10.

[7] Bouhatem, F., Ait El Hadj, A., Souam, F. (2020) 'Density-based Approach with Dual Optimization for Tracking Community Structure of Increasing Social Networks', *IJAIT International Journal on Artificial Intelligence Tools*, Vol. 29, No. 01.

[8] Boujlaleb, L., Idarrou, A., & Mammass, D. (2017) 'Tracking community evolution in social networks', *journal of theoretical & applied information technology,* Vol. 95, No. 22.

[9] Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., & Wagner, D. (2007, June) 'On finding graph clusterings with maximum modularity', In *International Workshop on Graph-Theoretic Concepts in Computer Science*, *Springer*, pp. 121-132.

[10] Cazabet, R., Amblard, F., & Hanachi, C. (2010) 'Detection of overlapping communities in dynamical social networks', *In 2010 IEEE second international conference on social computing. IEEE,* pp. 309-314.

[11] Chakrabarti, D., Kumar, R., & Tomkins, A. (2006) 'Evolutionary clustering', *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.554-560.

[12] Chan, S. Y., Hui, P., & Xu, K. (2009, February) 'Community detection of time-varying mobile social networks', In *International Conference on Complex Sciences*, *Springer*, pp. 1154-1159.

[13] Chen, Z., Wilson, K. A., Jin, Y., Hendrix, W., & Samatova, N. F. (2010, December) 'Detecting and Tracking Community Dynamics in Evolutionary Networks', In *2010 IEEE International Conference on Data Mining Workshops. IEEE*, pp. 318-327.

[14] Clauset, A., Newman, M. E. J., & Moore, C. (2004) 'Finding community structure in very large networks', *Phys.Rev.E,* Vol. 70, No. 6.

[15] Cordeiro, M., Sarmento, R.P., & Gama, J. (2016) 'Dynamic community detection in evolving networks using locality modularity optimization', *Social Network Analysis and Mining*, Vol. 6, No 1, pp.1.

[16] Dakiche, N., Tayeb, F. B. S., Slimani, Y., & Benatchba, K. (2019) 'Tracking community evolution in social networks: A survey'*, Information Processing & Management,* Vol. 56, No. 3, pp. 1084-1102.

[17] Egghe, L., & Rousseau, R. (1990) 'Quantitative methods in library, documentation and information science', *Elsevier Science*.

[18] Faloutsos, M., Faloutsos, P., & Faloutsos, C. (1999, August) 'On power-law relationships of the internet topology', In *ACM SIGCOMM computer communication review. ACM*, Vol. 29, No. 4, pp. 251-262.

[19] Fortunato, S. (2010) 'Community detection in graphs', *Physics Reports*, Vol. 486, No. 3, pp.75-174.

[20] Fortunato, S., & Barthelemy, M. (2007) 'Resolution limit in community detection', *PNAS*, Vol. 104, No. 1, pp.36-41.

[21] Girvan, M., Newman, M.E.J. (2002) 'Community structure in social and biological networks', *Proceedings of the National Academy of Sciences*, Vol. 99, No. 12, pp.7821.

[22] Gregory, S. (2010) 'Finding overlapping communities in networks by label propagation', *New Journal of Physics*, Vol. 12, No. 10.

[23] Han, J., Li, W., Zhao, L., Su, Z., Zou, Y., & Deng, W. (2017) 'Community detection in dynamic networks via adaptive label propagation', *PloS one*, Vol. 12, No. 11.

[24] Hopcroft, J., Khan, O., Kulis, B., & Selman, B. (2004) 'Tracking evolving communities in large linked networks', *Proceedings of the national academy of sciences of the United States of America*, Vol. 101, pp.5249-5253.

[25] Hu, Y., Chen, H., Zhang, P., Li, M., Di, Z., & Fan, Y. (2008) 'Comparative definition of community and corresponding identifying algorithm', *Physical Review E*, Vol. 78, No. 2.

[26] Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., & Barabási, A. L. (2000) 'The large-scale organization of metabolic networks'. *Nature*, Vol. 407, No. 6804, pp. 651.

[27] Kanawati, R. (2014) 'Seed-centric approaches for community detection in complex networks', *Springer International Publishing*, Vol. 8531, pp.197-208.

[28] Kernighan, B.W., & Lin, S. (1970) 'An efficient heuristic procedure for partitioning graphs', *Bell System Technical Journal*, Vol. 49, No. 2, pp.291-307.

[29] Lancichinetti, A., & Fortunato, S. (2009) 'Community detection algorithms: A comparative analysis', *Physical review. E*, Vol. 80, No. 5.

[30] Lancichinetti, A., & Fortunato, S. (2011) 'Limits of modularity maximization in community detection', *Physical review E*, Vol. 84, No. 6.

[31] Li, J., Huang, L., Bai, T., Wang, Z., & Chen, H. (2012, May) 'Cdbia: a dynamic community detection method based on incremental analysis', In *2012 International Conference on Systems and Informatics (ICSAI2012).* IEEE, pp. 2224-2228.

[32] Lin, Y. R., Chi, Y., Zhu, S., Sundaram, H., & Tseng, B. L. (2008, April) 'Facetnet: a framework for analyzing communities and their evolutions in dynamic networks', *In Proceedings of the 17th international conference on World Wide Web ACM,* pp. 685-694.

[33] Lin, Y.R., Chi, Y., Zhu, S., Sundaram, H., & Tseng, B.L. (2009) 'Analyzing communities and their evolutions in dynamic social networks', *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Vol. 3, No. 2, pp.8.

[34] Mancoridis, S., Mitchell, B. S., Rorres, C., Chen, Y., & Gansner, E. R. (1998) 'Using automatic clustering to produce high-level system organizations of source code', *Proceedings. 6th International Workshop on Program Comprehension. IWPC'98*, pp.45-52.

[35] Mauricio G, and Resende C, (2000) Detecting dense subgraphs in massive graphs, 17th international Symposium on Mathematical Programming.

[36] Milgram, S. (1997) 'The small world problem', *Psychology today*, Vol. 2, No. 1, pp 60-67.

[37] Mohar, B. (1997) 'Some applications of Laplace eigenvalues of graphs, Graph Symmetry: Algebraic Methods and Applications', *NATO ASI Series C*, Vol 497.

[38] Newman, M.E.J., & Girvan, M. (2004) 'Finding and evaluating community structure in networks', *Physical review E*, Vol. 69, No. 2.

[39] Nguyen, N. P., Dinh, T. N., Tokala, S., & Thai, M. T. (2011) 'Overlapping communities in dynamic networks: their detection and mobile applications', *In Proceedings of the 17th annual international conference on Mobile computing and networking. ACM,* pp. 85-96.

[40] Nguyen, N. P., Dinh, T. N., Xuan, Y., & Thai, M. T. (2011) 'Adaptive algorithms for detecting community structure in dynamic social networks', *In 2011 Proceedings IEEE INFOCOM. IEEE*, pp. 2282-2290.

[41] Palla, G., Barabási, A. L., & Vicsek, T. (2007) 'Quantifying social group evolution', *Nature,* Vol. 446, No. 7136, pp. 664.

[42] Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005) 'Uncovering the overlapping community structure of complex networks in nature and society'*, Nature*, Vol. 435, pp.814-818.

[43] Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004) 'Defining and identifying communities in networks', *Proceedings of the National Academy of Sciences*, Vol. 101, No. 9, pp.2658-2663.

[44] Raghavan, U.N., Albert, R., & Kumara, R. (2007) 'Near linear time algorithm to detect community structures in large-scale networks', *Physical Review E*, Vol. 76, No. 3.

[45] Rosvall, M., & Bergstrom, C.T. (2008) 'Maps of random walks on complex networks reveal community structure', *Proceedings of the National Academy of Sciences*, Vol. 105, No. 4, pp.1118-1123.

[46] Schaeffer, S.E. (2007) 'Graph clustering', *Computer Science Review*, Vol. 1, No. 1, pp. 27-64.

[47] Shang, J., Liu, L., Xie, F., Chen, Z., Miao, J., Fang, X., & Wu, C. (2014) 'A real-time detecting algorithm for tracking community structure of dynamic networks', *arXiv preprint arXiv,1407.2683*.

[48] Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y., & Schult, R. (2006, August) 'Monic: modeling and monitoring cluster transitions', In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining,* ACM. pp. 706-711.

[49] Tantipathananandh, C., & Berger-Wolf, T. Y. (2011, December) 'Finding communities in dynamic social networks', In *2011 IEEE 11th International Conference on Data Mining*. IEEE, pp. 1236-1241.

[50] Thomas, A., & Jean-Loup, G. (2010) 'Static community detection algorithms for evolving networks'. *WiOpt'10: Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pp. 508-514.

[51] Wang, Q. (2012) 'Overlapping community detection in dynamic networks', *Ecole normale supérieure de lyon-ENS LYON*.

[52] Wang, Y., Wu, B., & Du, N. (2008) 'Community evolution of social network: feature, algorithm and model', *arXiv preprint arXiv:0804.4356*.

[53] Xie, J., & Szymanski, B. K. (2013) 'Labelrank: A stabilized label propagation algorithm for community detection in networks', *In Proc. IEEE Network Science Workshop, West Point, NY*, pp.138–143.

[54] Xie, J., Chen, M., & Szymanski, B. K. (2013) 'LabelrankT: Incremental community detection in dynamic networks via label propagation', *In Proceedings of the Workshop on Dynamic Networks Management and* Mining. *ACM*, pp.25-32.

[55] Xie, J., Szymanski, B. K., & Liu, X. (2011) 'Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process', *In 2011 IEEE 11th International Conference on Data Mining Workshops. IEEE,* pp. 344-349.

[56] Xu, K. S., Kliger, M., & Hero, A. O. (2011, March) 'Tracking communities in dynamic social networks', In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction,* pp. 219-226.

[57] Yang, T., Chi, Y., Zhu, S., Gong, Y., & Jin, R. (2009, April) 'A bayesian approach toward finding communities and their evolutions in dynamic social networks', In *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, pp. 990-1001.

[58] Yin, G., Chi, K., Dong, Y., & Dong, H. (2017) 'An approach of community evolution based on gravitational relationship refactoring in dynamic networks', *Physics Letters A*, Vol. 381, No. 16, pp. 1349-1355.

[59] Zachary, W.W. (1977) 'An information flow model for conflict and fission in small groups', *Journal of Anthropological Research*, Vol. 33, pp.452-473.

[60] Chen, M., Nguyen, T. & Szymanski, B. K. (2015) 'A new metric for quality of network community structure'*, In arXiv preprint arXiv: 1507.04308*.

[61] Leskovec, J., Kleinberg, J. & Faloutsos, C. (ACM, 2005) 'Graphs over time: densification laws, shrinking diameters and possible explanations', *In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*), pp. 177-187.

[62] Souam, F., Aïtelhadj, A & Baba-Ali, R. (2014) 'Dual modularity optimization for detecting overlapping communities in bipartite networks', *Knowledge and information systems*, Vol.40, No. 2, pp. 455-488.

[63] Gehrke, J., Ginsparg, P & Kleinberg, J. M. (2003) 'Overview of the 2003 kdd cup', *SIGKDD Explorations*, Vol. 5, No. 2, pp. 149-151.