habilitation à diriger des recherches
de l'université Pierre et Marie Curie

Spécialité
Informatique

Présentée par
Jean-Loup Guillaume

Déterminisme et non-déterminisme au service de la détection de communautés dynamiques.

TABLE DES MATIÈRES

INTRODUCTION

De nombreux systèmes réels peuvent être modélisés par des graphes. On peut bien sûr penser aux réseaux sociaux, qu'ils soient en ligne ou pas, aux réseaux de communications téléphoniques, aux réseaux informatiques physiques ou virtuels tels que des réseaux pair-à-pair ou des réseaux de pages web, voir [5, 25, 39, 53, 77, 87] pour des exemples de travaux. La modélisation sous forme de graphes a également été appliquée dans beaucoup d'autres contextes, par exemple aux connexions entre aéroports [29], aux neurones et à leurs connexions synaptiques [1], aux synonymies entre mots [19], etc.

Le fait d'être modélisables par des graphes n'est pas le seul point commun entre ces exemples variés. Un certain nombre de travaux ont montré que ces réseaux, bien que différents par bien des aspects, sont aussi semblables par beaucoup d'autres : ils contiennent en effet des sommets très connectés aux autres et d'autres très isolés ; les voisins d'un sommet sont souvent connectés entre eux suivant le principe « les amis de mes amis sont mes amis » ; il existe des groupes de sommets fortement connectés entre eux, appelés *communautés* ; ils ont en commun bien d'autres propriétés [4]. Au-delà de ces similarités topologiques, plusieurs questions se posent de manière quasiment identique sur ces différents réseaux. Ainsi, que l'on considère la diffusion d'un virus informatique, d'une épidémie, d'un fichier sur un réseau pair-à-pair ou encore d'informations entre neurones, il s'agit toujours d'une propagation utilisant les connexions du réseau.

Avec la découverte de ces similarités, certains travaux ont commencé à appréhender ces réseaux comme un tout, tout en les étudiant également de manière individuelle. Ces réseaux ayant comme point commun d'être issus de contextes réels, ont reçu la dénomination commune de *graphes de terrain*.

Ce mémoire est dédié à une question spécifique aux graphes de terrain – la détection de communautés – que nous décrirons bientôt, mais qu'il est malgré tout important de resituer dans un contexte plus général. Toute étude sur les graphes de terrain doit commencer par la mesure d'un système réel afin de connaître les entités qui le composent et les connexions entre elles. Mesurer un système réel est rarement simple et les données obtenues sont presque toujours incomplètes et bruitées. La description et l'analyse des données modélisées sous forme de graphes peuvent ensuite se faire à l'aide de différents outils et à différents niveaux. On peut par exemple étudier le comportement individuel des sommets ou s'intéresser à des propriétés statistiques à un niveau plus macroscopique. On a en particulier souvent recours à la modélisation qui permet de synthétiser les données dans un modèle mathématique. Ce modèle peut aussi bien permettre de comprendre les données que de prédire leur évolution. Un exemple classique est la modélisation de la propagation d'épidémies, qui a pour objectif d'estimer ou de prédire le nombre de personnes infectées au cours du temps en fonction de la virulence de la maladie. Enfin, la taille des données requiert en général des outils algorithmiques performants et qui tirent parti, autant que possible, de la structure particulière des graphes de terrain. Ces différents aspects sont bien sûr fortement interconnectés.

L'étude des graphes de terrain a beaucoup progressé depuis une quinzaine d'années et l'état

de l'art fournit maintenant de nombreux outils pour les étudier. On dispose notamment d'algorithmes très rapides pour calculer automatiquement des communautés. Outre les questions fondamentales, de nombreux cas particuliers ont aussi été étudiés de manière plus spécifique dans différents contextes. Par exemple, dans [74, 89], des disciplines scientifiques sont inférées à partir de graphes de citations (*i.e.* qui cite qui dans un article scientifique). Dans [62], les auteurs montrent que, dans un réseau d'interaction entre protéines, des communautés calculées automatiquement regroupent des protéines de fonctions biologiques homogènes.

Le concept de « communauté » n'est pas simple à définir mais deux notions sont très fréquemment utilisées pour caractériser leurs membres : ils sont *similaires* et fortement *liés*. Pour ces raisons, les premières tentatives de modélisation du concept dans le cadre de la théorie des graphes ont interprêté ces notions par une faible distance (grande similarité ou proximité) ou une forte densité (forte connectivité). Les communautés ont ainsi parfois été définies comme des sous-graphes étant : des composantes connexes (il existe un chemin entre toute paire de sommets), des cliques (tous les sommets sont connectés) [64], des n-cliques (tous les sommets sont connectés à distance au plus n) [2], des *k-degree sets* ou *k-degree cores* (tous les sommets ont au moins k liens vers les autres) [79], et un certain nombre d'autres notions [36, 66].

En dehors de ces critères de forte densité ou de courte distance, un critère supplémentaire est apparu pour signifier qu'une communauté n'a en général de sens que vis-à-vis de l'extérieur et que ses membres devraient donc être plus similaires entre eux qu'avec les autres, ou plus reliés les uns avec les autres qu'avec l'extérieur. Ceci a conduit à des définitions du type : *k-outdegree set* (au plus k liens vers l'extérieur) [22], LS Set (aussi peu de liens vers l'extérieur que possible) [63], Alpha set (chaque membre a plus de liens internes que de liens externes) [22], etc.

La définition de la notion de communauté va souvent de pair avec la proposition de méthodes pour les calculer en pratique et, pour cette raison, un grand nombre d'algorithmes ont été développés (voir [41] pour une présentation très complète des méthodes proposées ces dernières années). Dans ce mémoire, nous nous focaliserons sur les méthodes dites de *partitionnement* où chaque sommet du graphe est associé à une et une seule communauté. Ce choix de partionner un graphe en communautés est très fort et il exclut notamment que des sommets appartiennent à plusieurs communautés. L'étude des *communautés recouvrantes* représente cependant une perspective très importante dans le domaine de la détection de communautés. Nous montrerons dans le chapitre 3 une méthode qui a été utilisée dans ce contexte et nous discuterons rapidement d'une autre méthode en conclusion.

Au cœur des méthodes de partitionnement on trouve souvent une formalisation de ce qu'est une « bonne partition » d'un graphe en communautés. Dans ce mémoire, nous nous concentrerons sur la *modularité*, une mesure de qualité des partitions introduite récemment [68] et qui, malgré ses défauts, s'est très largement imposée dans le domaine.

Enfin, la plupart des graphes de terrain sont dynamiques et évoluent au cours du temps par l'ajout ou la suppression de sommets et de liens. Cette dynamique n'a pas été prise en compte dans les premières études des graphes de terrain à cause de la difficulté à attaquer d'emblée l'analyse de la structure des graphes de terrain et l'étude de leur dynamique. Il y a quelques années, cette question de la dynamique a ressurgi, le domaine étant clairement défini et l'état de l'art suffisamment avancé.

Tout comme la structure peut être différente d'un graphe de terrain à l'autre, il n'y a pas non plus une dynamique unique. Les travaux actuels dans ce domaine sont encore à leurs débuts et cherchent à caractériser la dynamique des graphes réels. Il faut pour cela mesurer la dynamique de ces graphes, ce qui est très complexe car l'opération de mesure prend du temps, pendant lequel le graphe continue d'évoluer. Il faut ensuite décrire cette dynamique et, là encore, aucune

méthode claire n'existe : les quelques modèles existants, voir [4] par exemple, s'attachent plus au résultat final qu'à la manière dont ce résultat est atteint. Pour reprendre l'exemple de la propagation d'épidémies, s'il est intéressant de savoir quelle proportion de la population sera infectée *in fine*, il est aussi intéressant de savoir quels individus le seront, à quel moment, et comment la propagation a eu lieu, notamment si certains contacts la favorisent.

L'approche classique consiste à considérer un graphe dynamique comme une succession de graphes statiques entre lesquels des modifications se produisent. On peut alors décrire la dynamique en calculant plusieurs propriétés à chaque instant et en étudiant leur évolution. Cette méthode, valable dans de nombreux contextes, reste limitée car certaines propriétés, par exemple la durée de vie des liens, ne se décrivent pas naturellement dans ce cadre.

En revanche, cette vision est toujours au centre des études sur les communautés dynamiques. Ces dernières sont dans ce cas calculées à chaque instant, de manière plus ou moins indépendante. Nous montrerons que cette solution n'est pas utilisable directement du fait qu'il y a énormément de partitions de très bonne qualité mais structurellement assez différentes. Cela conduit, si l'on n'y prend pas garde, à observer des évolutions de la structure communautaire sans lien avec l'évolution de la topologie.

Plusieurs études très récentes commencent à utiliser cette approche et justifient de l'intérêt des études amont. Par exemple, dans [48], les auteurs étudient l'évolution du domaine des systèmes complexes à partir de la dynamique d'écriture et de citation d'articles. Dans [40], les auteurs calculent les corrélations entre les taux de change du marché des changes et mettent en évidence des modifications du comportement transactionnel via des réorganisations de la structure communautaire. Enfin dans [14], les auteurs utilisent des mesures dynamiques de la connectivité fonctionnelle du cerveau pendant l'apprentissage d'une tâche et parviennent à identifier une structure communautaire qui leur permet de prédire l'apprentissage ultérieur.

Ce mémoire est consacré à l'étude de la dynamique des communautés sous deux angles assez différents. Après une présentation rapide du contexte nécessaire à la bonne compréhension de ce mémoire, ainsi que des méthodes de calcul et d'évaluation de partitions en communautés dans le chapitre 1, je présenterai deux approches pour le calcul de communautés dynamiques. La première, intitulée *communautés multi-pas* et introduite dans le chapitre 2, consiste à rendre les algorithmes plus déterministes pour pouvoir suivre les communautés au fil du temps. La seconde, nommée *cœurs de communautés* et présentée dans le chapitre 3, exploite au contraire le fait que les algorithmes soient naturellement non-déterministes.

Je ne présenterai pas de manière complète les résultats auxquels j'ai contribué sur ces deux thèmes mais je tenterai d'en expliciter les points majeurs et de les mettre en perspective. Les détails techniques se trouvent bien entendu dans les publications auxquelles j'ai participé et qui seront référencées dans la suite. De même, je m'abstiendrai de présenter les états de l'art sur les différents sujets de manière extensive. Des articles très complets le font déjà de manière fort satisfaisante et je préfère les référencer que les paraphraser.

D'autre part, certains de mes travaux pendant ces dernières années ont été consacrés à d'autres sujets que la détection de communautés, mais toujours dans le domaine de l'étude des graphes de terrain. J'ai préféré ne pas parler de ces études par souci de cohérence mais certains aspects font partie de mes perspectives de recherche et j'en parlerai donc en conclusion.

Enfin, tous les travaux présentés ici ont été réalisés en collaboration avec d'autres chercheurs ou étudiants. J'utiliserai donc le pronom « nous » afin de ne pas laisser croire que ces travaux sont de mon seul fait, et me cantonnerai au « je » pour les remarques plus personnelles.

CHAPITRE 1

CONTEXTE

Avant de proposer des méthodes de détection de communautés dynamiques, il convient d'introduire un certain nombre de notations et de méthodes que nous utiliserons par la suite dans ce mémoire. Tout d'abord, nous allons discuter de l'évaluation de la qualité d'une partition en communautés et des principaux problèmes posés par la fonction la plus couramment utilisée : la modularité.

Ensuite, nous montrerons brièvement que l'approche naïve qui consiste à calculer des communautés à chaque instant puis à tenter de suivre leur évolution n'est pas utilisable directement à cause d'un phénomène d'instabilité causé principalement par le nombre élevé de partitions pertinentes.

Puis, nous présenterons la méthode de Louvain qui, depuis son introduction en 2008, fait référence dans le domaine de l'optimisation de la modularité tant pour la qualité de ses résultats que pour sa vitesse d'exécution. Il est nécessaire de la présenter en détail, car les approches que nous introduirons dans les chapitres ultérieurs sont fondées sur cette méthode.

Enfin, nous ferons un point très rapide sur les méthodes générales utilisées pour calculer des communautés dynamiques. Cette section sera volontairement très courte et nous renvoyons à [13] pour une description plus détaillée des différentes méthodes.

Notations

Étant donné un graphe $G = (V, E)$ non orienté avec $n = |V|$ sommets et $m = |E|$ liens, ce graphe peut être décrit par sa matrice d'adjacence (symétrique) A dont les éléments A_{ij} sont les poids des liens entre les sommets i et j. Dans le cas d'un graphe non pondéré, les éléments A_{ij} valent simplement 1 lorsqu'il existe un lien entre i et j, 0 dans le cas contraire. Notons qu'une valeur non nulle A_{ii} sur la diagonale correspond à une boucle (éventuellement pondérée) sur le sommet i. Les méthodes que nous présenterons dans la suite s'appliqueront de manière indifférenciée aux graphes pondérés ou non, sauf mention explicite du contraire. Par contre la transition des graphes non orientés aux graphes orientés est en général plus complexe.

Une communauté dans un graphe représente de manière équivalente un sous-ensemble C de V ou le sous-graphe de G induit par C. Une partition du graphe G en communautés attribue à chaque sommet i une unique communauté C_i et deux sommets i et j appartiennent à la même communauté si et seulement si $C_i = C_j$. Dans la suite, une telle partition d'un graphe en communautés sera notée $\mathcal{P}$.

1.1 Fonctions de qualité

Avant de parler de fonctions de qualité, il convient de rappeler un théorème d'impossibilité énoncé par Jon Kleinberg [56] :

Théorème 1 *For each $n \geq 2$, there is no clustering function f that satisfies Scale-Invariance, Richness, and Consistency.*

Cela signifie que dès que l'on considère un graphe pondéré ayant plus de 2 sommets, il n'est pas possible de trouver une fonction qui partitionne ce graphe et qui respecte simultanément :

- l'invariance d'échelle : si l'on multiplie tous les poids des liens du graphe par une constante, alors le partitionnement ne doit pas changer ;
- la richesse : en modifiant les poids des liens, il est possible d'obtenir toutes les partitions possibles ;
- la consistence : si les poids des liens intra-communautaires sont augmentés et que les poids des liens inter-communautaires sont diminués, alors le partitionnement ne doit pas changer.

Le théorème ne tient plus avec une relaxation simple de la consistence qui autorise que les augmentations et les diminutions donnent un sous-partionnement du graphe original, ce qui ne semble pas complètement choquant. Nous verrons cependant par la suite que les fonctions de qualité utilisées en pratiques ont des défauts bien plus gênants ; leur popularité tient à ce qu'elles donnent des résultats très pertinents dans des contextes réels.

De nombreuses méthodes de partitionnement de graphe reposent sur une fonction de qualité qui attribue un score à toute partition des sommets. Ainsi, la fonction Q est telle que $Q(\mathcal{P}_1) > Q(\mathcal{P}_2)$ si $\mathcal{P}_1$ est un meilleur (dans un sens à définir) partitionnement que $\mathcal{P}_2$ pour le graphe considéré.

Dans la suite, nous allons présenter en détail la modularité qui est la fonction de qualité la plus populaire du domaine malgré ses défauts. Nous présenterons également quelques autres fonctions moins utilisées.

1.1.1 La modularité

La modularité, introduite par Girvan et Newman [68] est basée sur l'idée intuitive qu'une communauté est un groupe de sommets fortement liés entre eux mais peu avec l'extérieur. De manière formelle, elle est définie pour une partition $\mathcal{P}$ d'un graphe ayant pour matrice d'adjacence A par :

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(C_i, C_j) = \frac{1}{2m} \sum_{C \in \mathcal{P}} \sum_{i,j \in C} \left[A_{ij} - \frac{k_i k_j}{2m} \right] ,$$

où la somme est faite sur toutes les paires de sommets appartenant à une même communauté et où k_i est le degré du sommet i.

Par construction, la modularité compare, pour chaque communauté, le nombre de liens internes avec le nombre attendu de liens dans un modèle de référence. Ce dernier est censé représenter de manière non biaisée les propriétés connues du graphe et le choix classique est celui d'un graphe aléatoire ayant les mêmes degrés que le graphe original sans aucune autre contrainte [16]. Ainsi, la probabilité d'existence d'un lien entre i et j vaut $\frac{k_i k_j}{2m}$.

En comparant les propriétés du graphe et celles d'un graphe aléatoire, la modularité ne cherche donc pas des groupes denses, mais des groupes de densité anormalement élevée. La modularité d'une partition est un nombre réel compris entre $-1/2$ et 1 et, en pratique, la modularité ne peut être égale à 1. Une borne supérieure plus précise, valant $1 - 1/\sqrt{2m}$, est donnée dans [42].

La modularité a initialement été introduite pour sélectionner le niveau le plus adapté dans une hiérarchie de partitions. Par la suite, elle est devenue un élément essentiel d'un grand

nombre de méthodes de détection de communautés qui tentent de trouver la partition ayant la modularité maximale. Il a été montré que l'optimisation de la modularité est un problème NP-complet en général [24] et, pour cette raison, de nombreuses heuristiques ont été proposées pour trouver des partitions de bonne modularité. Nous renvoyons à [32, 41] pour un aperçu assez large de la littérature sur le sujet.

Quelques propriétés simples de la modularité sont particulièrement aisées à dériver mais sont néanmoins instructives :

1. si tous les sommets sont regroupés dans la même communauté, on obtient une modularité nulle, donc même si la modularité peut théoriquement être négative, celle de la partition optimale d'un graphe est toujours positive ;
2. si une partition contient une communauté non connexe, alors on obtient un gain de modularité en séparant cette communauté en sous-communautés connexes, sans changer le reste de la partition ;
3. si le graphe n'est pas pondéré, alors un sommet de degré 1 est toujours placé dans la communauté de son unique voisin dans une partition optimale.

Limitations de la modularité

La modularité souffre intrinsèquement de plusieurs défauts. Une première limitation vient du fait qu'un graphe peut avoir une structure communautaire hiérarchique et être donc composé de partitions significatives à différentes échelles. Par construction, l'optimisation de la modularité arrivera au mieux à trouver une de ces partitions, négligeant de ce fait les autres niveaux de description significatifs du système. Des solutions ont été proposées qui sont basées sur l'ajout d'un paramètre de résolution $\alpha \in [0; +\infty[$ dans la définition de la modularité [75]. Ce paramètre permet de donner plus ou moins d'importance au terme aléatoire ($\frac{k_i k_j}{2m}$) et ne pénalise que très peu les gros regroupements s'il est proche de 0, alors qu'il privilégie les petites communautés s'il est élevé. Il faut ensuite trouver les bonnes valeurs de α qui permettent d'obtenir des niveaux pertinents. Cette modularité multi-échelles est définie comme :

$$Q_\alpha(\mathcal{P}) = \frac{1}{2m} \sum_{C \in \mathcal{P}} \sum_{i,j \in C} \left[A_{ij} - \alpha \frac{k_i k_j}{2m} \right] .$$

Une seconde limitation vient du fait que la modularité d'une partition est explicitement liée au nombre de liens du graphe. Considérons une partition $\mathcal{P}$ dans lequel deux groupes de sommets C_1 et C_2 sont séparés ($\mathcal{P} = \{\mathcal{P}_1, C_1, C_2\}$) et la même partition dans lequel ils sont regroupés ($\mathcal{P}' = \{\mathcal{P}_1, C_1 \cup C_2\}$), pour une partition $\mathcal{P}_1$ quelconque des sommets hors de C_1 et de C_2. La modularité se calculant comme une somme sur toutes les communautés, le choix de regrouper ou non C_1 et C_2 n'affecte pas les autres communautés ; on peut donc calculer le gain de modularité obtenu en regroupant C_1 et C_2 comme :

$$\begin{aligned} Q(\mathcal{P}') - Q(\mathcal{P}) &= \frac{1}{2m} \sum_{i,j \in C_1 \cup C_2} \left[A_{ij} - \frac{k_i k_j}{2m} \right] - \frac{1}{2m} \sum_{i,j \in C_1, i,j \in C_2} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \\ &= \frac{1}{2m} \sum_{i \in C_1, j \in C_2} \left[2A_{ij} - \frac{2k_i k_j}{2m} \right] . \end{aligned}$$

En conséquence, si le nombre de liens entre C_1 et C_2 est supérieur au nombre de liens attendu entre eux, la modularité incite à fusionner ces deux groupes. Or, le nombre de liens attendus dépend du nombre de liens total dans le graphe, m, et le choix de regrouper ou pas sera donc

indépendant de la structure de C_1 et C_2. Ce problème est connu sous le nom de résolution limite et a été présenté dans [42].

Un autre problème lié au précédent concerne la modularité de graphes n'ayant aucune structure communautaire, par exemple des graphes aléatoires [50, 76] ou des graphes réguliers comme par exemple des grilles [33]. Comme la modularité compare la structure détectée à celle attendue dans un graphe aléatoire, on s'attendrait naturellement à ce qu'un graphe aléatoire ait une modularité nulle. Or, il a été montré que la partition optimale d'un graphe aléatoire a une modularité positive qui augmente avec la taille du graphe. Ceci a deux causes principales qui sont liées :

1. une instance d'un graphe aléatoire est un « vrai » graphe et il y a donc des fluctuations qui vont créer des zones plus denses que prévu donnant lieu à des communautés de modularité non nulle ;
2. on compare une instance particulière avec un modèle de référence : dans la majorité des cas (graphes aléatoires de grande taille avec probabilité de connexion très faible), les probabilités d'existence des liens sont largement inférieures à 1 or l'instance a quant à elle des liens absents ou présents, donc des valeurs A_{ij} valant 0 ou 1. Dans ce dernier cas, tous les liens présents sont « inattendus » et incitent à des regroupements.

1.1.2 Quelques autres fonctions de qualité

Comme nous allons utiliser exclusivement la modularité dans la suite, nous ne présentons que très brièvement d'autres fonctions de qualité. La plupart de ces fonctions utilisent quelques ingrédients de base, comme notamment les accords positifs (sommets reliés et dans la même communauté), négatifs (sommets non reliés et dans des communautés différentes) ou des désaccords (les deux autres cas). La modularité quant à elle ne regarde que les accords positifs et les probabilités d'existence des liens.

Le critère de Zahn-Concorcet [96] consiste à ne regarder que les désaccords et à chercher la partition qui en minimise le nombre. Le critère de Mancoridis [65] se concentre au contraire sur les accords, qu'ils soient positifs ou négatifs, et cherche à en maximiser le nombre. Cette fonction semble intuitive car non seulement elle prend en compte les liens intra-communautaires, mais aussi l'absence de liens inter-communautaires. Pour la modularité, seul le premier aspect est présent, avec cependant une comparaison à l'aléatoire qui apporte une autre information.

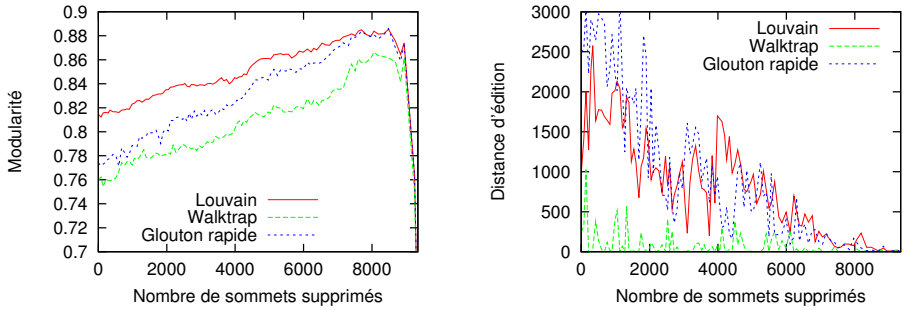
De nombreuses autres fonctions ont été proposées que ce soit pour la détection de communautés ou de manière plus générale pour comparer plusieurs partitions d'un ensemble. Cependant, nous aller montrer maintenant un problème majeur qui se pose lorsque l'on tente de chercher des communautés. Nous allons illustrer ce problème sur la modularité, mais il est très probable que toutes les fonctions de qualité en soient affectées, même si aucune étude en profondeur n'a été réalisée sur le sujet.

1.1.3 Le problème de l'instabilité

Nous proposons un exemple de dynamique simulée simple afin de mettre en évidence le problème de l'instabilité en utilisant la modularité comme fonction de qualité. Cette dynamique consiste à considérer un graphe puis à en supprimer les sommets un par un en recalculant les communautés après chaque suppression. Chaque instant temporel correspond donc à une suppression. Nous avons pour cela utilisé trois algorithmes très différents afin de nous assurer que nos observations ne soient pas liées au fonctionnement d'un algorithme en particulier.

La figure 1.1(a) présente la modularité de la partition obtenue après chaque suppression. On peut y voir qu'elle croît lentement, avec peu de variations, jusqu'à atteindre un point de

rupture qui correspond à la perte de connexité presque complète du graphe. En revanche, la *distance d'édition*[1] entre les partitions successives présentée sur la figure 1.1(b) donne une vision complètement différente. Selon les algorithmes, on a de quelques centaines à quelques milliers de sommets qui changent de communauté à chaque suppression de sommet, sachant qu'il y a 9377 sommets dans le graphe initial. Cela semble déraisonnable pour la suppression d'un seul sommet du graphe à chaque fois.



(a) (b)

FIGURE 1.1 : (a) Modularité de chaque partition et (b) distance d'édition entre deux partitions successives durant une dynamique simulée qui consiste à supprimer les sommets un par un dans un ordre aléatoire sur un graphe de coécriture d'articles [30] pour trois algorithmes de détection de communautés : la méthode de Louvain [18], Glouton Rapide [28] et WalkTrap[73].

Cette instabilité est encore plus flagrante dans le cas des algorithmes non-déterministes car différentes exécutions sur le même graphe, sans aucune modification topologique, donnent des résultats très différents, du même ordre de grandeur que ce que nous venons de voir.

Ce problème est fondamental et est la cause de la majorité des problèmes observés lorsque l'on tente de calculer des communautés dynamiques. En effet, si la suppression d'un seul sommet change radicalement la structure communautaire (ou du moins la vision que l'on en a avec la modularité), comment espérer comprendre l'évolution des communautés dans ce contexte ?

1.2 La méthode de Louvain

Nous allons maintenant présenter la méthode de Louvain [18], une méthode gloutonne pour optimiser la modularité. Les principaux avantages de cette méthode sont sa rapidité, qui lui permet de traiter des graphes ayant plusieurs milliards de sommets ou de liens, son aspect multi-échelle, c'est-à-dire sa faculté de découvrir des communautés à différentes échelles, et la très bonne qualité des partitions fournies. Elle est composée de deux phases répétées de manière itérative jusqu'à obtenir un maximum local de la modularité. L'algorithme part d'un graphe pondéré non orienté ayant n sommets.

Première phase : la partition initiale consiste à placer chaque sommet dans une communauté distincte. Nous considérons ensuite un sommet i et calculons le gain de modularité obtenu en supprimant i de sa communauté et en le plaçant dans la communauté de l'un de ses voisins[2] j. Le sommet i est ensuite effectivement déplacé dans la communauté du voisin j pour lequel ce gain est maximum (uniquement si le gain est positif, sinon i est replacé dans sa communauté

1. La distance d'édition est le nombre minimal de sommets à déplacer pour que les deux partitions soient égales. Elle se calcule rapidement avec la méthode présentée dans [57].

2. On n'explore donc que l'ensemble des partitions dont les communautés sont connexes ce qui est à rapprocher du fait que la partition de modularité maximale est aussi dans cet ensemble.

d'origine). Ce processus est appliqué de manière séquentielle sur tous les sommets, puis est itéré jusqu'à ce que plus aucun déplacement n'améliore la modularité. Nous insistons sur le fait qu'un même sommet peut être déplacé plusieurs fois suite à l'évolution des communautés auxquelles appartiennent ses voisins. Après cette première phase, on obtient une partition $\mathcal{P}$ ayant un certain nombre de communautés. Si $\mathcal{P}$ n'est pas la partition initiale alors l'algorithme passe à la deuxième phase, sinon l'algorithme est terminé et le résultat est $\mathcal{P}$.

Deuxième phase : le graphe est agrégé en un nouveau graphe dont les sommets sont les communautés découvertes durant la première phase. Pour cela, le poids des liens entre ces nouveaux sommets est donné par la somme des poids des liens qui existaient entre les sommets des deux communautés. Les liens entre des sommets d'une même communauté créent des boucles sur cette communauté dans le nouveau graphe. Cette phase d'agrégation ne change pas la modularité, c'est-à-dire que la partition du graphe initial a la même modularité que la partition triviale du graphe agrégé correspondant. Il est possible d'appliquer à nouveau la première phase sur ce graphe et d'itérer.

Dans la suite, nous nommerons *passe* une combinaison de ces deux phases. Cette répétition de passes construit naturellement une hiérarchie de communautés. La sortie de l'algorithme est donc une suite de partitions qui contiennent des communautés de plus en plus grandes. En général, seule la partition qui maximise la modularité trouvée à la dernière passe est utilisée, mais les niveaux intermédiaires sont aussi intéressants à étudier ; par exemple les auteurs de [37] montrent que le niveau le plus bas distingue mieux les spams dans des réseaux d'échange d'emails que les niveaux plus hauts qui sont pourtant de meilleure modularité.

1.2.1 Efficacité

Une partie de l'efficacité de l'algorithme réside dans le fait que la variation de modularité obtenue en supprimant un sommet i de sa communauté ou en l'insérant dans celle de l'un de ses voisins j peut se calculer avec uniquement la connaissance du nombre de liens du sommet vers la communauté considérée, du nombre de liens internes et de la somme des degrés des sommets de la communauté. Ces différents paramètres peuvent être stockés et mis à jour à chaque déplacement. Le déplacement d'un sommet a donc un coût linéaire en son degré et considérer une fois tous les sommets a un coût linéaire en le nombre de liens du graphe.

D'un point de vue théorique, la complexité dans le pire des cas est bien plus mauvaise que celle de la plupart des autres algorithmes. On peut la borner en montrant que le gain minimal obtenu lors du déplacement d'un sommet est supérieur à $1/2m^2$. On en déduit une convergence en $\mathcal{O}(m^2)$ déplacements, soit $\mathcal{O}(m^3)$ tests, car il faut au pire tester tous les sommets pour finalement trouver celui à déplacer.

La complexité de l'algorithme est en pratique bien plus faible : sur des graphes ayant une structure communautaire, chaque sommet est traité quelques dizaines de fois avant convergence [3]. Nous n'avons pas trouvé de corrélation entre cette valeur et la taille des graphes traités, nous considérons donc abusivement que le nombre de traitements par sommet est constant et que l'algorithme est donc linéaire en m. En d'autres termes, à moins de concevoir un algorithme sous-linéraire qui arrive à trouver des partitions de qualité en ne regardant pas tous les liens, on peut considérer qu'il est quasi optimal en temps.

Nous avons validé la méthode de Louvain sur divers modèles de graphes aléatoires ayant une structure communautaire [6, 61, 68] ainsi que sur un grand nombre de graphes de terrain de taille variable. Les résultats présentés dans le tableau 1.1 montrent clairement que la méthode de Louvain est nettement plus rapide que les autres algorithmes et qu'elle permet d'étudier

3. On s'arrête en général avant, dès qu'il n'y a presque plus de déplacements et que les gains deviennent trop faibles.

des graphes de taille sans précédent. Une version de l'algorithme nous a par exemple permis d'analyser un graphe ayant plus de 15 milliards de liens. Cette rapidité ne se fait pas aux dépends de la qualité, les valeurs de modularité étant également meilleures que celles de nombreuses autres méthodes. Il faut reconnaître l'existence d'approches fournissant des résultats de meilleure qualité que Louvain mais très souvent au prix d'une complexité qui limite leur utilisation à des graphes de taille très réduite.

Le tableau 1.1 (dernière ligne) présente également la modularité et les temps de calcul de la méthode de Louvain sur des graphes aléatoires. On peut y voir que ces derniers, malgré leur absence de communautés, affichent des modularités non nulles. Il est également intéressant de remarquer que le temps de calcul pour ces graphes est beaucoup plus long que pour les graphes réels. Cette différence est due à l'absence de structure communautaire, ce qui oblige l'algorithme à régulièrement remettre en cause les choix déjà effectués.

	Karaté	Arxiv	Internet	Web nd.edu	Téléphonie	WebUk05	WebBase01
sommets/liens	34/77	9k/24k	70k/351k	325k/1M	2M/5.4M	39M/783M	118M/1B
Glouton Rapide [28]	.38/0s	.772/3.6s	.692/799s	.927/84min	-/-	-/-	-/-
WalkTrap [73]	.42/0s	.757/3.3s	.729/575s	.895/111min	-/-	-/-	-/-
WT [88]	.42/0s	.761/0.7s	.667/62s	.898/248s	.553/367s	-/-	-/-
Louvain	.42/0s	.813/0s	.781/<1s	.935/<1s	.76/47s	.979/252s	.984/469s
Graphes aléatoires	.288/0s	.256/0s	.153/<1s	.2/2.6s	.41/41s	.108/45min	.226/4h

TABLE 1.1 : Résultats obtenus sur plusieurs graphes de taille croissante. Chaque ligne donne la modularité et le temps de calcul pour différentes approches. La méthode de Louvain nécessite environ 10Go de mémoire pour traiter le graphe WebBase01, principalement utilisés pour le stockage du graphe lui-même. Les graphes utilisés sont Karaté [95], Arxiv [30], Internet [51], Web nd.edu [5], Téléphonie [60], WebUk05 et WebBase01 [20, 21]. La dernière ligne correspond à l'application de la méthode de Louvain sur des graphes aléatoires de même taille (résultats moyennés sur 20 réalisations).

La méthode de Louvain peut être optimisée de plusieurs manières afin de l'accélerer encore et d'augmenter un peu la modularité. De même, nous avons étudié l'influence de l'ordre de traitement des sommets, de la résistance de la méthode au problème de limite de résolution, etc. Nous renvoyons à [9, 18] pour obtenir des détails qui ne sont pas utiles ici.

1.3 Communautés dynamiques

Avant de développer dans les chapitres suivants les approches que nous avons proposées pour calculer des communautés dynamiques, nous allons tout d'abord faire un point rapide sur les méthodes existantes. Nous renvoyons à [13] pour une description plus détaillée.

La plupart des approches dans ce domaine considèrent un graphe dynamique comme une séquence ordonnée de graphes statiques. Ayant à disposition de nombreux algorithmes de détection de communautés pour les graphes statiques, la difficulté semble alors reportée sur le problème d'appariement : comment établir un lien entre les communautés calculées à l'instant t et celles calculées à l'instant $t+1$?

Ce problème d'appariement est complexe car il faut être capable d'identifier les fusions de communautés, séparations, apparitions, disparitions, évolutions simples, etc. Les approches de suivi de communautés sont généralement basées sur des règles de décision liées à des méthodes ensemblistes : si deux communautés C_t et C'_{t+1} partagent beaucoup de sommets, alors elles sont liées d'une manière ou d'une autre ; au contraire, si elles en partagent très peu, elles ne le sont pas.

À cela se rajoute le problème de stabilité que nous avons évoqué plus tôt, à cause duquel les communautés peuvent changer de manière drastique et cela, sans lien apparent avec de réelles modifications de la structure.

La première méthode de suivi est proposée par [52], avec la définition d'un accord entre deux communautés comme étant la proportion de sommets dans l'intersection. C_t évolue en C'_{t+1} si C'_{t+1} est la communauté qui a le plus fort accord avec C_t. Des règles plus complexes permettent de gérer les cas de division, regroupement, apparition et disparition de communautés [7, 38, 49, 69, 70, 84].

Dans ce contexte, deux méthodes principales sont utilisées pour régler les problèmes d'instabilité :

1. ne considérer que les communautés stables, généralement définies comme les communautés qui continuent à exister même après une modification mineure [52] ;
2. suivre les sommets « centraux » des communautés, qui sont *a priori* plus stables et qui, s'ils se séparent, indiquent une séparation de la communautés [15, 27, 90, 92].

Une autre approche consiste à calculer les partitions sur l'union des graphes G_t et G_{t+1} puis à extraire les communautés aux instants t et $t+1$. L'extraction n'est pas forcément évidente mais les auteurs de [71] proposent une définition de communautés à base de cliques qui assure que les communautés de G_t et de G_{t+1} sont incluses dans celles de l'union. Cette définition est malheureusement trop restrictive et nécessite de définir des paramètres pour gérer les cas de regroupements et de scissions.

Les premières approches intégrant clairement le phénomène d'instabilité ont modifié la fonction de qualité afin d'intégrer un paramètre de stabilité pour obtenir une fonction du type $Q = Q_{t+1} + \alpha Q_{stabilite(t,t+1)}$, voir par exemple [26, 58]. Une approche similaire est basée sur des modifications des matrices d'adjacence [94] ou plus généralement sur les distances entre objets à classer [55], avec un paramètre α qui permet de régler l'importance entre la qualité intrinsèque et la stabilité. De manière assez similaire, il est possible d'exiger que la partition à l'instant $t+1$ soit aussi pertinente à l'instant t [83].

Les méthodes se limitant à certaines parties du graphe sont donc peu générales et nécessitent d'identifier de telles parties pour pouvoir les suivre. Les dernières approches à base de stabilisation sont au contraire plus pertinentes car beaucoup plus générales et sont proches de celles que nous allons présenter dans la suite de ce mémoire. Elles ont d'ailleurs été introduites de manière indépendante à la même période.

1.4 Synthèse

Nous avons présenté rapidement dans ce chapitre la modularité et la méthode de Louvain qui sont au centre des travaux de ce mémoire. Nous avons vu que la modularité n'est pas la seule fonction de qualité et qu'elle n'est pas exempte de limitations, notamment du fait de son instabilité. La méthode de Louvain, quant à elle, a un certain nombre d'avantages, notamment sa vitesse d'exécution et la qualité des partitions qu'elle fournit, ce qui permet de calculer sans problèmes de nombreuses partitions sur des graphes de grande taille. Cela est absolument nécessaire si l'on veut aborder le problème des graphes dynamiques, où le produit de la taille par le nombre de pas de temps implique que l'on doit traiter des données massives.

Le problème de stabilité que nous avons présenté est central dans le calcul de communautés dynamiques, car il empêche d'utiliser l'approche naïve consistant à calculer indépendamment les communautés à chaque instant en espérant recoller les morceaux par la suite. Je suis convaincu

que ce problème n'est dû ni aux algorithmes utilisés, ni à la modularité elle-même mais qu'il est intrinsèquement lié au fait de vouloir partitionner l'ensemble des sommets. De même, si l'on était capable de trouver la partition optimale, cela ne résoudrait en rien le problème dans le cas dynamique car une perturbation infime pourrait perturber complètement l'optimal. Pour tenter de passer outre ce problème, nous avons envisagé deux solutions.

La première solution que nous présenterons dans le chapitre 2 consiste à réduire le problème du non-déterminisme et à tenter de stabiliser le calcul de communautés à un instant $t+1$ en se servant de ce que nous avons calculé à l'instant t. L'idée sous-jacente est que si nous avons pu trouver une bonne partition à l'instant t, alors il est probable que de très légères modifications de cette partition soient suffisantes pour obtenir une bonne partition à l'instant $t+1$, sous réserve que le graphe ait peu évolué. Nous généralisons ensuite cette approche en considérant que la partition est sans doute toujours valable telle quelle à $t+1$ et peut-être même aux instants ultérieurs.

La seconde solution, que nous présenterons dans le chapitre 3, est complètement orthogonale et utilise fortement le non-déterminisme des méthodes de détection de communautés et l'instabilité. Une fois le constat fait qu'il existe de nombreuses partitions de bonne qualité, on peut se demander pourquoi en privilégier une plutôt qu'une autre (quand bien même ce serait l'optimale du point de vue de la modularité) et nous considérerons que les bonnes partitions apportent toutes une information pertinente. Nous tenterons donc de voir ce que l'on peut tirer d'un ensemble de partitions tant pour les communautés statiques que pour les communautés dynamiques.

CHAPITRE 2

STABILISATION

Comme nous l'avons vu précédemment, le problème de stabilité ne permet pas de calculer des communautés de manière indépendante à différents instants et d'espérer pouvoir recoller les morceaux par la suite.

Les deux approches que nous proposons dans ce chapitre ont été proposées et étudiées durant la thèse de Thomas Aynaud et ont pour objectif final de pouvoir suivre les communautés dans le temps tout en gardant la modularité comme fonction de base pour estimer la qualité d'une partition. La première approche consiste à briser l'indépendance entre les différents calculs en utilisant le résultat obtenu à un instant donné pour calculer le résultat à l'instant suivant [11]. La seconde approche consiste à chercher des communautés qui soient de bonne qualité sur plusieurs instants consécutifs. Nous les appelerons des *communautés multi-pas* [10]. Cela nécessite que le graphe n'ait pas trop évolué sur la période considérée, faute de quoi le résultat n'a que peu de chance de rester valable. Nous proposerons finalement une solution pour trouver de telles périodes stables à l'aide d'une segmentation temporelle hiérarchique [12].

2.1 Mémoires du passé

La méthode la plus simple pour rompre l'indépendance des calculs consiste à garder la mémoire des calculs effectués à tous les instants, ou au moins à l'instant t, lorsque l'on calcule les communautés à l'instant $t+1$. Pour cela il faut par exemple être capable d'injecter à l'instant $t+1$ les résultats obtenus à l'instant t. Avec la méthode de Louvain, cela est simple à faire car l'algorithme débute toujours avec une partition initiale qu'il va faire évoluer vers un optimum de modularité. Cette partition initiale est généralement une partition triviale où chaque sommet forme une communauté.

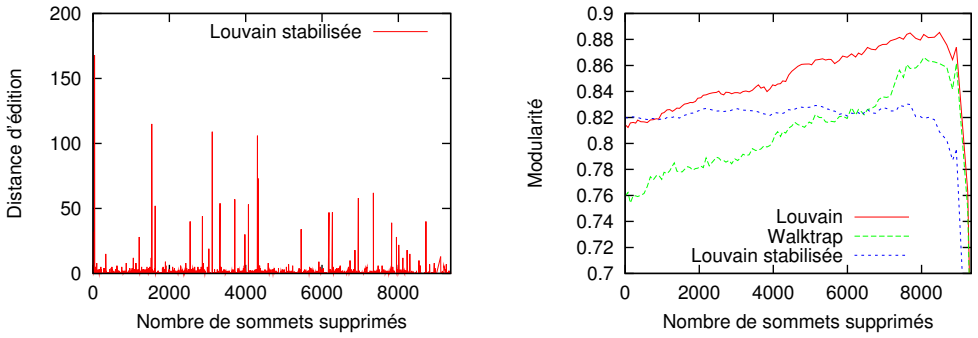
Dans le cas présent, nous pouvons justement utiliser la partition obtenue à l'instant t comme initialisation à l'instant $t+1$. L'algorithme débute alors avec des communautés qui ont déjà du sens, ou du moins qui en avaient à l'instant précédent et, si le graphe n'a pas trop évolué, on peut naturellement considérer qu'elles ont toujours du sens. Nous appelons cette méthode la méthode de Louvain stabilisée. Bien entendu, cette hypothèse est à considérer avec précaution et la méthode de Louvain va tenter d'ajuster cette partition en déplaçant des sommets. Notons que cette approche ne s'applique pas à toutes les méthodes de calcul de communautés : il faut pouvoir choisir la partition initiale tout en pouvant la remettre en cause par la suite.

2.1.1 Évaluation

Nous allons maintenant évaluer la méthode de Louvain stabilisée sur plusieurs graphes dynamiques, en la comparant à la méthode de Louvain classique ainsi qu'à WalkTrap. Il faut noter

que, comme la méthode de Louvain calcule les partitions à chaque instant de manière indépendante, on peut espérer qu'elle trouve la meilleure partition à chaque instant. Ses résultats peuvent donc être utilisés comme référence en termes de modularité.

La figure 2.1(a) illustre le gain de stabilité obtenu en supprimant les sommets d'un réseau un par un avec cette méthode. Elle est à rapprocher de la figure 1.1(b) que nous avons présentée précédemment. La distance d'édition entre les partitions successives est considérablement réduite, même par rapport à WalkTrap, et les communautés sont donc bien plus stables. Mis à part pour quelques sommets dont la suppression engendre des modifications plus importantes, de l'ordre de la centaine de sommets impactés, nous observons que, dans la majorité des cas, il ne se passe quasiment rien, c'est-à-dire que la structure communautaire change très peu. Cela semble bien plus satisfaisant.



(a) (b)

FIGURE 2.1 : (a) Distance d'édition entre deux partitions successives et (b) modularité de chaque partition durant une dynamique simulée qui consiste à supprimer les sommets un par un dans un ordre aléatoire.

Cependant, obtenir une méthode stable n'est pas suffisant car la qualité est aussi importante et il faut donc obtenir un compromis entre ces deux mesures. La figure 2.1(b) présente la modularité des trois méthodes. On observe que la qualité relative des partitions obtenues avec la méthode de Louvain stabilisée décroît avec le temps, comparativement à WalkTrap et à la méthode de Louvain. L'algorithme a en effet du mal à faire suffisament évoluer la décomposition en communautés pour réfléter les évolutions de la structure. La version modifiée reste meilleure que WalkTrap pendant une longue période car elle a un avantage initial, mais cet avantage disparaît petit à petit.

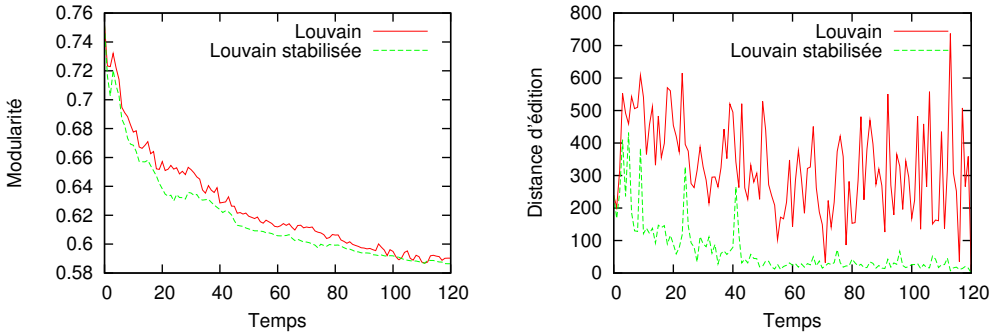
Il apparaît donc que l'approche suivie stabilise énormément l'algorithme mais que cela nuit à la qualité des résultats. Notamment, à chaque nouvelle exécution, on démarre l'algorithme avec une partition qui est certainement très proche d'un maximum local, ce qui laisse peu de liberté de mouvement. Pour que l'algorithme choisisse de déplacer un sommet d'une communauté à une autre ou de le remettre seul, il faut que le déplacement de ce sommet unique soit intéressant en termes de modularité. Or, c'est souvent le déplacement d'un ensemble de sommets qui fait la différence, ce que l'algorithme ne peut faire.

Pour relaxer la contrainte imposée par le choix de la partition initiale, il est possible de ne pas réinjecter exactement la partition obtenue à l'instant t quand on effectue le calcul à l'instant $t + 1$. On peut par exemple choisir une certaine fraction x des sommets et les sortir de leur communauté pour les remettre seuls. Plus cette fraction est élevée et plus l'algorithme retrouvera de liberté. À la limite, si $x = 1$, on retrouve exactement la méthode de Louvain. Bien entendu, cette liberté ne se retrouvera qu'au détriment de la stabilité mais des résultats expérimentaux [11] montrent qu'il est possible d'avoir un gain de modularité avec une perte de stabilité minimale en utilisant une peite valeur de x, de l'ordre de quelques pourcents.

Plus précisément, la valeur pertinente à choisir est liée à la dynamique du réseau. En particulier, si le réseau subit des variations importantes, alors la partition calculée à l'instant t sera mauvaise à l'instant $t+1$ et l'algorithme n'aura pas de difficultés à la faire évoluer, même avec $x = 0$. À l'inverse, si le réseau évolue peu, il faudra certainement prendre une valeur de x plus élevée pour donner plus de latitude à l'algorithme. Dans sa thèse [8], Thomas Aynaud propose de tester les valeurs extrêmes $x = 0$ et $x = 1$ afin d'avoir une référence sur la stabilité maximale et la qualité maximale atteignables, puis d'augmenter x petit à petit en partant de 0 jusqu'à observer un gain de modularité suffisant sans perte de stabilité trop importante.

Nous avons également étudié la méthode de Louvain stabilisée sur des réseaux réels ayant des dynamiques différentes, ce qui a confirmé les observations faites ci-dessus [8] . Les quatre réseaux que nous avons étudiés sont les suivants :

- Blogs : ce réseau est purement croissant, c'est-à-dire qu'aucun sommet ni lien ne disparaît. On observe sur ce réseau que la stabilité est naturellement liée à l'évolution du réseau et que les pics d'instabilité correspondent globalement à des instants de fortes modifications (voir figure 2.2). On identifie notamment un pic d'instabilité à l'instant 40 qui correspond à un changement de mesure.
- Mrinfo : ce réseau est très stable et a des communautés très marquées. La méthode de Louvain était déjà assez stable pour ce réseau, et la méthode de Louvain stabilisée l'est encore plus avec une perte de qualité négligeable.
- Radar et Imote : ces deux réseaux sont très dynamiques et la partition trouvée à un instant est en général de mauvaise qualité à l'instant suivant, ce qui implique qu'elle est remise en cause en profondeur. Malgré tout, au prix d'une perte de qualité raisonnable, on observe entre 3 et 4 fois moins de mouvements avec la stabilisation pour Radar (voir figure 2.3).



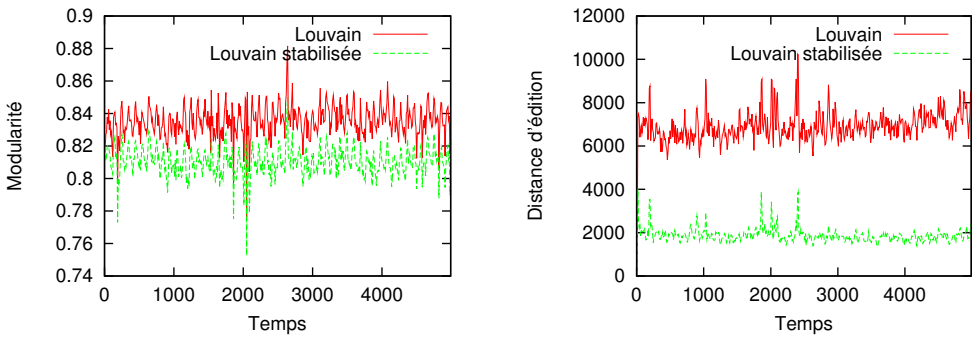
(a) (b)

FIGURE 2.2 : (a) Modularité et (b) distance entre deux partitions successives de la méthode de Louvain stabilisée sur le réseau Blogs.

2.1.2 Événements ou artéfacts ?

Il faut noter que, même avec la version la plus stable et dans le cas d'évolutions minimales comme sur la dynamique simulée, on observe toujours des pics d'instabilité, que nous appelons des *événements*. La question se pose alors de savoir si ces mouvements de masse sont significatifs et correspondent à des modifications majeures de la structure communautaire, ou s'il s'agit d'un artéfact lié à la modularité ou bien à l'algorithme.

Afin de trancher, nous avons décidé de ne supprimer qu'un sommet i quelconque et d'étudier l'impact de sa suppression sur la partition calculée en fonction de son importance. On espère que la suppression de sommets importants (fort degré, forte centralité, etc.) aura un impact



(a) (b)

FIGURE 2.3 : (a) Modularité et (b) distance entre deux partitions successives de la méthode de Louvain stabilisée sur le réseau Radar.

plus fort sur la structure communautaire que la suppression de sommets périphériques. Bien entendu, il faut être très prudent sur la notion d'importance. Donc, après avoir calculé une partition $\mathcal{P}$ sur le graphe, nous avons calculé pour chaque sommet i la partition $\mathcal{P}_i$ obtenue à l'aide de la méthode de Louvain stabilisée sur le graphe privé de i afin de mesurer la différence $\Delta(\mathcal{P}, \mathcal{P}_i)$ entre les deux partitions.

Nous avons ensuite tenté de corréler $\Delta(\mathcal{P}, \mathcal{P}_i)$ avec un grand nombre de propriétés classiques des graphes de terrain pour le sommet i. Aucune des propriétés que nous avons étudiées n'implique de corrélation nette, certaines distances élevées venant de sommets peu importants et inversement. Il semble cependant que les propriétés locales (centralité au sein de la communauté *versus* centralité au sein du graphe) soient plus corrélées, mais de manière trop faible pour que l'on puisse en tirer des conclusions. Cela signifie donc, soit que les événements observés sont un artéfact de la méthode, soit qu'il faut chercher les corrélations ailleurs.

Afin d'avancer sur ce point, nous avons étudié les sommets i pour lesquels $\Delta(\mathcal{P}, \mathcal{P}_i)$ est élevée afin de comprendre à quoi ce phénomène est dû, en particulier quels sont les sommets impactés et leur situation dans le graphe. On observe qu'en général, l'impact est proche du sommet supprimé, mais qu'il y a aussi de nombreux cas où cette distance est grande sans qu'il y ait de modifications à des distances intermédiaires. Ces cas semblent typiquement être des artéfacts de l'algorithme.

Nous avons finalement tenté d'isoler l'impact de la partition $\mathcal{P}$ injectée de celui du non déterminisme de la méthode de Louvain. Pour cela, nous avons d'une part injecté plusieurs fois la même partition $\mathcal{P}$ dans la méthode de Louvain et, d'autre part, injecté différentes partitions dans une version déterministe de la méthode de Louvain [1]. Les résultats obtenus montrent que si l'on garde la même partition initiale, indépendamment du fait que la méthode de Louvain soit déterministe ou pas, alors on obtient une distance $\Delta(\mathcal{P}, \mathcal{P}_i)$ constante mais qu'inversement si l'on change cette partition initiale, les distances varient pour un sommet i donné, même avec une version déterministe de la méthode de Louvain. La partition initiale a donc un impact fort dans les pics de distance observés et l'on ne peut pas se contenter d'étudier le sommet supprimé pour comprendre ce qui va advenir.

1. Il y a trois instants auxquels la méthode de Louvain fait des choix non-déterministes : ordre de traitement de sommets, choix en cas de gain de modularité équivalent lorsque l'on tente de déplacer un sommet et numérotation des communautés durant la deuxième phase. Des choix déterministes peuvent être faits dans chacun de ces cas.

2.1.3 Bilan

L'instabilité est au cœur des méthodes de détection de communautés, notamment du fait qu'il existe de nombreuses partitions de bonne qualité. Utiliser simplement les méthodes classiques dans le cadre dynamique ne donne donc pas de résultats pertinents. Pour y remédier, nous avons proposé une modification simple de la méthode de Louvain qui permet un gain de stabilité, au prix d'une perte de qualité des partitions.

Nous avons malheureusement constaté que malgré cette stabilisation, il y a toujours des pics d'instabilité que nous n'arrivons pas à expliquer et certains indices, tels que la non localité des modifications, nous laissent présager qu'une partie au moins des modifications seraient indépendantes des évolutions topologiques et seraient donc des artéfacts de la méthode de détection ou de la modularité. Nous avons aussi observé une dépendance forte à la partition initiale ce qui fait que, dans le cas d'une dynamique sur plusieurs pas de temps, la partition trouvée au tout premier instant va conditionner les résultats des instants suivants.

Un aspect positif, que nous allons explorer par la suite, est que si l'on étudie une dynamique faible et que l'algorithme utilisé est stable, alors une partition trouvée à un instant t est en général valable à l'instant $t+1$. À l'extrême, on pourrait considérer que la partition calculée à l'instant t est toujours valable à l'instant $t+1$, mais aussi à l'instant $t+2$, et aux instants suivants.

2.2 Communautés multi-pas

Comme nous venons de le voir, dès lors que le graphe n'a pas trop évolué, une partition de bonne qualité à l'instant t est certainement toujours valable à l'instant $t+1$. Inversement, la partition trouvée à l'instant $t+1$ est certainement bonne pour l'instant t aussi. Partant de ce constat simple, nous allons tenter de trouver quelle serait la meilleure partition à la fois à l'instant t et à l'instant $t+1$. Cette partition permettrait de décrire la structure sur un plus long terme. Bien entendu, si nous parvenons à trouver une partition de qualité sur deux instants de temps consécutifs, il est possible de généraliser pour plusieurs pas de temps. Il reste donc à définir précisément ces communautés valables sur plusieurs pas de temps, que nous appelons *communautés multi-pas*, et à proposer une méthode pour les calculer.

Soit $G = \{G_1, G_2, ..., G_n\}$ un graphe dynamique défini sur un ensemble de pas de temps $S = \{1, 2, ..., n\}$, où $G_i = (V_i, E_i)$ est l'instantané au pas de temps i ayant pour sommets V_i et pour liens E_i. On notera $V = \cup_{i\in\{1,...,n\}} V_i$ l'ensemble de tous les sommets et on cherchera une partition de V. Nous noterons $Q(G_i, \mathcal{P})$ la modularité de la partition $\mathcal{P}$ sur G_i en restreignant $\mathcal{P}$ à une partition de V_i.

Définition 1 *La modularité moyenne $Q_{avg}(G, \mathcal{P})$ de la partition $\mathcal{P}$ de V sur une fenêtre de temps $T \subseteq S$ est définie comme :*

$$Q_{avg}(G, \mathcal{P}, T) = \frac{1}{|T|} \sum_{i\in T} Q(G_i, \mathcal{P}) \quad .$$

Cette définition considère de la même manière tous les pas de temps, mais il est possible de les pondérer, si l'on souhaite indiquer que certains pas de temps sont plus importants selon des critères à définir.

Définition 2 *La modularité moyenne pondérée par $w = \{w_i\}_{i\in\{1,...,n\}}$ de la partition $\mathcal{P}$ de V durant la fenêtre de temps T est définie comme :*

$$Q_{avg}(G, \mathcal{P}, T) = \frac{1}{\sum_{i\in T} w_i} \sum_{i\in T} w_i \cdot Q(G_i, \mathcal{P}) \quad .$$

2.2.1 Méthodes de détection

La maximisation de la modularité moyenne est un problème NP-complet, comme la maximisation de la modularité [24]. Nous allons donc chercher à trouver des partitions de bonne modularité moyenne.

Nous proposons tout d'abord une adaptation de la méthode de Louvain qui consiste grossièrement à calculer en parallèle sur tous les instantanés les gains de modularité induits par les déplacements des sommets, puis à choisir le déplacement qui maximise le gain moyen. Ensuite, quand la méthode de Louvain agrège les communautés, il faut simplement le faire en parallèle sur tous les instantanés.

Nous avons testé une deuxième méthode plus rapide qui optimise une fonction proche de la modularité moyenne. Elle consiste à construire un graphe qui est l'union de tous les instantanés de T : chaque lien dans ce *graphe union* est pondéré par le nombre de pas de temps de T durant lesquels ce lien existe. On applique ensuite la méthode de Louvain directement sur ce graphe union et le résultat est la partition multi-pas. L'exécution est ici encore plus rapide car si l'on excepte la construction du graphe union, on n'a besoin que d'une exécution de la méthode de Louvain, indépendamment du nombre de pas de temps.

Cependant, cette méthode optimise une fonction qui n'est pas exactement la modularité moyenne. En effet, la modularité moyenne peut se réécrire comme :

$$Q_{avg}(\pi, G, T) = \sum_{c \in \Pi} \left(\sum_{t \in T} \frac{l_{tc}}{L_t} - \sum_{t \in T} \left(\frac{d_{tc}}{2.L_t} \right)^2 \right) \ ,$$

alors que la modularité sur le graphe union est égale à :

$$Q_{union}(\pi, G, T) = \sum_{c \in \Pi} \frac{\sum_{t \in T} l_{tc}}{\sum_{t \in T} L_t} - \left(\frac{\sum_{t \in T} d_{tc}}{2. \sum_{t \in T} L_t} \right)^2 \ .$$

On a donc des inversions entre fractions et sommes. Néanmoins, cette méthode est très rapide car les graphes unions sont en général bien plus petits que la somme des tailles des instantanés. Nous verrons par la suite que cette méthode est une bonne première approximation et donne des résultats de bonne modularité moyenne.

2.2.2 Modularité moyenne des différents réseaux

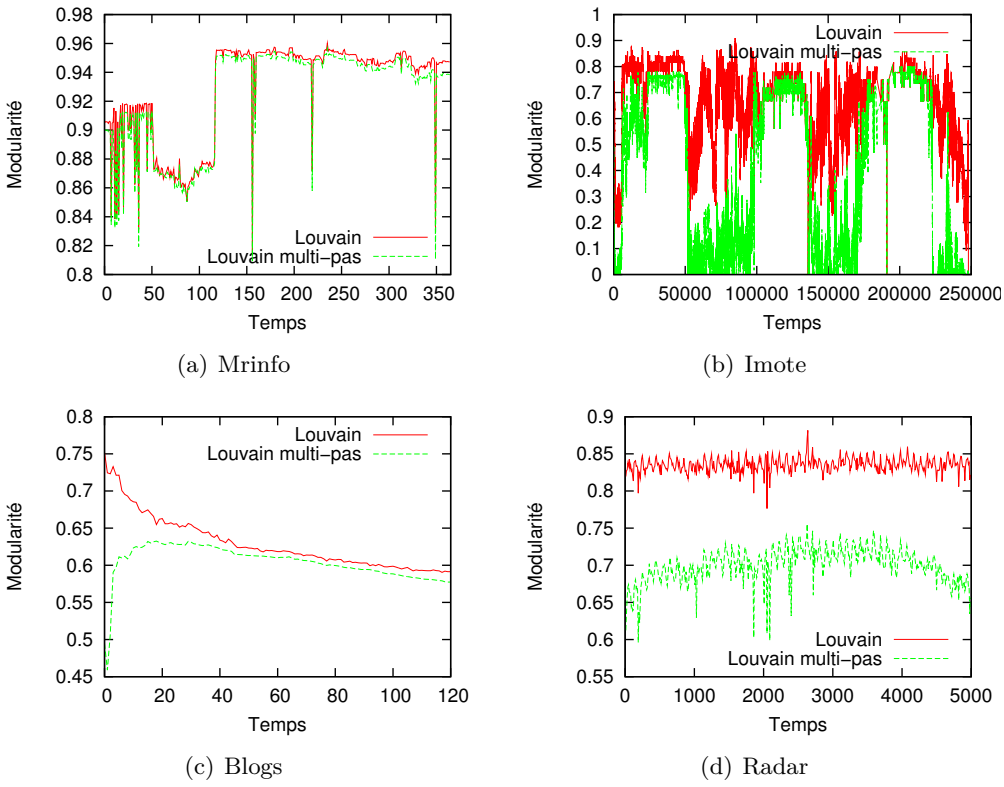
Pour étudier l'efficacité des algorithmes, nous avons comparé les qualités de plusieurs partitions en utilisant toute la durée de la mesure comme fenêtre de temps ($T = \{1, ..., n\}$) avec la méthode moyenne et la méthode union. À titre de comparaison, nous avons également calculé les partitions avec la méthode de Louvain à chaque instant indépendamment, ce qui donne une borne supérieure à ce que l'on peut obtenir avec les autres méthodes qui sont beaucoup plus contraintes.

	méthode union	méthode moyenne	méthode de Louvain
Blogs	0,60	0,61	0,62
Mrinfo	0,910	0,920	0,923
Imote	0,38	0,39	0,56
Radar	0,70	0,70	0,84

TABLE 2.1 : Modularités des différentes méthodes. La modularité pour la méthode de Louvain est la moyenne des modularités à chaque instant.

Le tableau 2.1 présente les modularités obtenues pour chacune des trois méthodes. Bien que les méthodes union et moyenne soient assez proches en termes de qualité, la méthode moyenne est toujours meilleure. La méthode union est cependant intéressante du fait de sa rapidité [2]. On observe également que sur certains réseaux, la qualité moyenne est très proche de la modularité obtenue en moyenne avec la méthode de Louvain alors que la modularité moyenne calcule une seule partition contrairement à la méthode de Louvain.

Plus précisément, la figure 2.4 compare à chaque instant t la modularité de la partition calculée avec la méthode de Louvain à la modularité de la partition moyenne (restreinte aux sommets de V_t).



(a) Mrinfo (b) Imote

(c) Blogs (d) Radar

FIGURE 2.4 : Modularités des différents partitions étudiées sur les jeux de données.

Sans rentrer dans les détails sur ces figures, on peut voir que sur les réseaux Mrinfo et Blogs, on arrive à trouver une partition multi-pas dont la qualité est quasiment optimale à chaque instant. Pour Imote et Radar, en revanche, ce n'est pas possible et la partition multi-pas a une qualité constamment en deça de l'optimal. Cela vient du fait que les deux premiers réseaux sont assez stables et que cette partition multi-pas est pertinente, alors que les seconds ne le sont pas.

En conclusion, l'efficacité de cette fonction de qualité dépend fortement des caractéristiques de la dynamique du graphe. Cependant, les contraintes imposées à la méthode sont très fortes et l'on imagine facilement que l'on cherche quelque chose qui n'existe pas en voulant trouver une unique partition pertinente pour l'évolution complète d'un réseau. En particulier, on peut penser que chaque réseau évolue par phases et que l'on aurait sans doute plus de succès en se limitant à ces phases, à condition de pouvoir les identifier.

2. Sur ces quatre réseaux le calcul est quasiment instantané avec la méthode union, une fois le graphe calculé, alors qu'avec la méthode moyenne ils vont de 30 secondes pour Blogs jusqu'à plus de 4 heures pour Radar qui possède de nombreux pas de temps.

2.3 Fenêtres de temps

Le fait de calculer la modularité moyenne sur toute la période est certainement une erreur sauf dans le cas particulier d'un graphe dont la structure reste globalement constante. Nous allons maintenant exploiter le fait que l'on peut choisir la fenêtre de temps sur laquelle se fait l'optimisation afin de se limiter à celles qui ont un sens. Il est important de noter qu'une fenêtre de temps n'est pas forcément constituée de pas de temps consécutifs : on peut vouloir par exemple retrouver une structure qui se répète chaque matin. Une fenêtre de temps T est donc de manière générale un sous-ensemble de S (l'ensemble des pas de temps).

Dans le réseau Imote, la structure change entre le jour (travail) et la nuit (sommeil), mais elle change aussi durant la journée en fonction des différentes sessions, repas et pauses. Outre l'aspect de contiguité temporelle déjà évoqué, la notion de hiérarchie est aussi importante : répétition des journées, constituées de sessions répétées, de pauses, etc. Ce genre de segmentation hiérarchique est habituellement représenté par un arbre dont les feuilles sont les pas de temps et dont chaque sommet intermédiaire correspond aux regroupements des pas de temps contenus dans les sous-arbres. La figure 2.5 illustre une telle décomposition sur un exemple fictif.

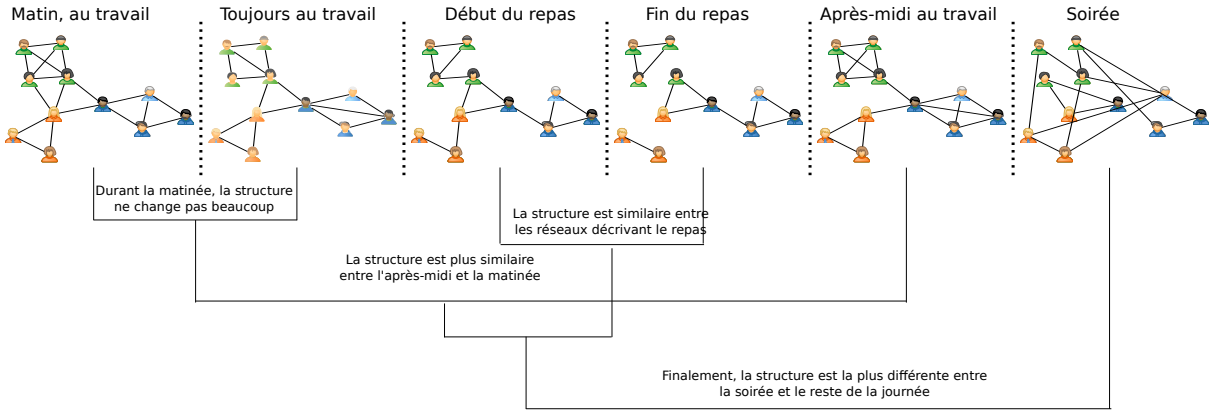


FIGURE 2.5 : Un exemple naïf de décomposition hiérarchique en pas de temps de l'évolution d'un réseau social. Durant la matinée, les gens travaillent et les pas de temps correspondants sont regroupés. Pendant le repas, la structure change à nouveau. Durant l'après-midi, les gens retournent au travail et donc la structure ressemble à la matinée, et ainsi de suite.

2.3.1 Calcul des fenêtres

Les algorithmes gloutons se prêtent bien à la segmentation hiérarchique : on commence par créer des fenêtres ne contenant qu'un pas de temps puis, de manière itérative, on regroupe les deux fenêtres les plus similaires jusqu'à avoir regroupé toutes les fenêtres ou avoir décidé que plus aucun regroupement pertinent n'est possible. Cet algorithme très classique nécessite uniquement une fonction de similarité entre fenêtres.

Dans notre cas, comme l'on souhaite optimiser la modularité moyenne, il est préférable de regrouper des fenêtres pour lesquelles le regroupement donne une bonne modularité moyenne. De manière intuitive, si la partition $\mathcal{P}_1$ d'une fenêtre T_1 est bonne pour une fenêtre T_2 de partition $\mathcal{P}_2$ et inversement, alors on peut considérer que les deux fenêtres sont similaires.

Définition 3 *Étant données deux fenêtres de temps T_i et T_j et leurs partitions correspondantes $\mathcal{P}_i$ et $\mathcal{P}_j$, la similarité entre T_i et T_j est définie comme :*

$$Sim(T_i, T_j) = Q_{avg}(G, \mathcal{P}_i, T_j) + Q_{avg}(G, \mathcal{P}_j, T_i)$$

Il faut remarquer que l'on n'effectue pas de comparaison directe des partitions à cause des phénomènes d'instabilité. Bien entendu, l'approche que nous avons suivie n'est pas la seule envisageable mais elle a le mérite de la simplicité.

Cette méthode construit donc une segmentation hiérarchique binaire (regroupement des fenêtres deux par deux) des pas de temps et fournit une forêt de fenêtres car on arrête l'aggrégation si la similarité est négative. La complexité de l'algorithme se calcule simplement, en effet s'il y a n pas de temps, il faut effectuer $2n$ détections de communautés en tout : une sur chaque pas de temps initiallement (avec la méthode de Louvain) puis pour chacun des n regroupements il faut recalculer la partition multi-pas. Dans la pratique, les graphes étant petits, chaque décomposition est presque instantanée et la principale limitation vient de la mémoire nécessaire pour stocker le graphe.

2.3.2 Résultats

Il faut noter que l'approche contiguë est très sensible et, s'il y a un instant qui est significativement différent des autres dans une suite de pas de temps, alors la vision contiguë va nécessairement casser la suite en deux sous-parties autour de cet instant. En contrepartie cela peut permettre d'identifier des ruptures de la structure de manière automatique. L'approche non contiguë semble donc plus pertinente dans l'absolu car elle permet justement de trouver des similarités à des moments différent, cependant l'analyse de telles segmentations est en général plus complexe.

Nous avons étudié plusieurs réseaux dynamiques et, pour chacun, nous avons calculé des segmentations avec des fenêtres de temps contiguës ou non. Nous illustrons les résultats sur deux exemples pour lesquels la méthode est pertinente.

La figure 2.6 présente la segmentation du réseau Mrinfo avec des fenêtres non contiguës. On observe des regroupements complexes mais en étudiant la segmentation en détail on peut retrouver trois phases de la dynamique qui sont connues sur ce réseau, ce qui valide que le découpage à du sens. On peut aussi remarquer que bien qu'aucune contrainte de contiguïté ne soit imposée, les fenêtres calculées sont, de fait, très souvent constituées de pas de temps consécutifs, ce qui paraît raisonnable car deux pas de temps consécutifs ont plus de chance d'être similaires structurellement.

Nous obtenons des résultats similaires avec Imote (figure 2.7). En imposant la contiguïté, les fenêtres les plus grandes peuvent facilement être identifiées à des journéees, des nuits, des sessions, etc. (nous n'avons pas la correspondance exacte entre les instants de l'expérience et les heures de la conférence et ne pouvons qu'émettre des hypothèses). Sur la version sans contiguïté imposée, on arrive même à retrouver des demi-journées, ce qui est très positif étant donnée la qualité des données. Il est cependant délicat d'extraire des informations de ces arbres car ils contiennent plusieurs dizaines de milliers de sommets.

Enfin, sur les deux derniers réseaux, on obtient des résultats plus mitigés. Sur Blogs, il est difficile d'identifier clairement quelque chose : le réseau se construisant uniquement par ajout de sommets et de liens, il y a peu de fenêtres pertinentes et la non-contiguïté n'apporte rien de plus. Sur Radar enfin, on ne dispose de quasiment aucune donnée permettant de valider les fenêtres obtenues. Seuls quelques événements connus sont bien identifiés.

Les résultats obtenus sont prometteurs. Les décompositions obtenues sont souvent pertinentes et identifient clairement des phases, quand elles existent, et les deux approches avec ou sans fenêtres contiguës sont complémentaires. Les premières sont plus simples à interpréter et indiquent clairement des moments où la structure change radicalement, alors que les secondes fournissent des résultats plus fins, moins contraints temporellement et pouvant détecter des structures répétées.

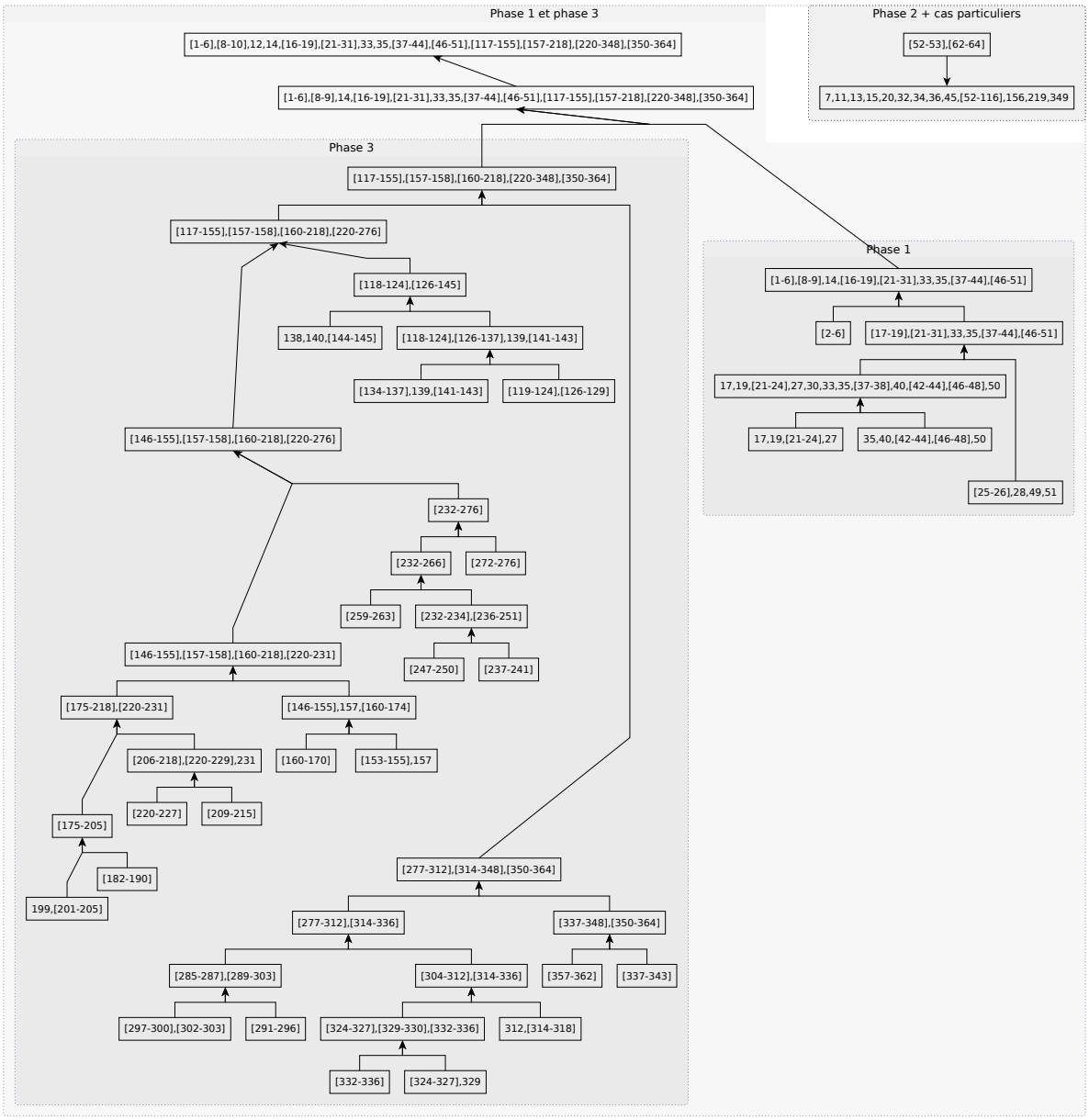


FIGURE 2.6 : Segmentation hiérarchique de Mrinfo. Les valeurs dans les sommets de la forêt représentent les fenêtres de temps : $[a-b]$ représente l'intervalle des pas de temps entre a et b, et a,b représente l'union de a et b. Les trois phases connues sur ce réseau, [1-51], [52-116] et [117-365] sont indiquées sur la figure. Ces phases sont facilement visibles sur la figure 2.4(a).

2.4 Synthèse

Nous avons montré dans ce chapitre que malgré l'existence de nombreuses partitions de bonne modularité, il est possible de calculer des communautés dynamiques sans trop d'instabilité. Nous avons pour cela proposé deux méthodes basées sur le même principe : si une partition est bonne à un instant, alors la même partition ou une partition très proche sera toujours valable à l'instant suivant, sous réserve que le graphe évolue peu.

Partant de ce principe, nous avons tout d'abord proposé de fournir à la méthode de Louvain une partition initiale correspondant à ce qui avait été calculé à l'instant précédent afin de la

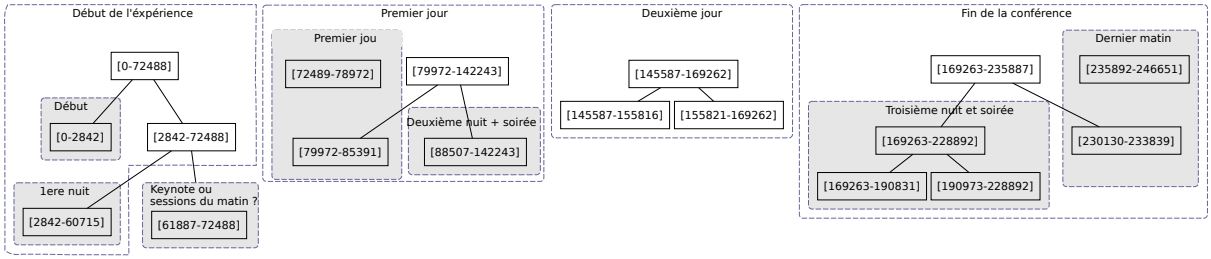


FIGURE 2.7 : Premiers niveaux de la décomposition hiérarchique du temps sur Imote. L'arbre étant trop grand, nous ne représentons donc ici que les premiers niveaux après un filtrage partiellement manuel.

guider dans l'optimisation. Nous avons montré que cela stabilisait grandement la méthode avec une perte de qualité assez réduite comparativement. Malheureusement, les quelques instabilités restantes ne sont pas compréhensible avec nos connaissances actuelles.

Nous avons ensuite proposé une autre version dans laquelle la partition est figée pendant plusieurs pas de temps, en espérant qu'elle reste de bonne qualité. Ceci soulève de nombreuses questions, comme la définition de la qualité sur plusieurs pas de temps, les moyens pour optimiser cette qualité et les pas de temps qu'il convient de choisir pour que cela ait un sens. Nous avons répondu à ces différentes questions pour parvenir à une méthode qui est capable d'identifier des fenêtres de temps imbriquées pertinentes et, pour chacune d'elles, de fournir une partition qui est globalement de bonne qualité pendant la durée de la fenêtre de temps.

Les deux approches ont été validées sur des données réelles et nous avons montré que l'on est capable, avec des performances plus ou moins bonnes, de trouver des événements connus par ailleurs, d'identifier des phases dans l'évolution et donc de fournir des partitions en communautés qui ont du sens pour les réseaux étudiés.

L'approche consistant à chercher des fenêtres pertinentes peut être modifiée de diverses manières et des améliorations sont possibles. Tout d'abord, la fonction de qualité choisie est loin d'être parfaite car elle mesure en même temps la similarité entre les partitions et la nature modulaire des graphes sous-jacents : deux fenêtres de temps identiques seront d'autant plus proches que leur structure est modulaire.

L'autre axe d'amélioration dans la définition de la similarité vient du fait qu'en faisant la somme des modularités, on perd toute notion de causalité : la succession des pas de temps G_1, G_2, G_3 est strictement équivalente à la succession G_3, G_2, G_1. Trouver une métrique pouvant prendre cela en compte donnerait très certainement des résultats plus pertinents. Cela pourrait passer par des jeux de pondération.

Ces différentes approches, malgré les problèmes de stabilité résiduels, sont cependant pertinentes pour calculer des communautés dynamiques et permettent de suivre simplement l'évolution de ces communautés. Nous verrons au chapitre suivant une solution pour tenter de circonvenir le problème de la stabilité.

CHAPITRE 3

CŒURS DE COMMUNAUTÉS

La modularité, malgré sa popularité, possède un certain nombre de défauts comme nous avons pu le voir précédemment. L'un des défauts principaux est lié au fait qu'il existe de nombreuses partitions de qualité comparable. Ainsi, un algorithme classique confronté à deux graphes isomorphes peut fournir des partitions significativement différentes. La significativité des partitions obtenues n'est alors pas évidente et trouver « la bonne partition » n'a pas forcément de sens. Même dans le cas où l'on arriverait à trouver l'optimum de modularité, comment justifier que, pour un millième de point de modularité en plus, on en préfère nettement une ?

Nous proposons ici une approche orthogonale qui consiste à voir l'existence de différentes partitions de bonne qualité comme une force. En effet, bien que ces partitions soient différentes, elles présentent en général un certain nombre de similitudes. L'étude de celles-ci fournit des informations sur des zones du graphe qui sont très robustes aux perturbations, et aux changements d'algorithme ou de conditions initiales. Ces similarités entre partitions nous permettrons de définir des *cœurs de communautés*.

L'étude de ces cœurs était centrale dans la thèse de Massoud Seifi. Dans ce chapitre nous montrerons leur pertinence dans des cas concrets, leur utilité dans la comparaison avec les graphes aléatoires [81] et dans le suivi de communautés dans les graphes dynamiques [82].

3.1 Approches similaires

Combiner différents résultats afin d'obtenir une solution plus consensuelle n'est pas une nouvelle idée [34]. Dans le cas des communautés cela peut permettre d'identifier des ensembles de sommets plus stables et plus significatifs. Comme la plupart des algorithmes de détection de communautés sont déterministes, cela implique souvent de casser ce déterminisme, ce qui a été principalement fait via des perturbations plus ou moins fortes du réseau étudié. L'idée consiste à générer de nombreux réseaux similaires au réseau étudié en effectuant de petites perturbations. Ensuite, les communautés sont détectées dans chaque réseau, puis agrégées. Parmi les méthodes de perturbation, on peut citer :

- le fait de renuméroter les sommets. Ceci tire parti du fait que la plupart des algorithmes traitent les sommets dans un ordre donné et que le modifier change le résultat. Cette méthode est utilisée notamment dans [59, 91] ;
- le recâblage de lien [54], où une fraction α des liens sont recâblés en essayant de respecter au maximum la distribution des degrés du graphe original. En fonction du paramètre α, le réseau est plus ou moins similaire au réseau de départ ;
- la modification du poids de chaque lien suivant diverses lois [45, 78].

Une fois les différentes partitions générées, il faut en extraire les communautés consensuelles. Ceci peut se faire en regardant pour chaque paire de sommets (i, j) si les deux sommets sont

classés dans la même communauté ou dans des communautés différentes. Il est alors possible de définir une matrice de taille $n \times n$, que nous noterons $P_{ij}^{\mathcal{N}} = [p_{ij}]_{n\times n}^{\mathcal{N}}$, telle que les éléments p_{ij} de cette matrice représentent la fréquence d'appartenance de i et j à la même communauté. À partir de cette matrice,

- les auteurs de [78] cherchent des ensembles maximaux de sommets qui sont regroupés dans 95% ($p_{ij} > 95\%$) des échantillons au moins ;
- dans [45], seuls les liens de fréquence p_{ij} supérieure à un seuil donné sont conservés et les composantes connexes de ce réseau sont les communautés finales.

Une idée supplémentaire dans [45] consiste à comparer le p_{ij} des liens internes à une communauté à celui des liens externes : si les liens internes ont un $p_{ij} = 1$ et que les liens entre communautés ont un $p_{ij} = 0$, alors la communauté est très stable. De manière plus générale les auteurs proposent de regarder la stabilité globale d'un ensemble de partitions avec :

$$S = \frac{-1}{m} \sum_{i,j} \{p_{ij} \log_2 p_{ij} + (1 - p_{ij}) \log_2(1 - p_{ij})\} \ ,$$

dont la valeur vaut 1 si tous les p_{ij} valent $0,5$, et 0 si les p_{ij} valent soit 0 soit 1.

Enfin, de nombreuses approches similaires en fouille de données combinent les résultats de plusieurs *clusterings* afin d'obtenir un regroupement consensuel plus significatif (voir par exemple [17, 43]).

Les travaux présentés dans la suite sont comtemporains de ceux présentés précédemment et un certain nombre de points sont similaires, notamment sur les principes de base. Néanmoins, ils apportent quelques variations, en particulier sur le fait qu'aucun seuil n'est fixé, que toutes les paires de sommets sont considérées et pas seulement les paires reliées. Enfin, nous montrerons que cette approche permet de constater l'absence de communautés stables sur des graphes aléatoires et qu'elle peut servir de base pour l'étude des communautés dynamiques.

3.2 Méthodologie

De manière intuitive, un cœur de communauté est un ensemble de sommets qui ont une forte tendance à être assignés à la même communauté et ce, indépendamment de l'algorithme de détection utilisé, de l'ordre de traitement des sommets et d'autres perturbations mineures du réseau.

Idéalement, pour évaluer ces cœurs de manière fiable, il faudrait pouvoir calculer toutes les partitions de bonne qualité, chacune apportant une information pertinente sur la structure communautaire du réseau. Une approximation simple consiste à appliquer $\mathcal{N}$ fois un algorithme de détection de communautés de sorte qu'il fournisse des partitions différentes via l'une des méthodes présentées précédemment. Nous nous basons ensuite sur la matrice des p_{ij} et, dans la suite, nous travaillerons essentiellement sur cette matrice qui contient une information très riche. Il faut noter que la matrice des p_{ij} peut être vue comme la matrice d'adjacence d'un graphe de co-appartenance.

Si l'on souhaite réellement obtenir des cœurs, étant donné un seuil $\alpha \in [0, 1]$, on peut supprimer tous les liens de poids $p_{ij} < \alpha$ de ce graphe de co-appartenance. Les composantes connexes de ce graphe seuillé sont les α-cœurs de communautés. Il faut noter que la méthode présentée ne consiste pas à chercher des ensembles de sommets qui se connectent *tous* les uns aux autres avec un $p_{ij} \geq \alpha$ car cela reviendrait à chercher des cliques dans le graphe seuillé et les cœurs pourraient alors se recouvrir (ce qui serait pertinent, mais ce n'est pas l'objectif ici). La figure 3.1 illustre cette procédure.

Le calcul des cœurs tel que présenté utilise plusieurs paramètres :

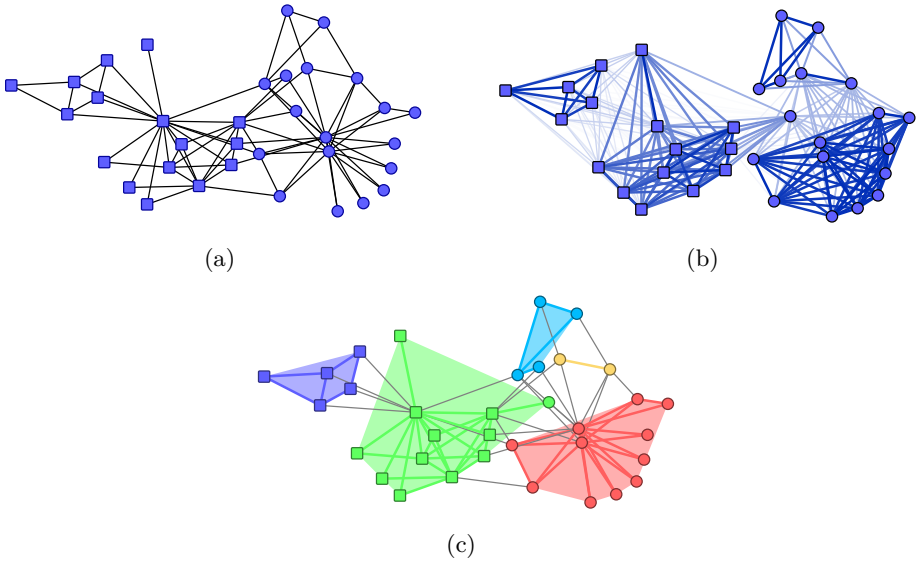


FIGURE 3.1 : Calcul des cœurs : (a) le graphe original, (b) le graphe de co-appartenance pour lequel le poids d'un lien (i, j) est la fréquence d'appartenance de i et j à la même communauté et (c) les cœurs de communautés après la suppression des liens de poids inférieur à un seuil α arbitraire.

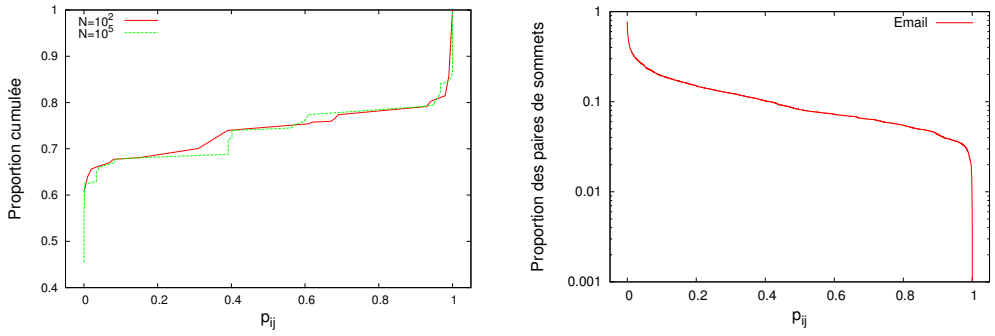
- le choix de l'algorithme de détection de communautés : des études préliminaires montrent que le choix de l'algorithme n'a pas une influence significative sur les résultats. Cependant, il faut exécuter de nombreuses fois l'algorithme sur des graphes de grande taille. Pour ces raisons, nous avons privilégié la méthode de Louvain qui est rapide et performante ;
- la méthode de non-déterminisation : la méthode de Louvain traitant les sommets dans un ordre aléatoire, aucune modification particulière n'est nécessaire ;
- le nombre d'exécutions $\mathcal{N}$: il faut pouvoir obtenir suffisamment de partitions distinctes sans augmenter la complexité outre mesure. De manière empirique, on peut voir que la matrice des p_{ij} converge polynômialement lorsque $\mathcal{N}$ augmente. Plus le graphe est grand plus il faudrait de partitions mais, en pratique, on fera l'inverse pour des raisons évidentes de complexité, on utilisera donc quelques centaines ou quelques milliers d'exécutions selon la taille du graphe considéré ;
- le choix du seuil α : le choix d'un seuil est complexe et le caractère multi-échelle de la structure communautaire n'incite pas à aller dans cette direction. Nous regarderons l'éventail complet des valeurs de α par la suite.

3.2.1 Pertinence et caractéristiques des cœurs

Afin d'établir la pertinence des cœurs, il convient tout d'abord de vérifier que la matrice des p_{ij} contient effectivement des valeurs intéressantes. Pour cela, nous nous sommes intéressés aux distributions des fréquences des p_{ij}. Nous avons appliqué notre algorithme au graphe du club de karaté de Zachary [95]. La figure 3.2(a) montre les distributions cumulatives de la proportion de paires de sommets qui sont dans le même groupe pour 100 et 10^5 exécutions. On peut voir sur cet exemple que la majorité des paires sont presque toujours soit groupées ($p_{ij} = 1$) soit séparées ($p_{ij} = 0$), mais qu'il existe aussi quelques paires de sommets qui sont parfois ensemble et parfois séparées. L'existence de ces sommets à la frontière des communautés a été utilisée pour chercher des partitions recouvrantes, comme par exemple dans [90].

Sur des graphes issus de domaines différents et de tailles différentes, notamment des réseaux de collaboration, des réseaux d'emails et l'Internet, nous observons que même sur les graphes

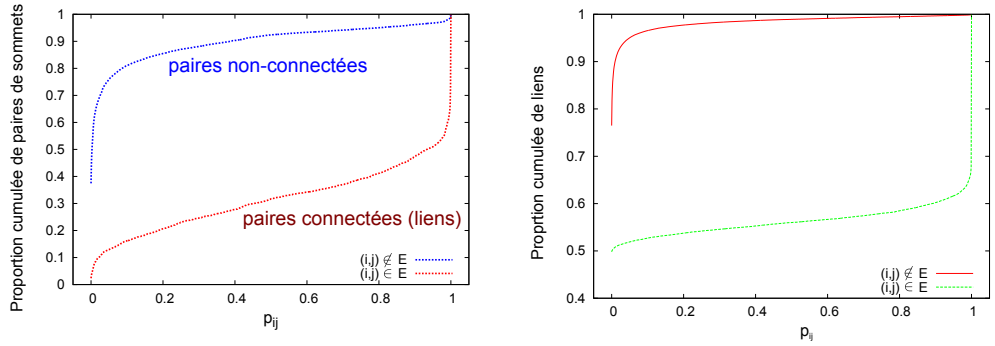
de grande taille un grand nombre de paires ne sont jamais ensemble, et qu'un nombre faible mais non négligeable de paires sont toujours ensemble. La figure 3.2(b) présente un exemple sur un réseau d'échange d'emails.



(a) (b)

FIGURE 3.2 : (a) Distribution cumulative des p_{ij} pour le graphe du club de karaté de Zachary et (b) distribution cumulative inverse des p_{ij} pour un réseau d'échange d'emails.

En particulier, si l'on étudie les paires de sommets qui sont toujours regroupés, il s'agit surtout de sommets reliés dans le graphe original, comme illustré par la figure 3.3. Inversement, la majorité des paires de sommets ayant un $p_{ij} = 0$ correspondent à des sommets non reliés dans le graphe original. Plus précisément, on observe sur ces deux exemples que la moitié environ des paires de sommets liés ont un p_{ij} très proche de 1 et que, au contraire, la majorité des non-liens ont un p_{ij} très proche de 0.



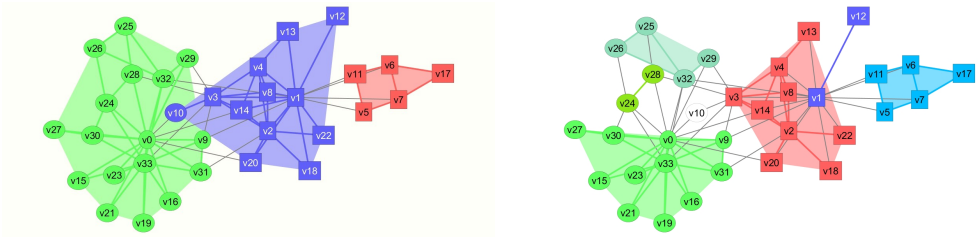
(a) Réseau d'emails (b) Réseau de collaboration

FIGURE 3.3 : Distribution cumulative des p_{ij} pour des paires de sommets, liés ou non.

Structure hiérarchique des cœurs

Lorsque l'on fait varier le seuil α, le nombre de cœurs varie aussi, ce qui induit une structure hiérarchique. On peut calculer les différents niveaux de cette hiérarchie en supprimant les liens du graphe de co-appartenance par p_{ij} croissant. À chaque fois que la suppression d'une paire déconnecte une composante connexe, un nouveau niveau est créé dans le dendrogramme. À titre illustratif, les cœurs du graphe du club de karaté de Zachary, identifiés en utilisant deux seuils différents, sont présentés sur la Figure 3.4.

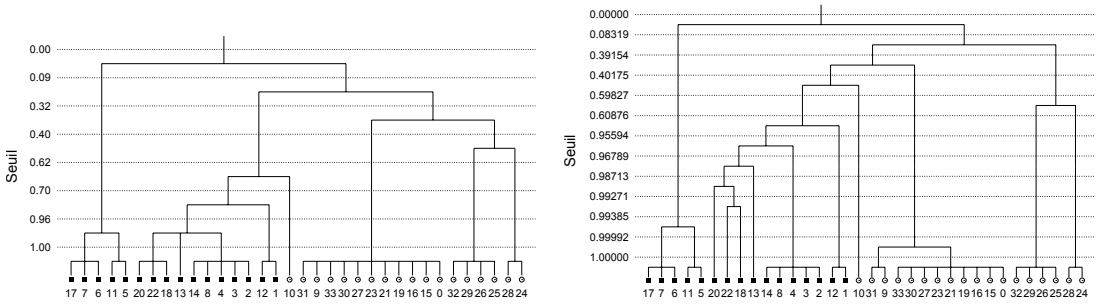
La figure 3.5 montre le dendrogramme du club de karaté pour deux valeurs de $\mathcal{N}$. Sur cette figure, la forme des sommets (ronde/carrée) représente la classification manuelle faite par



(a) $\alpha = 0,39$ (b) $\alpha = 0,96$

FIGURE 3.4 : Cœurs identifiés sur le club de karaté de Zachary en utilisant deux seuils différents.

Zachary. Comme on peut le voir, la division trouvée par l'algorithme avec $\mathcal{N} = 100$ et $\alpha = 0.32$ correspond presque parfaitement à ces groupes manuels. Il faut cependant noter que la division la plus proche de la division réelle ne se fait pas en deux cœurs mais en trois, et que chacun des trois cœurs ne contient que des sommets du même groupe manuel, à l'exception du sommet 10 qui est mal classé [1]. À titre de comparaison, le découpage optimal en termes de modularité est composé de quatre groupes, mais aucune valeur de α ne permet de retrouver ces 4 communautés.



(a) $\mathcal{N} = 100$ (b) $\mathcal{N} = 10^5$

FIGURE 3.5 : Structure hiérarchique des cœurs de communautés pour le club de karaté de Zachary pour (a) $\mathcal{N} = 100$ et (b) $\mathcal{N} = 10^5$. La forme des sommets (ronde ou carré) représente la classification manuelle faite par Zachary.

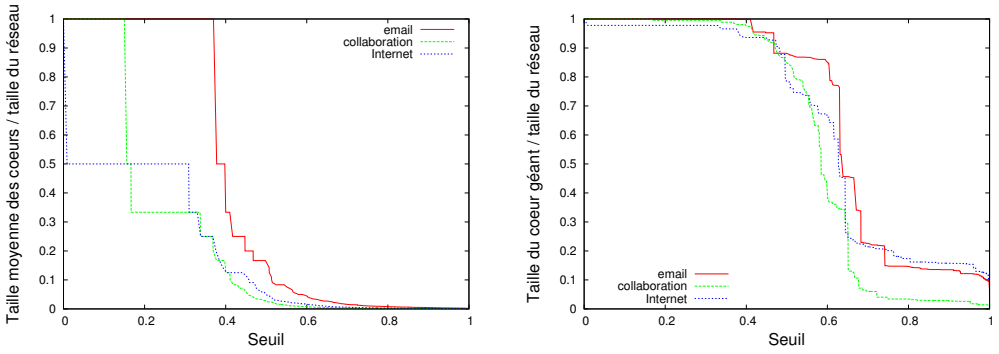
On observe que le choix d'un $\mathcal{N}$ très élevé ne donne pas nécessairement de meilleurs résultats. En effet, lorsque le nombre d'exécutions augmente, la probabilité de tomber sur une partition de très mauvaise qualité augmente aussi, ce qui perturbe la matrice des p_{ij}. On a donc en général plus de découpages et plus de niveaux de hiérarchie quand $\mathcal{N}$ augmente. Au contraire, une valeur faible de $\mathcal{N}$ ne permet pas à l'algorithme de converger et les cœurs n'auront pas forcément de sens et auront les mêmes défauts que les communautés elles-mêmes.

Nombre et taille des cœurs

Le choix du seuil α a une influence directe sur le nombre et la taille des cœurs. Comme la figures 3.6(a) l'illustre, avec un seuil proche de zéro on obtiendra des cœurs de grande taille (parfois plus grands que les communautés elles-mêmes), alors qu'un seuil très élevé conduira à des cœurs très petits et en particulier beaucoup de cœurs ne contenant qu'un seul sommet (dits cœurs triviaux). Nous observons aussi sur la figure 3.6(b) qu'avec un seuil jusqu'à $0,5$, nous avons un cœur contenant la grande majorité des sommets. Lorsque le seuil augmente, ce cœur éclate rapidement en plusieurs cœurs plus petits. Sur certains réseaux réels comme celui de

1. Pour rappel, ce sommet ne peut pas être classé correctement car les choix de ce sommet sont contraires à ses relations sociales, comme expliqué par Zachary [95].

l'Internet ou le réseau d'emails, on observe toujours des grands cœurs contenant jusqu'à 10% des sommets, même avec un seuil égal à 1. Ceci est généralement le signe de communautés très marquées.



(a) (b)

FIGURE 3.6 : (a) Taille relative moyenne des cœurs et (b) taille du plus grand cœur en fonction du seuil pour le graphe du club de karaté de Zachary.

3.2.2 Significativité des cœurs

L'une des méthodes classique pour évaluer la significativité d'une partition est d'évaluer sa ressemblance avec une classification issue de la réalité de terrain généralement faite « à la main » par un expert du domaine. Une partition sera d'autant meilleure qu'elle sera proche de cette classification. Un grand nombre de critères d'évaluation et de mesures de similarité entre deux partitions d'un même ensemble d'objets ont été proposés [72] et le choix d'une mesure appropriée est par conséquent difficile à faire. Nous présenterons dans la suite des résultats utilisant l'information mutuelle normalisée [86] et la distance d'édition, mais nous avons vérifié que d'autres mesures donnent des résultats qualitativement similaires.

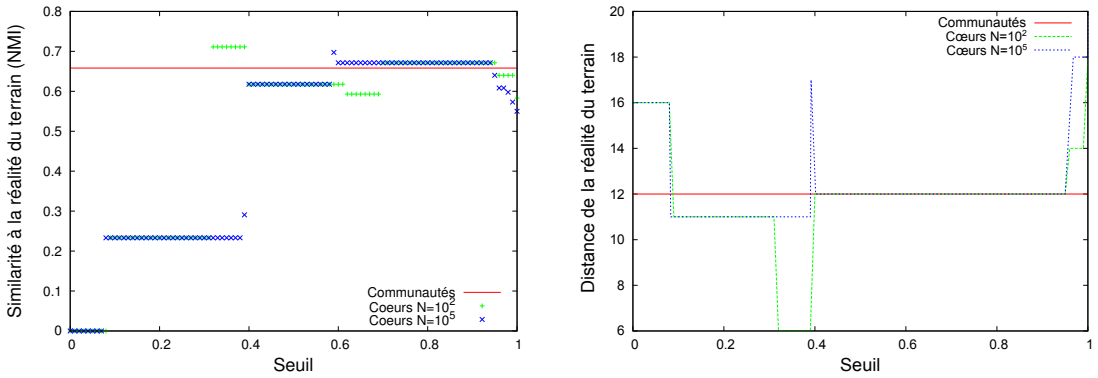
Afin d'évaluer la performance de notre approche, nous avons donc comparé la similarité des communautés (trouvées par l'algorithme classique de Louvain) et des cœurs de communautés à la classification issue de la réalité du terrain. Ceci a été fait pour des réseaux artificiels générés par le modèle proposé par Girvan-Newman [46] et plusieurs réseaux réels dont la « vraie » structure communautaire est connue. Nous ne présentons que les résultats pour le graphe du club de karaté de Zachary, mais les résultats sont similaires sur tous les exemples que nous avons étudiés.

Sur ce graphe, pour certaines valeurs du seuil entre $0,3$ et $0,4$, la classification manuelle est presque parfaitement identifiée (voir figure 3.7). Sur d'autres graphes, il y a toujours une valeur du seuil pour laquelle les cœurs sont plus proches de la réalité de terrain que les communautés, mais c'est en général pour des valeurs du seuil plus élevées, comprises entre $0,7$ et 1.

Ces résultats confirment la pertinence de la structure communautaire multi-échelle formée par les cœurs de communautés, non pas en remplacement des communautés classiques mais en complément. Dans ce qui va suivre, nous montrerons leur utilité dans deux contextes différents.

3.3 Cœurs dans les graphes aléatoires

Une utilisation classique des graphes aléatoires consiste à savoir si une propriété observée sur un graphe réel est normale (attendue). Par exemple, nous avons vu que la notion de modularité, utilisée pour évaluer la qualité d'une partition, est basée sur une comparaison à des graphes



(a) Information mutuelle normalisée (NMI)

(b) Distance d'édition

FIGURE 3.7 : Comparaison de la similarité entre les cœurs et les communautés par rapport à la réalité de terrain pour le club de karaté de Zachary, pour (a) l'information mutuelle normalisée et (b) la distance d'édition.

aléatoires. Dans ce contexte, une partition est de qualité si elle est sensiblement meilleure que la même partition sur un graphe aléatoire. Or, nous avons vu dans la chapitre 1 que malgré cette comparaison aux graphes aléatoires, il est possible de trouver des partitions ayant une bonne modularité dans des graphes aléatoires.

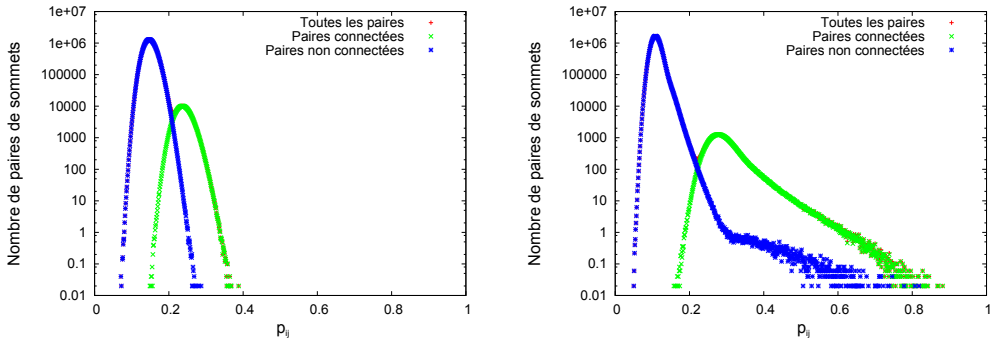
Un bon algorithme de détection de communautés devrait être capable de reconnaître que ces communautés dans des graphes aléatoires ne sont pas de vraies communautés et devrait donc, de manière plus générale, indiquer à la fois la présence et l'absence d'une structure communautaire. Nous montrons maintenant que les cœurs le permettent. Pour cela, nous utilisons deux modèles de graphes aléatoires parmi les plus simples : le modèle d'Erdős-Rényi [35] qui respecte le nombre de sommets et de liens et le modèle configurationnel [16] qui respecte en outre la distribution des degrés.

3.3.1 Distribution des p_{ij} pour les graphes aléatoires

La figure 3.8 présente la distribution des p_{ij} pour toutes les paires de sommets, les paires de sommets reliés et les paires de sommets non reliés pour des graphes aléatoires de degré moyen respectivement 20 et 100. Dans le cas d'un degré moyen élevé (figure 3.8(a)), nous observons que la distribution des p_{ij} est une combinaison de deux lois homogènes, une pour les paires de sommets connectés et une pour les paires de sommets non connectés. Dans le cas d'un degré moyen plus faible (figure 3.8(b)), les lois sont beaucoup moins homogènes et le phénomène s'amplifie avec la diminution du degré moyen.

En faisant varier la taille des graphes, nous observons qu'à degré moyen constant, plus le graphe est de grande taille, plus la variance est faible. À la limite, n tendant vers l'infini, on peut donc supposer que la distribution des p_{ij} serait piquée autour de deux valeurs, une pour les paires connectées et une pour les paires non connectées. Cette remarque aura un impact fort dans les prochains paragraphes.

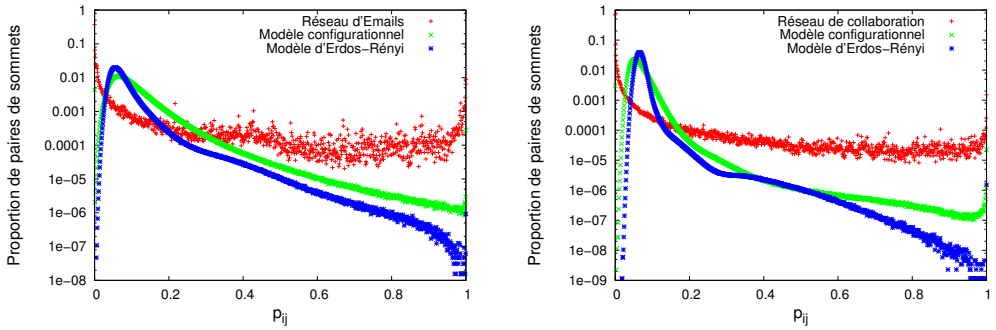
Nous avons aussi comparé la distribution des p_{ij} d'un réseau d'emails et d'un réseau de collaboration avec des graphes aléatoires de la même taille construits par le modèle d'Erdős-Rényi et par le modèle configurationnel. La figure 3.9 montre ces distributions. Nous avons vu précédemment que l'on observe généralement une forte proportion de paires de sommets avec un p_{ij} nul ou proche de 1. Au contraire, dans les graphes aléatoires, on n'observe presque aucune paire de sommets avec un p_{ij} nul, c'est-a-dire qu'étant donné deux sommets quelconques du



(a) degré moyen 100 (b) degré moyen 20

FIGURE 3.8 : Distributions des p_{ij} pour toutes les paires de sommets, uniquement les paires de sommets reliés et uniquement les paires de sommets non réliés pour des graphes aléatoires ayant 1000 sommets. La courbe avec toutes les paires est confondue avec les deux autres.

graphe, il y a au moins une exécution qui les a regroupés et ce, quelle que soit leur position dans le graphe. Inversement, il y a très rarement des paires de sommets qui sont toujours ensemble dans le modèle d'Erdős-Rényi et à peine plus avec le modèle configurationnel. Ces dernières sont majoritairement dûes à la présence de nombreux sommets de degré 1 qui sont toujours placés dans la communauté de leur unique voisin.



(a) Emails (b) Collaboration

FIGURE 3.9 : Distribution des p_{ij} pour deux réseaux réels et des réseaux aléatoires comparables générés avec les modèles d'Erdős-Rényi et configurationnel.

3.3.2 Propriétés des cœurs

La figure 3.10 représente la taille moyenne des cœurs, en fonction du seuil, pour un réseau de collaboration et des graphes aléatoires de la même taille générés par nos deux modèles. On observe que le choix d'un seuil faible conduit à un seul cœur contenant tous les sommets. Puis, après une transition de phase rapide, on n'observe que des cœurs triviaux contenant un seul sommet. On peut donc identifier les graphes qui n'ont pas de structure communautaire : ils n'ont pas de décomposition intermédiaire pertinente.

L'existence de la transition de phase est simple à montrer avec des hypothèses de champ moyen [2]. On montre que les valeurs des p_{ij} sont plus élevées pour les paires de sommets reliés

2. La théorie du champ moyen, courante en physique statistique, consiste à remplacer toutes les interactions entre particules par un unique champ externe. Ce champ externe correspond à l'interaction moyenne exercée sur

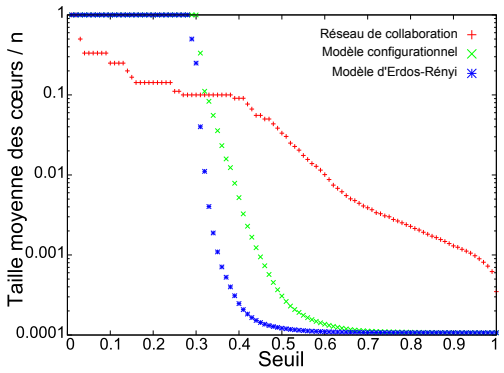


FIGURE 3.10 : Taille relative moyenne des cœurs pour un graphe réel et deux graphes aléatoires de même taille.

que pour les paires de sommets non reliés et qu'elles sont très concentrées autour d'une valeur moyenne. Partant de là, avec un seuil en dessous de cette valeur moyenne, toutes les paires sont conservées alors qu'au-delà de ce seuil, aucune paire n'est conservée, d'où l'existence d'une transition de phase. Une version détaillée de cette preuve est disponible dans [80].

Le moment précis où cette transition de phase a lieu peut aussi être calculé et correspond à la proportion de liens intra-communautaires dans une partition de bonne qualité d'un graphe aléatoire. Afin de pouvoir obtenir une preuve complète, il serait donc nécessaire de pouvoir calculer cette proportion de liens intra-communautaires. À l'heure actuelle, c'est un problème ouvert qui est généralement abordé uniquement de manière numérique [50].

3.4 Dynamique des cœurs

L'approche classique qui consiste à appliquer des algorithmes statiques à chaque instant de l'évolution d'un réseau dynamique, de manière indépendante, donne généralement de mauvais résultats comme nous l'avons montré dans le chapitre 1 et les méthodes stabilisées ne résolvent pas entièrement la problème comme nous l'avons vu dans le chapitre 2.

Nous allons donc maintenant réévaluer cette méthode avec les cœurs, tout d'abord sur des dynamiques simulées simples puis sur un cas réel. Nous comparerons à chaque fois les modifications engendrées par les évolutions pour des communautés et des cœurs afin de montrer que les cœurs semblent mieux adaptés que les communautés à l'étude de la dynamique. Les travaux présentés ci-dessous sont préliminaires et il reste beaucoup de travail avant d'en arriver à des méthodes permettant effectivement de suivre l'évolution de cœurs au cours du temps.

3.4.1 Impact de la dynamique sur la structure hiérarchique

Etant donné un graphe dynamique G composé d'une suite de graphes G_t, à chaque instant t de son évolution, nous considérons $P^{\mathcal{N}}_{ij\ G_t}$ la matrice des p_{ij} du graphe G_t à l'instant t.

L'impact de la dynamique du graphe peut être étudié à trois niveaux : sur le graphe complet, sur un sommet et sur une paire de sommets. Dans le premier cas, on peut calculer simplement la distance euclidienne entre les matrices avant et après l'évolution :

une particule par le reste du système. Cela revient, entre autres, à négliger toutes les fluctuations et les éventuelles corrélations. On perd donc en précision de la description, mais les résultats sont généralement beaucoup plus simples à obtenir.

$$distance(P^{\mathcal{N}}_{ij\ G_t}, P^{\mathcal{N}}_{ij\ G_{t+1}}) = \sqrt{\sum_{i,j \in (V_t \cap V_{t+1})} |p_{ij_{G_t}} - p_{ij_{G_{t+1}}}|^2} \ .$$

On peut également calculer la différence entre les deux matrices $\Delta p_{ij} = p_{ij_{G_t}} - p_{ij_{G_{t+1}}}$. Si l'on souhaite étudier l'impact de l'évolution sur un sommet i, on peut calculer la somme des valeurs absolues de la i-ème ligne de cette matrice :

$$I(i) = \sum_{j=1}^{n} |p_{ij_{G_t}} - p_{ij_{G_{t+1}}}| = \sum_{j=1}^{n} |\Delta p_{ij}| \ .$$

Enfin, pour étudier l'impact sur une paire de sommets (i, j), on peut se restreindre à l'entrée correspondante de la matrice Δp_{ij}.

3.4.2 Méthodologie

Avant d'étudier de vrais réseaux dynamiques, nous allons simuler une dynamique à partir d'un réseau statique existant. Comme précédemment, cette dynamique doit être aussi simple que possible, contrôlée et garantir que le réseau garde sa structure communautaire malgré l'évolution. Nous utilisons donc la méthode consistant à enlever un seul sommet d'un réseau statique.

Nous avons appliqué cette dynamique simple à un réseau Emails ayant 1133 sommets et 5451 liens (des études sur d'autres réseaux donnent des résultats similaires). Nous avons donc considéré pour chaque sommet u le réseau initial après la suppression de ce sommet. On notera $G_{\setminus u}$ le graphe obtenu en enlevant le sommet u de G.

Afin d'évaluer la stabilité des cœurs et de la comparer avec la stabilité des communautés, nous avons d'abord calculé la matrice des p_{ij} du graphe initial $G(V, E)$, notée $P^{\mathcal{N}}_{ij\ G}$ ainsi que les matrices des graphes $G_{\setminus u}$ notées $P^{\mathcal{N}}_{ij\ G_{\setminus u}}$ avec $\mathcal{N} = 1000$. Ensuite, nous avons calculé les distances euclidiennes entre $P^{\mathcal{N}}_{ij\ G}$ et toutes les matrices $P^{\mathcal{N}}_{ij\ G_{\setminus u}}$. La figure 3.11 illustre les distributions cumulatives de ces distances pour le réseau d'Emails. Pour $\mathcal{N} = 1$ (ce qui correspond simplement à la méthode de Louvain), nous avons exécuté cinq fois l'algorithme afin de nous assurer de ne pas trouver un cas pathologique. Nous observons que les distances euclidiennes pour les cœurs sont beaucoup plus faibles que celles de la structure communautaire.

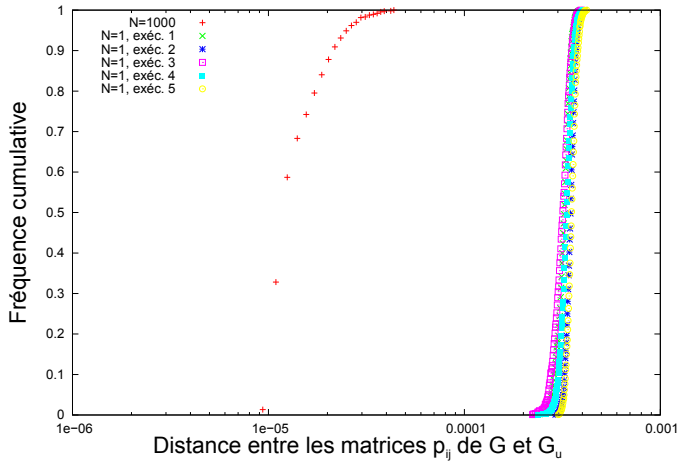
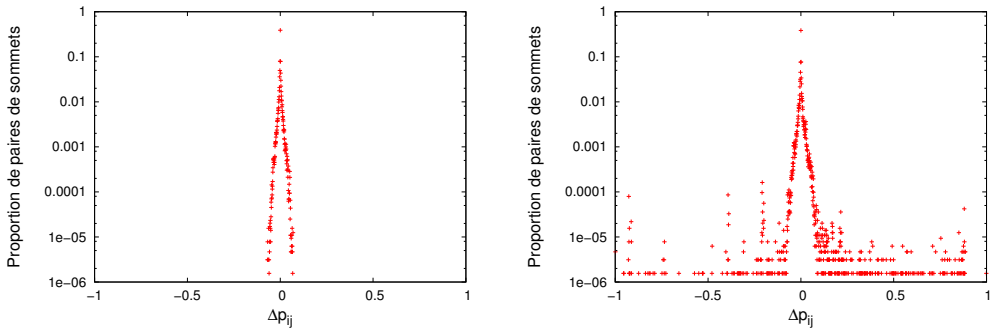


FIGURE 3.11 : Distributions cumulatives des distances euclidiennes entre les matrices avant et après suppression d'un sommet pour les cœurs et les communautés sur le réseau d'Emails.

3.4.3 Impact sur la structure des cœurs

Pour la suppression d'un sommet u, nous avons aussi calculé la matrice des variations $\Delta p_{ij} = p_{ij_G} - p_{ij_{G_{\setminus u}}}$ et étudié la distribution des valeurs dans cette matrice. Nous observons deux types de distributions différents, comme illustré sur la figure 3.12 pour le réseau d'Emails avec une dynamique simulée. La figure 3.12(a) présente la distribution des variation de p_{ij} lors de la suppression du sommet 830. Nous voyons que sur cette figure, les valeurs sont très faibles ce qui signifie que la suppression de ce sommet n'a qu'une faible influence sur la structure des cœurs. En revanche, pour le sommet 731 (figure 3.12(b)), on peut observer des valeurs beaucoup plus élevées, proches de -1 et 1.

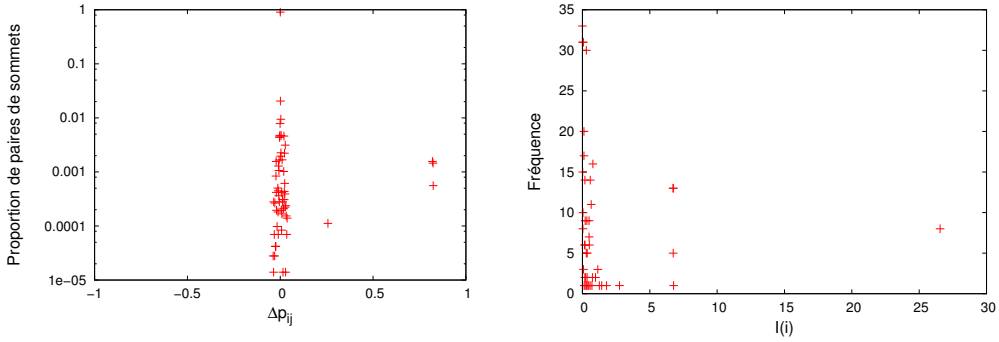
Un Δp_{ij} égal à 1 signifie que i et j n'étaient jamais ensemble avant la suppression et qu'ils sont toujours ensemble après, i et j se sont donc rejoints. Inversement, un Δp_{ij} égal à -1 signifie que le cœur contenant i et j a été scindé.



(a) (b)

FIGURE 3.12 : Distributions des Δp_{ij} pour (a) un sommet ayant un impact faible (distribution proche de 0) et (b) un sommet ayant un impact fort (distribution avec des valeurs proches de 1 et de -1).

Certains cas sont particuliers et ont un impact unidirectionnel. C'est par exemple le cas du sommet 376 du réseau de chercheurs dont les seules variations significatives sont positives (voir figure 3.13(a)). Pour aller plus loin, la figure 3.13(b) montre la distribution des impacts $I(i) = \sum_{j=1}^{n} |\Delta p_{ij}|$ sur les sommets. Nous observons que 8 sommets sont très fortement impactés (avec un impact total de 26) et que 18 sommets sont assez fortement impactés (avec un impact total proche de 7). Ceci correspond simplement à la fusion de deux cœurs, un petit de 8 sommets et un autre de 18 sommets. La dissymétrie dans les impacts vient de la taille des cœurs originaux.

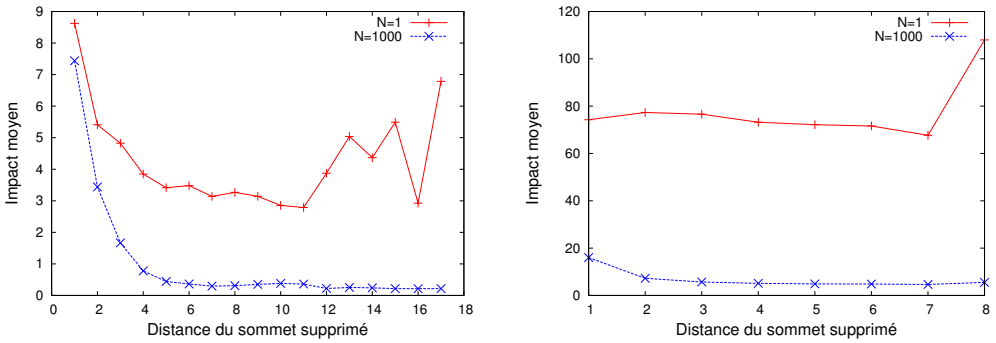


(a) (b)

FIGURE 3.13 : (a) La distribution des Δp_{ij}, lors de la suppression du sommet 376 du réseau de collaboration. (b) La distribution des impacts totaux de cette suppression sur chaque sommet i, $I(i)$.

3.4.4 Localité de l'impact

Nous avons montré que les cœurs sont beaucoup plus stables que les communautés quand le graphe est légèrement modifié, même si la suppression de certains sommets a un impact plus fort. L'un des défauts principaux de la modularité est, qu'outre l'importance des changements, ceux-ci peuvent avoir lieu dans tout le graphe et pas seulement à proximité des modifications, ce qui est peu intuitif. La figure 3.14 illustre l'impact moyen de la suppression d'un sommet sur les autres sommets du graphe en fonction de leur distance au sommet supprimé. Nous observons que cet impact pour les cœurs est faible et très localisé en général et, qu'inversement, il est moins lié à la distance au sommet supprimé pour les communautés.

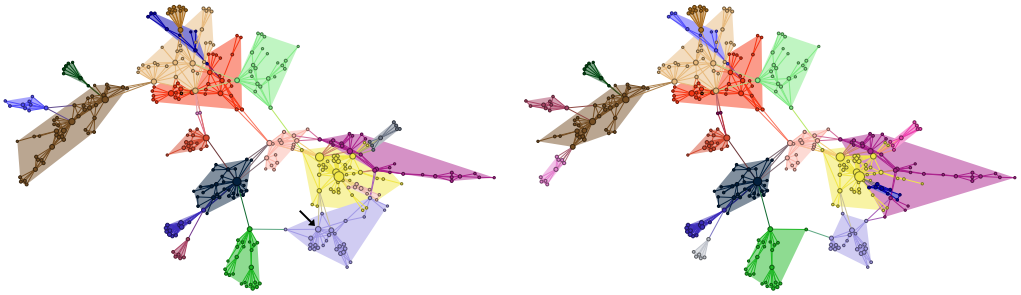


(a) Netsci (b) Email

FIGURE 3.14 : Impact moyen de la suppression d'un sommet sur les sommets à distance k du sommet supprimé en fonction de k.

Malgré tout, nous observons parfois des impacts forts sur des sommets plus éloignés de la modification. Par exemple, comme la figure 3.15 l'illustre, la suppression du sommet 112 du réseau de collaboration (pointé par une flèche) a un impact fort sur des sommets qui en sont éloignés et résulte en la scission d'un cœur en deux sous-cœurs.

Ce résultat est à la fois négatif et aussi très contre-intuitif car les cœurs sont justement censés absorber les variations non locales dûes aux instabilités de l'algorithme. Les quelques cas d'impacts non locaux que nous avons étudiés sont dû à un problème de résolution limite. Ce problème implique que plus le réseau est grand, plus les communautés le sont et donc les cœurs aussi, et inversement. Ainsi la suppression d'un sommet de fort degré peut aboutir à une scission alors que celle d'un sommet de faible degré ne provoquera pas de cassure.



(a) Avant suppression (b) Après suppression

FIGURE 3.15 : Les cœurs identifiés avant et après la suppression du sommet 112 (pointé par une flèche en bas à droite sur (a)). Le problème de résolution limite est visible pour le cœur le plus à gauche de (b).

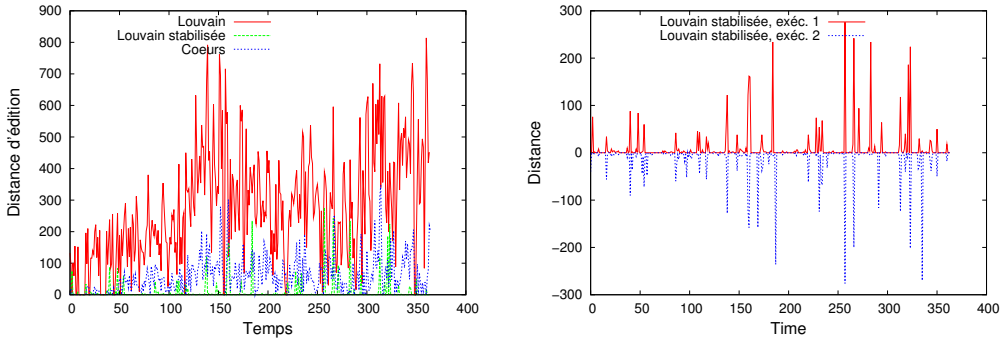
Ce problème de résolution limite est pénalisant dans notre contexte et trouver une méthode

permettant de distinguer les cas de résolution limite des cas d'impact réel est une perspective importante de ce travail.

3.4.5 Dynamique réelle – réseau `mrinfo`

Finalement, nous souhaitons maintenant étudier un cas de dynamique réelle. Outre la compréhension de la dynamique des communautés, l'une des applications majeure est de détecter automatiquement des événements dans les dynamiques de graphes de terrain. Même s'il est difficile de définir clairement ce qu'est un événement, dans notre cas cela consiste à identifier des modifications profondes, occasionnelles et inhabituelles qui ne sont pas conformes au « comportement attendu ».

La figure 3.16(a) illustre la distance d'édition entre les partitions de deux pas de temps successifs pour la méthode de Louvain, la méthode de Louvain stabilisée et les cœurs sur le réseau Mrinfo. Nous observons que les cœurs sont beaucoup plus stables que la méthode de Louvain mais moins stables que la méthode de Louvain stabilisée. Nous avons choisi un seuil de $0,86$ pour les cœurs, qui donne les résultats les plus stables.



(a) (b)

FIGURE 3.16 : (a) Distance d'édition entre communautés ou cœurs pour deux pas de temps successifs sur le réseau `mrinfo`. Les cœurs sont identifiés avec $\mathcal{N} = 1000$ et $\alpha = 0,86$. (b) Distance d'édition entre deux pas de temps successifs pour deux exécutions différentes de Louvain stabilisée sur le réseau `mrinfo`. Les résultats de la deuxième exécution sont présentés avec des valeurs négatives pour simplifier la lecture.

Nous avons cependant des motifs de satisfaction car, bien que la méthode de Louvain stabilisée soit plus stable, le fait qu'elle utilise la partition de l'instant précédent comme initialisation aide l'algorithme mais en même temps lui impose une contrainte. La conséquence de cela, illustrée par la figure 3.16(b), est que si l'on applique la méthode de Louvain stabilisée plusieurs fois au même réseau, certains pics de modifications changent de position à chaque exécution (un changement important a lieu à l'instant 340, mais d'autres moins visibles sont présents à différents endroits). Cela signifie que les événements observés avec cette méthode sont assez dépendants de la toute première partition trouvée et sont donc sujets à caution. Pour les cœurs, ce problème ne peut pas se poser car les calculs sont indépendants à chaque instant.

Il reste beaucoup à faire pour comprendre les communautés sur des graphes dynamiques, mais les cœurs semblent un outil pertinent du fait notamment qu'ils apportent une stabilité naturelle sans contrainte temporelle.

3.5 Synthèse

De nombreux algorithmes ont été proposés pour identifier la structure communautaire des graphes de terrain, mais ils souffrent généralement de non-déterminisme et d'instabilité. Nous avons présenté ici une méthodologie qui tire parti de ce non-déterminisme, et avons proposé une définition du concept de cœurs de communautés, un algorithme simple de calcul associé et une validation de la pertinence de ces cœurs. Des études sur des graphes réels montrent que, sous certaines conditions, les cœurs sont plus similaires à la réalité de terrain que les communautés.

Le calcul des cœurs est, pour l'instant, assez peu efficace puisqu'il nécessite de construire une matrice de probabilité de co-apparition des sommets dans les communautés ce qui est ingérable pour des graphes de grande taille. Une perspective toute naturelle serait d'optimiser algorithmes et structures de données pour ne pas rester limité à des graphes de quelques milliers de sommets.

En outre, nous avons basé le calcul des cœurs sur l'hypothèse que la méthode de Louvain va fournir des partitions de bonne qualité en nombre suffisant pour capturer la majorité des similitudes. Or, il n'est pas impossible que la méthode de Louvain soit fortement biaisé et qu'en fait, les similitudes observées ne soient pas celles des partitions de bonne qualité, mais juste des partitions de bonne qualité obtenues par la méthode de Louvain. Ce point devra être étudié.

L'application de cette méthodologie à des graphes aléatoires a clairement mis en évidence l'absence de structure communautaire réelle dans ces graphes, bien qu'il soit possible d'y trouver des partitions de modularité non triviale. Cette absence est parfaitement visible sous la forme d'une transition de phase pour laquelle un cœur géant est instantanément détruit.

Les perspectives sont nombreuses, la première consistant à prouver formellement l'existence d'une transition de phase dans le cas des graphes aléatoires suivant plusieurs modèles. Il en va de même pour certains types de graphes (arbres, grilles, tores, etc.) sur lesquels on peut trouver des communautés (au sens de la modularité) alors qu'ils n'en contiennent pas réellement.

Dans le cas dynamique, nous avons montré que si de très faibles perturbations du réseau peuvent générer des transformations majeures des communautés, il n'en est pas de même pour les cœurs pour lesquels l'influence des changements est faible et locale, problèmes de résolution limite exceptés. Si les cœurs sont moins stables que certaines approches stabilisées, ils ont l'avantage d'être calculés indépendamment à chaque instant avec une stabilité plus claire.

Le problème de l'étude des communautés dans les graphes dynamiques en est encore à ses balbutiements. Concernant les cœurs, il faudrait dans un premier temps réussir à contourner les problèmes dûs à la résolution limite, afin de pouvoir clairement identifier les évolutions réelles de la structure. Ensuite, il faudrait valider de manière approfondie l'utilité des cœurs sur des graphes réels, par exemple en identifiant des événements connus ou inconnus.

Enfin, la méthodologie des cœurs peut être généralisée et étendue à d'autres approches classiques de détection de communautés. Par exemple, la méthode de Louvain stabilisée est non déterministe et nous avons vu qu'il est sensible à la première partition calculée. Il serait par exemple possible de calculer des cœurs au premier instant puis d'utiliser la méthode de Louvain stabilisée ensuite. Il serait aussi possible de calculer des similarités entre différentes exécutions de la méthode de Louvain stabilisée ou des partitions multi-pas, et d'appliquer les mêmes concepts que pour les cœurs. De manière plus générale, chercher des consensus entre différentes exécutions parait nécessaire dans tous les cas où trouver un optimal est, soit difficile, soit non pertinent.

CONCLUSIONS ET PERSPECTIVES

Même si les premières définitions de communautés dans les graphes datent maintenant d'une soixantaine d'années, la dynamique des graphes de terrain et l'étude des communautés en particulier ne font l'objet d'études intensives que depuis une petite décennie. Il est par contre certain qu'elles vont continuer à être étudiées encore longtemps, les données étant de plus en plus nombreuses et de plus en plus riches. On découvre également en permanence des applications potentielles au fur et à mesure que de nouvelles données sont modélisées sous forme de graphe.

Par exemple, dans le domaine des neurosciences, on est maintenant capable d'étudier le sytème nerveux complet de certaines espèces. Ainsi, un ver microscopique, le *Caenorhabditis elegans* possède 302 neurones et environ 7000 synapses dont on connaît exactement les connexions [1]. Dans un tout autre domaine, on voit de plus en plus d'objets équipés de capacités de communication qui sont connectées à Internet (on parle alors d'Internet des objets). On ne comptera plus alors en millions mais en milliards d'objets connectés, et l'Internet aura certainement une dynamique beaucoup plus forte. Comprendre la dynamique de ces systèmes en général et de leurs communautés en particulier est un défi auquel il faudra répondre.

À une échelle plus modeste, nous avons présenté dans ce mémoire deux axes différents pour aborder le problème de la dynamique des communautés. Le premier axe est basé sur l'idée intuitive que si le graphe évolue peu, alors sa représentation sous forme de communautés devrait de même évoluer assez peu. Cette hypothèse est cependant assez difficile à vérifier en pratique à cause du problème d'instabilité qui bouleverse les partitions même en l'absence de modifications topologiques. Nous avons montré qu'il est malgré tout possible d'obtenir un gain significatif de stabilité en gardant en mémoire des calculs passés et nous avons étendu cette solution à l'extrême en ne faisant plus évoluer les partitions, ce qui a donné lieu a des *communautés multi-pas*. Couplée à une méthode de détection de fenêtres de temps pertinentes, notre approche permet *in fine* de calculer une hiérarchie de plages temporelles et des partitions associées, que nous avons pu confronter avec succès à des données réelles.

Le deuxième axe que nous avons présenté utilise au contraire le non-déterminisme des algorithmes ou plutôt l'existence de nombreuses partitions de qualité similaire afin d'en extraire des similitudes. Nous avons tout d'abord montré que la notion de *cœur* est pertinente et qu'il est possible de les calculer dans les graphes de terrain. Nous avons ensuite montré que, contrairement à ce que l'on observe dans les graphes réels, la structure hiérarchique des cœurs est inexistante dans les graphes aléatoires, bien qu'ils contiennent des partitions de bonne modularité. Les cœurs permettent donc aisément de distinguer les vraies structures communautaires de celles causées par des fluctuations diverses ou des limitations de la modularité. Nous avons également entamé des travaux pour montrer que les cœurs, étant très stables, devraient être utiles pour étudier les graphes dynamiques et que, mis à part un problème de résolution limite, ils arrivent à absorber toutes les variations non locales.

Pour ces deux approches, nous avons déjà présenté un certain nombre de perspectives et, pour les cœurs en particulier, il reste beaucoup à faire dans le contexte dynamique : accéléra-

tion des calculs, validation des résultats sur des données simulées et dans différents contextes réels, etc. La difficulté que nous avons touchée du doigt avec les communautés multi-pas est celle de la validation car nous n'avons pas encore suffisamment de connaissances précises sur la dynamique de réseaux réels qui permettraient de confirmer ou d'infirmer la pertinence de communautés identifiées automatiquement. Les deux approches ont le même objectif mais sont aussi différentes et complémentaires. L'approche multi-pas permet notamment de détecter des plages temporelles pertinentes et est donc un outil plus abouti, alors que les cœurs semblent apporter une solution satisfaisante au problème de stabilité. Je pense qu'un mélange des deux approches, avec notamment des cœurs multi-pas, apportera beaucoup au domaine.

Ces approches, ainsi que la plupart de celles issues de la littérature, sont cependant encore trop imparfaites notamment à cause de leur instabilité et j'ai donc du mal à recommander leur utilisation en l'état. Or, il faut noter que dans un certain nombre de contextes des études utilisent des communautés dynamiques et obtiennent des résultats satisfaisants. C'est notamment le cas pour les exemples que j'ai évoqués dans l'introduction.

En ce sens, j'ai reçu un jour un message d'un chercheur s'étonnant que la méthode de Louvain ne soit pas déterministe et me demandant s'il m'était possible de lui fournir une version qui le soit. J'ai tenté de raisonner l'utilisateur en arguant du fait que les partitions se valent du point de vue modularité et qu'en privilégier une n'aurait pas vraiment de sens. Je lui ai finalement fourni une version déterministe et, si je ne sais pas ce qu'il a fait de la partition, j'imagine qu'il y a trouvé son compte. Je pense que j'avais raison de tenter de le convaincre qu'une version déterministe n'apporte rien mais, dans le même temps, les utilisateurs confrontés à des données réelles ont besoin d'un résultat à étudier. Pourquoi pas un plutôt qu'un autre ? Les études sur les cœurs ont montré qu'il y a de fortes similitudes entre les partitions et il est possible que toute partition contient un fond de vérité que l'utilisateur trouvera et qui confortera ses hypothèses. Nous avions sans doute tous les deux raison, mais il était plus pragmatique que moi.

Ce décalage entre les chercheurs qui conçoivent des algorithmes et ceux qui les utilisent est assez important, les premiers étant généralement critiques sur leurs méthodes et les second utilisant les algorithmes comme des boîtes noires. On pourrait dire que ces deux catégories d'utilisateurs sont dans des « communautés » disjointes avec peu de connexions entre elles. La question se pose alors de savoir s'il existe des recouvrements, c'est-à-dire des chercheurs qui sont à la fois concepteurs et utilisateurs d'algorithmes.

Il faut aussi noter que l'étude classique des graphes de terrain se restreint à l'étude des connexions et n'exploite pas, ou peu, les informations spécifiques aux individus. Au contraire, dans le domaine de la fouille de données, l'approche consiste à se concentrer beaucoup plus sur les individus mais en négligeant ou en simplifiant leurs relations sociales. La bonne solution est sans doute un mélange des deux et, vu sous l'angle des communautés, cela revient à ajouter des informations sur les sommets et à les prendre en compte pour calculer les communautés, comme par exemple dans [47, 67]. Même si nous nous sommes pour l'instant focalisés sur la dynamique, c'est clairement un axe à étudier de manière plus approfondie et qui pourra à la fois apporter et apprendre des méthodes dynamiques.

Retour sur l'instabilité

Revenons à présent sur le problème fondamental de l'instabilité. J'ai volontairement abordé ce problème de manière succincte dans le chapitre 1 car je ne vois pas pour l'instant comment le fait de le comprendre permet de le résoudre.

Pour mieux comprendre ce problème d'instabilité, il faut se rappeler que le nombre de partitions d'un ensemble à n éléments se compte avec les nombres de Bell qui ont tendance à

croître très rapidement. Ainsi, le nombre de partitions du club de karaté de Zachary, que nous avons utilisé à plusieurs reprises dans ce mémoire, est de l'ordre de 2.10^{28} alors qu'il ne contient que 34 sommets. De même, le nombre de partitions d'un graphe à un million de sommets s'écrit avec un peu plus de 4 millions de chiffres [3]. Pour relativiser, il faut tout de même noter que seules les partitions connexes sont réellement pertinentes d'après une propriété que nous avons énoncée plus tôt et que, même parmi celles-ci, les partitions connexes de qualité raisonnable sont extrêmement rares. Par exemple, sur un graphe à 16 sommets que j'utilise souvent pour expliquer la méthode de Louvain, il y a un peu plus de 10 milliards de partitions, mais seules 44484 sont connexes [4].

Parmi toutes ces partitions, il y en a malgré tout une (ou quelques-unes) dont la qualité est supérieure à celle de toutes les autres, c'est-à-dire qui est optimale au sens de la fonction de qualité choisie. Cependant, il y a fort à parier que, quelle que soit la fonction choisie (dès lors qu'elle est raisonnable), il y aura un grand nombre de partitions de qualité très proche de celle des partitions optimales. En conséquence, toute modification, même mineure, du graphe fera sans doute passer la partition optimale au second plan et la nouvelle partition optimale sera probablement radicalement différente de l'ancienne, d'où une forte instabilité.

La conclusion que je tire des remarques précédentes est que si la recherche de la partition optimale au sens de la modularité est un problème intéressant d'un point de vue théorique, je suis convaincu qu'il n'aidera pas réellement à comprendre la structure communautaire d'un graphe de terrain donné et encore moins à faire du suivi de communautés dynamiques. En ce sens, je pense que la course à la modularité et à la rapidité n'est plus l'enjeu majeur du domaine, même s'il a pu l'être à une époque où les algorithmes étaient lents et peu efficaces.

On peut sans doute aussi penser que la modularité n'est pas la bonne fonction et on aurait certainement raison au vu de ses nombreux défauts et des critiques formulées à son égard depuis son introduction en 2004. Malgré tout, il y a énormément d'applications en biologie, en neuroscience, en bibliométrie, en informatique, en physique, et dans d'autres domaines qui utilisent la modularité et les algorithmes d'optimisation associés de manière satisfaisante (suffisament pour les experts de ces domaines) ce qui permet de justifier l'intérêt qui lui est porté. Ainsi, malgré le théorème de Kleinberg, malgré les problèmes de résolution limite et d'instabilité, la modularité est à ce jour une réponse correcte au problème de l'évaluation de la qualité d'un partitionnement.

Je pense de plus, sans avoir d'arguments formels pour soutenir cette thèse, que le problème d'instabilité est indépendant de la fonction de qualité utilisée. Il y a donc certainement des fonctions de qualité « plus satisfaisantes » (selon des critères à préciser) que la modularité et je suis de près des travaux qui tentent d'en proposer de nouvelles mais le nombre de partitions distinctes posera problème dans tous les cas. Je ne pense pas que celle qui remplacera la modularité ait été publiée et je pense qu'elle ne pourra s'imposer que si elle est associée directement à un algorithme d'optimisation très performant, par exemple avec une méthode « à la Louvain ».

Si trouver l'optimal de modularité ne permettra pas de résoudre les problèmes de stabilité et si changer de fonction de qualité ne changera rien non plus, on peut tenter de changer le problème et les idées que je vais développer dans la suite vont dans ce sens. Le fait de vouloir

3. Il est intéressant de voir que l'on a calculé les cœurs avec quelques centaines ou quelques milliers de partitions et que l'on observait déjà une convergence forte des matrices p_{ij} alors que nous n'avions fait qu'effleurer la variété de partitions possibles. Cela est peut-être dû, comme nous en avons déjà discuté, au fait que la méthode de Louvain pourrait n'explorer qu'une portion infime de ces partitions.

4. Je tiens à remercier Clémence Magnien qui, à ses heures perdues, a écrit quelques lignes de code pour calculer efficacement toutes les partitions connexes d'un graphe quelconque. Lorsqu'elle a tenté de le lancer sur le graphe du club de karaté de Zachary, le programme s'est interrompu après un mois de calculs à cause d'une panne de courant. Le programme avait déjà généré plusieurs milliards de partitions connexes !

partitionner notamment est ce qui, au fond, paraît à la fois le plus contraignant et le moins intuitif. Prenons l'exemple d'un réseau social et tentons d'en trouver les communautés. Le fait de vouloir partitionner impose que chaque individu soit dans une et une seule communauté : il faudra ou bien le mettre avec sa famille, ou bien le mettre avec ses collègues, à moins de regrouper tout ce beau monde (en n'oubliant pas les membres de son club de sport, ses followers sur twitter et ses « amis » facebook). À vouloir partitionner, on va nécessairement scinder le réseau à des endroits arbitraires alors qu'il serait plus simple d'autoriser les individus à appartenir à plusieurs communautés, ce qui implique que les communautés puissent se recouvrir.

La perspective de communautés recouvrantes

La simplicité n'est bien entendu qu'apparente. Comme pour la dynamique, si l'on a commencé par ignorer l'aspect recouvrant, c'est parce que le problème est beaucoup plus complexe. D'un point de vue combinatoire en particulier, le nombre de communautés recouvrantes est élevé : il y a 2^n communautés distinctes possibles dans un graphe à n sommets, et donc 2^{2^n} découpages d'un graphe en communautés recouvrantes. Les approches actuelles s'orientent dans deux directions.

La première direction consiste à étudier un moyen terme entre partitionnement et communautés recouvrantes, en cherchant notamment des ensembles de sommets très marqués communautairement et des zones qui sont plus intercommunautaires. Dans [91] par exemple, une notion similaire aux cœurs est utilisée : des groupes de sommets qui sont ensemble à chaque exécution (des 1-cœurs avec notre terminologie) sont clairement des communautés. Au contraire, si deux sommets sont classés ensemble une fois sur deux, alors le lien qui les unit est moins fort. Les auteurs parlent alors de communautés floues et de zones recouvrantes.

La seconde direction consiste à cesser d'étudier le graphe dans son intégralité et à repasser au niveau de l'individu. Plutôt que de vouloir trouver toutes les communautés du graphe, on s'intéresse donc plutôt aux communautés auxquelles un individu appartient. Ces communautés sont certainement en très petit nombre et de taille plus raisonnable. Du fait qu'elles sont centrées sur un individu, on les appelle communautés ego-centrées ou ego-munautés [44].

Je m'intéresse à ce thème notamment dans le cadre de la thèse de Maximilien Danisch [31]. À partir d'une nouvelle mesure de similarité, il est possible d'identifier les sommets qui sont dans le voisinage au sens large (pas uniquement les voisins directs) d'un individu donné. Nous utilisons ensuite cette similarité pour chercher des communautés multi-ego-centrées, c'est-à-dire des communautés définies par le voisinage de plusieurs individus. Par exemple, si je considère mon réseau social, je fais certainement partie de plusieurs communautés (famille, amis, collègues et j'en passe), mais si je fais maintenant de même pour une autre personne de l'équipe *Complex Networks*, il est alors probable que l'intersection de nos deux communautés égo-centrées permette de reconstituer l'équipe *Complex Networks*, puisque la prise en compte d'un autre point de vue permet de mieux cibler le contexte.

Les résultats actuels valident le concept de communautés multi-ego-centrées et montrent qu'il est effectivement possible à la fois de classer les individus d'un réseau en fonction de leur similarité à un individu donné, et de trouver des individus dans le voisinage de ce dernier qui permettent d'identifier des communautés multi-ego-centrées. Dans les exemples que nous avons étudiés, chaque communauté autour d'un individu peut être identifié grâce à un seul de ses voisins (au sens large à nouveau). On peut donc se limiter à des communautés bi-centrées, sans avoir besoin d'étudier des communautés définies par 3 individus ou plus. Avant de pouvoir attaquer les aspects dynamiques avec cette méthode, il faut encore effectuer de nombreux travaux de formalisation et de validation, mais il faudra à terme étudier la dynamique sous cet angle.

Des travaux très préliminaires sur ce thème et non encore publiés montrent par ailleurs que cette technique permettra sans doute d'aborder des problèmes de classification dans les graphes : étant donné un graphe et un étiquetage partiel des sommets, peut-on inférer les étiquettes des autres sommets ? Les résultats sont très encourageants.

Le mot de la fin

Le domaine de la détection de communautés dynamiques fait donc son chemin, en parallèle de l'étude des graphes de terrain dynamiques, chacune s'enrichissant des avancées de l'autre. Obtenir des données, les comprendre et les modéliser permet de mieux définir des algorithmes de détection de communautés et, inversement, la compréhension de la dynamique des communautés est une brique de l'ensemble qui apporte un éclairage différent. Les études de ces dernières années dans le domaine des communautés dynamiques, et les nôtres en particulier, n'ont pas encore abouti à des propositions qui fassent consensus. La détection de communautés dynamiques est donc toujours un problème largement ouvert du domaine et, en ce sens, reste une perspective majeure de mes travaux. Je continuerai donc à m'investir dans ce domaine.

Comme je le disais en introduction, mes travaux ne se limitent pas à l'étude des communautés statiques ou dynamiques, même si elle occupe une part significative de mon temps. Les graphes de terrain sont le dénominateur commun à tous mes travaux et principalement les aspects liés à leur dynamique. En particulier, je travaille aussi sur l'analyse et la modélisation de données dynamiques [23, 85], et plus récemment sur l'étude de l'impact de la dynamique sur les phénomènes de diffusion sur lequel je travaille avec Alice Albano [3] ainsi que sur d'autres problèmes algorithmiques que la détection de communautés, comme le calcul de plus courts chemins temporels [93]. L'étude de la dynamique sous d'autres angles permet de mieux comprendre la dynamique des communautés et inversement, je continuerai donc aussi à étudier d'autres problèmes liés aux graphes de terrain et à leur dynamique.

BIBLIOGRAPHIE

[1] T.B. Achacoso and W.S. Yamamoto. *AY's Neuroanatomy of C. elegans for Computation*. CRC Press, 1992.

[2] R.D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3 :113–126, 1973.

[3] A. Albano, J.-L. Guillaume, and B. Le Grand. File diffusion in a dynamic peer-to-peer network. In *Proceedings of the Mining Social Network Dynamic 2012 Workshop (MSND), In conjunction with the international conference World Wide Web*, pages 1169–1172, 2012.

[4] R. Albert and A.-L. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74 :47–97, 2002.

[5] R. Albert, H. Jeong, and A.-L. Barabasi. The diameter of the world wide web. *Nature*, 401 :130–131, 1999.

[6] A. Arenas, A. Díaz-Guilera, and C.J. Pérez-Vicente. Synchronization reveals topological scales in complex networks. *Phys. Rev. Lett.*, 96 :114102, 2006.

[7] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 921. ACM, 2007.

[8] T. Aynaud. *Détection de communautés dans les graphes dynamiques*. PhD thesis, Université Pierre et Marie Curie, 2011.

[9] T. Aynaud, V.D. Blondel, J.-L. Guillaume, and R. Lambiotte. Optimisation locale multi-niveaux de la modularité. In C.-E. Bichot and P. Siarry, editors, *Partitionnement de graphe : optimisation et applications*. Hermes-Lavoisier, traité ic2 edition, 2011.

[10] T. Aynaud and J.-L. Guillaume. Long range community detection. In *Latin American Workshop on Dynamic Networks*, 2010.

[11] T. Aynaud and J.-L. Guillaume. Static community detection algorithms for evolving networks. In *Wireless Networks*, pages 508–514, 2010.

[12] T. Aynaud and J.-L. Guillaume. Multi-Step Community Detection and Hierarchical Time Segmentation in Evolving Networks. In *SNAKDD*, volume 11, 2011.

[13] T. Aynaud, J.-L. Guillaume, E. Fleury, and Q. Wang. Communities in evolving networks : definitions, detection and analysis techniques. In N. Ganguly, A. Mukherjee, M. Choudhury, F. Peruani, and B. Mitra, editors, *Dynamics of Time Varying Networks*. Birkhauser, Springer, to appear.

[14] D.S. Bassetta, N.F. Wymbs, M.A. Porter, P.J. Mucha, J.M. Carlson, and S.T. Graftonb. Dynamic reconfiguration of human brain networks during learning. *PNAS*, 108 :18 :7641–7646, 2011.

[15] M.G. Beiró and J.R. Busch. Visualizing communities in dynamic networks. In *Latin American Workshop on Dynamic Networks*, volume 1, 2010.

[16] E.A. Bender and E.R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3) :296–307, 1978.

[17] I. Bifulco, C. Fedullo, F. Napolitano, G. Raiconi, and R. Tagliaferri. Robust clustering by aggregation and intersection methods. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 732–739. Springer, 2008.

[18] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech*, 10008 :1–12, 2008.

[19] V.D. Blondel and P. Senellart. Automatic extraction of synonyms in a dictionary. In *Proceedings of the SIAM Text Mining Workshop*, pages 7–13, 2002.

[20] P. Boldi, B. Codenotti, M. Santini, and S. Vigna. Ubicrawler : A scalable fully distributed web crawler. *Software : Practice & Experience*, 34 :711–726, 2004.

[21] P. Boldi and S. Vigna. The WebGraph framework I : Compression techniques. In *Proc. of the Thirteenth International World Wide Web Conference (WWW 2004)*, pages 595–601, Manhattan, USA, 2004. ACM Press.

[22] S.P. Borgatti, M.G. Everett, and P.R. Shirey. Ls sets, lambda sets, and other cohesive subsets. *Social Networks*, 12 :337–358, 1990.

[23] P. Borgnat, E. Fleury, J.-L. Guillaume, C. Robardet, and A. Scherrer. Description and simulation of dynamic mobility networks. *Computer Networks*, 52 :2842–2858, 2008.

[24] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing Modularity is hard. *ArXiv Physics e-prints*, August 2006.

[25] J. Candia, M.C. González, P. Wang, T. Schoenharl, G. Madey, and A.-L. Barabási. Uncovering individual and collective human dynamics from mobile phone records. *Journal of Physics A : Mathematical and Theoretical*, 41(22) :224015, 2008.

[26] S.-Y. Chan, P. Hui, and K. Xu. Community Detection of Time-Varying Mobile Social Networks. *Complex Sciences*, pages 1154–1159, 2009.

[27] Z. Chen, K.A. Wilson, Y. Jin, W. Hendrix, and N.F. Samatova. Detecting and Tracking Community Dynamics in Evolutionary Networks. *2010 IEEE International Conference on Data Mining Workshops*, pages 318–327, December 2010.

[28] A. Clauset, C. Moore, and M.E.J. Newman. Finding community structure in very large networks. *Physical Review E*, 70 :066111, 2004.

[29] V. Colizza, A. Barrat, M. Barthélemy, and A. Vespignani. The role of the airline transportation network in the prediction and predictability of global epidemics. *Proc. Natl. Acad. Sci.*, 103 :7 :2015–2020, 2006.

[30] Cornell KDD cup datasets, 2003. http ://www.cs.cornell.edu/projects/kddcup/datasets.html.

[31] M. Danisch, J.-L. Guillaume, and B. Le Grand. Towards multi-ego-centered communities : a node similarity approach. *Int. J. of Web Based Communities*, to appear.

[32] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics : Theory and Experiment*, page P09008, 2005.

[33] F. de Montgolfier, M. Soto, and L. Viennot. Asymptotic modularity of some graph classes. In *22nd International Symposium on Algorithms and Computation (ISAAC)*, pages 435–444, 2011.

[34] E. Diday. Une nouvelle méthode en classification automatique et reconnaissance des formes la méthode des nuées dynamiques. *Revue de Statistique Appliquée*, 19(2) :19–33, 1971.

[35] P. Erdös and A. Rényi. On random graphs, i. *Publicationes Mathematicae (Debrecen)*, 6 :290–297, 1959.

[36] M.G. Everett and S.P. Borgatti. Peripheries of cohesive subsets. *Social Networks*, 21 :397–6407, 1999.

[37] P. Tsigas F. Moradi, T. Olovsson. An evaluation of community detection algorithms on large-scale email traffic. In *In Proc. 11th Intl. Symposium on Experimental Algorithms*, pages 283–294, 2012.

[38] T. Falkowski, M. Spiliopoulou, and J. Bartelheimer. Community dynamics mining. In *Proceedings of 14th European Conference on Information Systems (ECIS 2006)*. Citeseer, 2006.

[39] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proceedings of SIGCOMM '99*, pages 251–262, 1999.

[40] D.J. Fenn, M.A. Porter, P.J. Mucha, M. McDonald, S. Williams, N.F. Johnson, and N.S. Jones. Dynamical clustering of exchange rates. *Quantitative Finance*, 0 :1–28, 2012.

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

[42] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104 :36–41, 2007.

[43] A.L.N. Fred and A.K. Jain. Robust data clustering. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, page 128, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[44] A. Friggeri, G. Chelius, and E. Fleury. Egomunities, exploring socially cohesive person-based communities. In *The International School and Conference on Network Science*, 2011.

[45] D. Gfeller, J.-C. Chappelier, and P. De Los Rios. Finding instabilities in the community structure of complex networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 72(5 Pt 2) :056135, November 2005.

[46] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12) :7821, 2002.

[47] J.D. Cruz Gomez, C. Bothorel, and F. Poulet. Semantic clustering of social networks using points of view. In *CORIA : conférence en recherche d'information et application*, 2011.

[48] S. Grauwin, G. Beslon, E. Fleury, S. Franceschelli, C. Robardet, J.-B. Rouquier, and P. Jensen. Complex Systems Science : Dreams of Universality, Reality of Interdisciplinarity. *Journal of the American Society for Information Science and Technology*, 2012.

[49] D. Greene and D. Doyle. Tracking the evolution of communities in dynamic social networks. In *Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1–13. IEEE, 2010.

[50] R. Guimerà, M. Sales-Pardo, and L.A.N. Amaral. Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E*, 70 :025101, 2004.

[51] M. Hoerdt and D. Magoni. Completeness of the internet core topology collected by a fast mapping software. In *11th IEEE International Conference on Software, Telecommunications and Computer Networks (softcom)*, Split (Croatie), october 2003.

[52] J.E. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *National Academy of Sciences of the United States of America*, 101(Suppl 1) :5249, 2004.

[53] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter : understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65, 2007.

[54] B. Karrer, E. Levina, and M.E.J. Newman. Robustness of community structure in networks. *Phys. Rev. E*, 77 :046119, 2008.

[55] M.S. Kim and J. Han. A particle-and-density based evolutionary clustering method for dynamic networks. *Proceedings of the VLDB Endowment*, 2(1) :622–633, 2009.

[56] J.M. Kleinberg. An impossibility theorem for clustering. In *NIPS*, pages 446–453, 2002.

[57] H.W. Kuhn, P.J. Haas, I.F. Ilyas, G.M. Lohman, and V. Markl. The Hungarian method for the assignment problem. *Masthead*, 23(3) :151210, 1993.

[58] R. Kumar, A. Tomkins, and D. Chakrabarti. Evolutionary clustering. In *Proc. of the 12th ACM SIGKDD Conference*, 2006.

[59] R. Lambiotte. Multi-scale modularity in complex networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 546–553. IEEE, April 2010.

[60] R. Lambiotte, V.D. Blondel, C. de Kerchove, E. Huens, C. Prieur, Z. Smoreda, and P. Van Dooren. Geographical dispersal of mobile communication networks. *Physica A*, 387 :5317–5325, 2008.

[61] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, 78 :046110, 2008.

[62] A.C.F. Lewis, N.S. Jones, M.A. Porter, and C.M. Deane. The function of communities in protein interaction networks at multiple scales. *BMC Systems Biology*, 4 :100, 2010.

[63] F. Luccio and M. Sami. On the decomposition of networks in minimally interconnected subnetworks. *IEEE Transactions on Circuit Theory*, 16 :2 :184–188, 1969.

[64] R.D. Luce and A.D. Perry. A method of matrix analysis of a group structure. *Psychometrika*, 14 :95–116, 1949.

[65] S. Mancoridis, B.S. Mitchell, and C. Rorres. Using automatic clustering to produce high-level system organizations of source code. In *In Proc. 6th Intl. Workshop on Program Comprehension*, pages 45–53, 1998.

[66] R. Mokken. Cliques, clubs and clans. quality and quantity. *European journal of methodology*, 13 :161–173, 1979.

[67] J. Neville, M. Adler, and D. Jensen. Clustering relational data using attribute and link information. In *Proceedings of the Text Mining and Link Analysis Workshop*, 2003.

[68] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69 :026113, 2004.

[69] M. Oliveira and J. Gama. Bipartite graphs for monitoring clusters transitions. *Advances in Intelligent Data Analysis IX*, M :114–124, 2010.

[70] M. Oliveira and J. Gama. Understanding Clusters Evolution. In *Workshop on Ubiquitous Data Mining*, volume D, pages 16 – 20, 2010.

[71] G. Palla, A.-L.Barabasi, and T. Vicsek. Quantifying social group evolution. *Nature*, 446 :664–667, 2007.

[72] D. Pfitzner, R. Leibbrandt, and D. Powers. Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems*, 19(3) :361–394, 2009.

[73] P. Pons and M. Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10 :191–218, 2006.

[74] I. Rafols and L. Leydesdorff. Content-based and algorithmic classifications of journals : Perspectives on the dynamics of scientific communication and indexer effects. *Journal of the American Society for Information Science and Technology*, 60 :1823–1835, 2009.

[75] J. Reichardt and S. Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Phys Rev Lett*, 93 :218701, 2004.

[76] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1) :016110, 2006.

[77] M. Ripeanu, A. Iamnitchi, and I. Foster. Mapping the gnutella network. *IEEE Internet Computing*, 6 :50–57, 2002.

[78] M. Rosvall. Mapping change in large networks. *PLoS One*, pages 1–9, 2010.

[79] S.B. Seidman. Internal cohesion of ls sets in graphs. *Social Networks*, 5 :97–107, 1983.

[80] M. Seifi. *Coeurs stables de communautés dans les graphes de terrain.* PhD thesis, Université Pierre et Marie Curie, 2012.

[81] M. Seifi and J.-L. Guillaume. Community Cores in Evolving Networks. In *Proceedings of the Mining Social Network Dynamic 2012 Workshop (MSND), In conjunction with the international conference World Wide Web WWW*, pages 1173–1180, 2012.

[82] M. Seifi, J.-L. Guillaume, I. Junier, J.B. Rouquier, and S. Iskrov. Stable community cores in complex networks. In *Proceedings of the 3rd Workshop on Complex Networks (CompleNet 2012)*, 2012.

[83] X. Song, Y. Chi, B.L. Tseng, D. Zhou, and K. Hino. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 153–162. ACM New York, NY, USA, 2007.

[84] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. Monic : modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 706–711. ACM New York, NY, USA, 2006.

[85] A. Stoica, Z. Smoreda, C. Prieur, and J.-L. Guillaume. Age, gender and communication networks. In *Proceedings of the Workshop on the Analysis of Mobile Phone Networks, satellite workshop to NetSci*, 2010.

[86] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3 :583–617, 2003.

[87] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the facebook social graph. *arXiv :1111.4503*, pages 1–1, 2011.

[88] K. Wakita and T. Tsurumi. Finding community structure in mega-scale social networks : [extended abstract]. In *WWW '07 : Proceedings of the 16th international conference on World Wide Web*, pages 1275–1276. ACM, 2007.

[89] M.L. Wallace, Y. Gingras, and R. Duhon. A new approach for detecting scientific specialties from raw cocitation networks. *Journal of the American Society for Information Science and Technology*, 60 :240–246, 2009.

[90] Q. Wang and E. Fleury. Mining time-dependent communities. In *Latin American Workshop on Dynamic Networks*, 2010.

[91] Q. Wang and E. Fleury. Community detection with fuzzy community structure. In *ASONAM 2011*, 2011.

[92] Y. Wang, B. Wu, and N. Du. Community Evolution of Social Network : Feature, Algorithm and Model. *Science And Technology*, 60402011, 2008.

[93] J. Whitbeck, M. Dias de Amorim, V. Conan, and J.-L. Guillaume. Temporal reachability graphs. In *The 18th Annual International Conference on Mobile Computing and Networking, Mobicom'12*, pages 377–388, 2012.

[94] K. Xu, M. Kliger, and A. Hero. Tracking Communities in Dynamic Social Networks. *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 219–226, 2011.

[95] W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33 :452–473, 1977.

[96] C.T. Zahn. Approximating symmetric relations by equivalence relations. *SIAM Journal on Applied Mathematics*, 12 :840–847, 1964.

Résumé

De nombreux systèmes, tels que des réseaux sociaux ou des réseaux informatiques, peuvent être modélisés par des graphes, que l'on appelle alors graphes de terrain. Un certain nombre de travaux ont montré que ces graphes, bien que différents par bien des aspects, sont aussi semblables par beaucoup d'autres et notamment ils possèdent tous une structure communautaire assez forte, c'est-à-dire qu'ils sont formés de sous-ensembles de sommets densément connectés. Si l'on se restreint à une partition en communautés, on dispose de méthodes efficaces pour calculer cette structure, notamment la méthode de Louvain que j'ai contribué à créer et qui est la plus efficace dans le domaine. Or, la plupart de ces réseaux réels sont dynamiques et évoluent au cours du temps par l'ajout ou la suppression de sommets et de liens. Cette dynamique touche naturellement les communautés et il faut donc proposer de nouvelles méthodes pour les calculer et les analyser.

Nous nous sommes intéressé dans ce mémoire à l'approche naturelle qui consiste à considérer un graphe dynamique comme une succession de graphes statiques, puis à calculer une partition en communautés à chaque instant et, enfin, à essayer de faire le lien entre les communautés à différents instants. Nous avons montré que cette approche n'est pas utilisable directement car une modification mineure de la topologie peut engendrer des modifications très importantes de la structure communautaire, d'où un phénomène d'instabilité. Nous avons alors proposé deux approches pour tenter de résoudre ce problème.

La première approche considère que si le graphe évolue peu, ses communautés devraient rester globalement stables. Nous avons donc tout d'abord tenté de stabiliser un algorithme existant en gardant la mémoire des calculs passés, ce qui a donné des résultats bien meilleurs mais avec toujours une instabilité résiduelle. Puis, nous avons étendu cette solution en calculant des partitions multi-pas de bonne qualité sur plusieurs instants de temps. Nous avons couplé cela avec une méthode de décomposition hiérarchique du temps afin de calculer des plages temporelles sur lesquelles ces partitions multi-pas ont un sens. Cette méthode à été appliquée avec succès à des données réelles.

La seconde approche considère que même s'il y a de nombreuses partitions de qualité, elles ne sont pas complètement différentes. Nous avons donc proposé une méthode pour calculer en pratique ces similitudes, qui permettent de définir des cœurs de communautés. Nous avons montré que les cœurs sont pertinents dans les graphes de terrain et permettent de les distinguer des graphes sans réelle structure communautaire (comme les graphes aléatoires par exemple). Nous avons également entamé des travaux pour montrer que les cœurs peuvent être utilisés dans le cas dynamique et qu'ils sont naturellement stables et que les modifications qu'ils peuvent subir sont cette fois très corrélées aux modifications topologiques.

Summary

Many real-world complex systems, such as social networks or computer networks can be modeled as graphs, called complex networks. Many studies have shown that these networks are very similar in many regards. In particular they all possess a very strong community structure, i.e., they are composed of dense subgraphs weakly interconnected. Restricting the definition of communities to partitions of the set of nodes has given rise to many efficient algorithms to unfold this structure, in particular the Louvain method which is currently the most efficient one. Since most complex networks are evolving, through the appearance and disappearance of nodes and links, the community structure is also evolving and new techniques are needed for computing and analyzing such evolving communities.

We study here the natural approach which consists in considering an evolving graph as a series of static graphs. Then we can compute communities at each time step independently and compute the correspondances between different time steps. We show that this approach is not directly useful since minor modifications of the topology can result in massive modification of the community structure, and henceforth we have a stability issue. We then propose two solutions to overcome this.

The first one considers that if a network undergoes a small evolution then the communities should not change much. Therefore, we stabilize an existing algorithm by taking previous computation in consideration. This gives better results, but there is a residual instability. We then extend this solution with the computation of multi-steps communities which have a good quality for several time steps. We finally add a hierarchical clustering of time so as to identify time windows on which such multi-steps communities are relevant. This approach is validated on real data.

The second approach is based on the idea that, even if there are many partitions of high quality, they are not so different. We propose a solution to compute these similarities, and use them to define community cores. We show that such cores make sense in real networks and also allow to distinguish real complex networks from graphs without a real community structure, such as random graphs. We finally start a study the evolution of cores which are naturally stable and whose modifications are strongly related to real topology modifications.