Détection de communautés orientée sommet pour des réseaux mobiles opportunistes sociaux

Maël Canu




Détection de communautés orientée sommet pour des réseaux mobiles opportunistes sociaux

THÈSE

présentée et soutenue publiquement le 20 décembre 2017

pour l'obtention du

Doctorat de l'Université Pierre et Marie Curie (Sorbonne Université(s))

(mention informatique)

par

Maël Canu

Composition du jury

Présidente :	Clémence Magnien	Directrice de Recherche LIP6, Université Pierre et Marie Curie, Paris
Rapporteurs :	Jean-Loup Guillaume	Professeur des Universités L3i, Université de La Rochelle
	Nicolas Labroche	Maître de Conférences (HDR) LI, Université François Rabelais, Tours
Examinateurs :	Marcin Detyniecki (invité)	Head of Research Data Innovation Lab, AXA (Suresnes)
	Vincent Labatut (invité)	Maître de Conférences LIA, Université d'Avignon et des Pays du Vaucluse
	Anne Laurent	Professeure des Universités LIRMM, Université de Montpellier
Encadrants :	Marie-Jeanne Lesot	Maître de Conférences (HDR) LIP6, Université Pierre et Marie Curie, Paris
	Adrien Revault d'Allonnes	Maître de Conférences LIASD, Université Paris 8, Saint-Denis

Laboratoire d'Informatique de Paris 6 (LIP6, UMR 7606)
École Doctorale Informatique, Télécommunications et Électronique (EDITE, ED130)

Mis en page avec la classe thesul.

Remerciements

Il est difficile d'établir précisément les remerciements que l'on souhaite formuler à la fin d'un travail de thèse tant les personnes à remercier sont nombreuses. De manière générale, à moins d'aimer travailler seul dans une grotte, éventuellement enchaîné à son bureau [1], il paraît indispensable de bénéficier d'un soutien moral sous une forme ou une autre, et quelques mots sur une page numérotée en chiffres romains sont finalement bien peu par rapport au service rendu.

Je vais néanmoins tenter d'être relativement exhaustif, en limitant les noms de famille à une seule lettre, ce qui a d'une part l'avantage ludique de permettre à chacun(e) de chercher où il/elle apparaît, et d'autre part de ne pas apparaître trop ostensiblement au détour d'une simple recherche Google. Commençons.

Pour avoir accepté de relire soigneusement mon manuscrit et d'avoir écrit un rapport dessus, je commence par Jean-Loup G. et Nicolas L., avec une pensée émue pour ce dernier puisque je l'aurai retrouvé à toutes les étapes de mon parcours dans le supérieur, de mon chargé de TD LI101 peu après la sortie du lycée, à mon master en Intelligence Artificielle et Décision, puis à mon équipe de thèse et au final à mon rapport et jury de thèse.

Également aux autres membres de mon jury : Anne L. et Vincent L. que je remercie d'avoir fait le déplacement depuis le sud un mercredi de décembre pour venir m'écouter parler. « Big up » à Vincent en particulier avec qui j'ai eu plaisir à échanger lors des conférences MARAMI et à qui je souhaite un beau parcours dans le monde académique français. Je remercie bien évidemment Clémence M. d'avoir présidé ce jury, ainsi que plus généralement d'animer la recherche en réseaux complexes à l'UPMC à travers l'équipe ComplexNetworks. Je pense plus précisément aux séminaires fréquents et toujours très intéressants qui m'ont permis de solidifier mes connaissances dans la discipline. Au passage, je souhaite également remercier son collègue Matthieu L., et plus personnellement Fabien T. et Maximilien D. qui en plus d'être très sympas sont très talentueux. En particulier je souhaite un beau démarrage et une belle réussite dans sa carrière d'enseignant-chercheur à Maximilien. Je n'oublie pas non plus Raphaël T. à qui je souhaite bon courage pour la fin de sa thèse.

Je remercie bien évidemment mon équipe d'accueil au LIP6, LFI (anciennement partie de MALIRE), remplie de gens sympas et très talentueux au premier rang desquels Marie-Jeanne L. qui m'a accompagné au long de ces 4 années et 6 productions scientifiques, et qui a su m'écouter et me guider avec patience, compréhension et détermination du début à la fin et que je ne saurais jamais assez remercier, Marcin D. qui, bien que maintenant chez Axa Data Innovation Lab, a su me mettre sur la voie de la thèse et m'a donné beaucoup de conseils utiles. Je pense également ici à Adrien R. d'A. qui, bien que ne faisant plus vraiment partie intégrante de LFI a fait partie intégrante de mon encadrement de thèse à la suite de Marcin et a su me donner nombre de conseils et avis utiles, a toujours pris le temps avec Marie-Jeanne de me suivre et m'aider à perfectionner mes travaux.

Bien évidemment les autres membres de l'équipe LFI ne sont pas en reste : Christophe M., Bernadette B.-M. et Sabrina T. (et recomptons Nicolas L. qui était là au début) que j'ai côtoyés avec plaisir au long de ces 4 années, qui m'ont conseillé lors des réunions d'équipe et qui ont organisé sans relâche les goûters subséquents à ces réunions :)

1. Toute ressemblance à l'image d'épinal collant lourdement à un scientifique est bien sûr totalement fortuite.

Et que serait cette équipe sans les doctorants que j'ai pu y rencontrer : Fabon D., Amal O., Wenyi X., Christian K., Gilles M., Xavier R., Pierre-Xavier. L, Sébastien L., Wenlu Y., François M., Khalil L., Daniel M., Thibault L., Simon G., Marcin L. et Iwo D. Au milieu de tous ces gens formidables il semble plus facile d'appréhender la recherche et d'évoluer dans ce monde.

Une pensée particulière pour Jonathan A., avec qui j'ai de plus eu le plaisir de suivre le master, puis également d'enseigner le Python.

Et enfin à Arthur G. et Bénédicte L. avec qui j'ai pu partager de nombreuses joies universitaires, quelques bars, quelques restaurants chinois, quelques parties de Dota2 et même vaguement quelques chatons au fil de nombreuses années ! :)

Plus généralement la vie en laboratoire aurait été beaucoup moins plaisante ou évidente sans les nombreuses personnes qui le composent, en premier lieu les deux directeurs successifs Patrick G. et Jean-Claude B. qui ont réalisé un travail formidable d'autant plus qu'il n'est vraiment pas facile, les équipes administratives et techniques qui fournissent un soutien logistique toujours apprécié et d'ailleurs indispensable, je pense en particulier à Ghislaine M., Jacqueline L.-B., Nadine T. (pour la gestion), Christophe B., Vincent C., Jean-Pierre A., Benoît G. (pour trifouiller les ordis et les imprimantes ;)

Au hasard des noms et des rencontres, mes enseignants Vincent G., Nicolas B., Matthieu C., Ludovic D., Sylvain L. et Patrick G. à nouveau (MLIA) Bernd A., Anne L., Stéphane G. et Hubert N. (BD), Evripidis B., Christoph D. (RO) et Jean-Gabriel G. (ACASA) du master IAD, et de nombreux doctorants : Simon B., Elie G.-S., Mickaël P., Gabriella C., Ludovic D.-S., Aurélia L. D'autres équipes, je pense à Cécile P., Pacôme E., Loïc G., Hakan M., Narcisse N., Benjamin B. et Alexandre R.

A mes collègues enseignants également au premier rang desquels François P., le brave, qui fait tourner les UEs d'infos de Polytech' Sorbonne avec brio, également Thibault H., Jorge S., Cécile B., Olivier M. et Olivier S.

De mon bref passage à Paris Diderot également, une pensée particulière en premier pour Benjamin H. de M. à qui je souhaite bon courage et bonne chance pour s'insérer dans le monde universitaire français, puis Cristina S., Fabien de M., Michel H., Jean-Baptiste Y., Ines K., Olivier C., Fabien V., Constantin E., Hugues F., Carole D.

La grande famille ne la fac ne saurait être évoquée sans l'AEIP6, association au sein de laquelle j'ai pu autant m'épanouir et faire de belles rencontres que boire du café et manger des pizzas. Pour de sombres raisons d'égalitarismes et de manque de place, je fais un bisou à Florent C. mais ensuite les noms seront donnés en vrac, des vieux : Alexandre M., Clément D., Myriam R.-S., Arnaud S., Eric D., Adrien G., Marko M., Maxime L., des moins vieux : Alexandre C., Nicolas L., Victor D., James C., Chérif E.S., Paul G., Marie-Diana T., Louisa B., Kevin G., Yassine A., Erika B., Clément L., Vincent C.

Une mention spéciale tout de même aux licornes plus récentes : Yann-T R., Corentin A., Olivier B., Arthur R., Rémy B. et la formidable Margot C., une dreamteam so choupi.

Et bien sûr aux licornes toutes neuves et très goinfres : Daphné H.-B., Jérémie C., Alexandre L., Ines D., Mehdi E., Max C., Martin D., Kevin M., Frédérique F., William F., William H.

Une mention très spéciale à Margot C. (encore), Ilyas T., Déborah L., Amandine T., Maroua B., Kevin T., Nathanaël B.–T. et Rachel K. qui sont venus à mon pot de soutenance (ou même au blabla précédent le pot ! :) J'espère n'avoir oublié personne... (qu'au moins on ne m'en voudra pas d'avoir essayé)

Et enfin beaucoup d'amis rencontrés parmi mes anciens étudiants de Polytech' avec qui j'ai appris à découvrir les quais Saint-Bernard à la morte saison et les raclettes d'octobre : Fadwa A., Clément L., Nicolas T., Alice W., Yasmine M., Yann N., Julie NG., Victor V. qui sont venus à ma soutenance, Marie-Laure B., Jules C., Fanja R., Floriane D., Maxence B., Thanh Thào T., Julien T.

Je n'oublie pas les PeiP an III bien sûr, en particulier Clément OLC. et Jiax G. !

Parce qu'il m'arrive d'avoir des loisirs et de boire de l'alcool, d'avoir des discussions passionnantes, parfois deux de ces choses à la fois, je ne peux évidemment pas ne pas citer (en faisant attention à l'ordre alphabétique pour ne pas entretenir de rivalité) : Bruno S., Félicia S., Kévin V., Lucas V., Raphaël G.-R. qui ont tous droit à des bisous illimités.

En premier accessit je pense également à Romain G., Wandrille D., Damien S., Jia-Hua X., Clément D., Olivier G.

Parce que j'ai encore tellement de gens formidables dans ma vie, avec qui je passe des goûters, des dîners, des vacances, des mariages et même maintenant des baptêmes, je me dois de citer Charlotte B.(C.dL.), François B., Flore L., Jessica G., Florence P., Thibault D., Camille C., Matthieu L. et Caroline B. Avec leurs +1 lorsque leur nom n'y est pas, bien entendu.

Et bien évidemment par-dessus tout la formidable Margaux S. (S.-G. maintenant) que je connais depuis plus que la moitié de mon âge et que j'ai toujours plaisir à revoir, surtout les jours de ses hebdoversaires :) Je n'oublie pas également tou(te)s les super docteurs (éventuellement à brève échéance) qu'elle m'a permis de rencontrer, en premier lieu son génial mari Ludovic G. avec qui je dois toujours déjeuner à Jussieu même si ça fait pas mal d'années maintenant, la meilleure rousse du monde Annaelle M. (J.) et bien sûr les plus belles boucles brunes de Claire M.

La liste n'est pas encore finie mais heureusement il me reste encore un peu de place pour Laëtitia C., Hasina R., Pierre-Marie B., Alexandre F.-R., Damien D., Carl K. que je regrette de ne pas voir plus souvent. De paire avec ces charmants meudonnais je rajoute volontiers de vieux/vieilles ami(e)s Anouch et Elfie C., Andaine M., Audrey L., Vincent C., Lucie H., Kévin P., Claire et Aude L., Noémie T., Elodie P., Denis J., Marion B., Isabelle P. Et bien sûr le natif 91 du tonnerre, Alexandre L.

Petite mention spéciale également à Emeline B. qui sera la meilleure docteure en Media Studies, Design et Interactions Humain-Machine du monde ! Et bien sûr « big up » à Emile C. et un peu par extension à Sacha et Max.

Parce que toutes les bonnes choses, et surtout les longues listes, ont une fin, je termine (presque) avec ceux et celles qui ont été là le plus proche et/ou le plus longtemps. Mes incomparables parents Dr. Patrick C. et Michelle P., qui en plus de me dire de ranger ma chambre et d'accepter d'être caution de mon appartement ensuite, ont toujours été en soutien et à l'écoute, ils ont donc droit à de gros câlins et ma gratitude éternelle ! Ma sœur également, Morgane C. future 3ème docteur de la famille. A Piou, à Pims. Pensée émue pour Pascal qui n'aura jamais pu me voir enfin docteur mais à qui je pense toujours.

Plus loin dans ma famille Catherine, Hervé, Fiona, Mélissa, Clara, Renée, Didier, Pauline, Céline, Denis, les conjoints et les enfants !

Et évidemment Elisabeth parce que comment se motiver à travailler tout ça sans la certitude qu'après je peux retrouver Elisabeth ? Tout simplement :)

« Big up » à Marie-P, Christophe et Niels en passant ! (même un peu à Kitty)

Et enfin, un peu en vrac parce que je n'ai pas su les classer et parce que classer les gens ne leur rend pas nécessairement honneur : Marjolaine T. sœur de David T. frère de Amandine T., Léa G. harpiste de l'Obs. de Paris, Hélène S. notre prof de harpage, Aurélien J. de... Pokémon T. et tellement plus encore d'autant plus qu'il a assisté à ma soutenance, avec en accessit Laurent C. et Samy S., Kevin S. frère de Margaux S.-G., Florence T. de G-Scop, Ivan B. de LO, Audrey C. la belle infirmière qui, j'espère, se reconnaîtra.

Une pensée particulière pour Marc P. sans qui les cours d'histoire n'auraient pas été aussi chouette, ni bien sûr le voyage au Vietnam (pensée également à Huguette et Jean P., Suzanne B.)

Les gens du projet Homo Textilus : Marie R. pour ses efforts, Constance M., et aussi Ludovic K., Charles T.

Les dino-choupis du RIDE : Morgane M. et Anaïs D. (B.) !

Le CLAVIM que j'ai du quitter par manque de temps : Antoine E., Willy L., Johan L., Philippe B., Cédric L., Arnaud P., Vincent C. encore bien sûr, et aussi Angeline, Virginie, Charles, Charlie et les autres !

Un gros câlin à Richoult parce que c'est le meilleur créateur de sagas MP3 du monde, avec extension à Mel, Hesdea, Erika, Phil_Goud, LeMago, la team MP3@Paris et tellement tous les autres !

Finalement, à toi aussi, parce que tu as lu ces remerciements jusqu'à la fin. Encore plus si tu lis ma thèse ensuite ! :)

Cette thèse a été financée dans le cadre du projet ANR Homo Textilus « Le vêtement interactif et ses accessoires : prospection de l'habillage intelligent du corps ».

Référence du projet : ANR-11-SOIN-0007

Sommaire

Notations

Nous présentons ci-dessous les différentes notations mathématiques générales utilisées à travers le manuscrit et leur signification. Des notations plus spécifiques sont introduites là où elles sont nécessaires.

G	Graphe mathématique
V	Ensemble de sommets d'un graphe, ici de taille finie
n	Nombre de sommets d'un graphe de taille finie : $n = \|V\|$
E	Ensemble d'arêtes d'un graphe non orienté, de taille finie : $E \subseteq V \times V$
A	Ensemble d'arcs d'un graphe orienté, de taille finie : $A \subseteq V \times V$
m	Nombre d'arêtes ou d'arcs d'un graphe de taille finie : $m = \|E\|$ ou $m = \|A\|$
C	Ensemble des communautés détectées ou existantes sur un graphe
d_v	Degré du sommet $v \in V$
$\Gamma(v)$	Ensemble des voisins d'un sommet $v \in V$: les sommets partageant une arête ou un arc avec v
$V(S)$	Ensemble des sommets d'un sous-graphe S donné
$E(S)$	Ensemble des arêtes d'un sous-graphe S donné

Introduction générale

Contexte

Le monde qui nous entoure peut être vu comme un ensemble d'interactions entre éléments, qui peuvent être d'origine naturelle et concrète, par exemple des contacts entre être humains, ou bien basées sur des constructions plus abstraites comme les interactions économiques régissant les marchés financiers. Elles peuvent avoir lieu à l'échelle microscopique, telles les interactions entre protéines en permanence à l'œuvre dans notre corps, ou au contraire à une échelle macroscopique à l'instar des interactions gravitationnelles entre corps célestes.

Ces interactions ou ensembles d'interactions, appelés *systèmes complexes* (Vicsek, 2002), sont étudiés dans de nombreux domaines scientifiques : sociologie, physique, économie, biologie... Si les plus simples ont pu être expliquées au fil du temps, une grande majorité d'entre elles échappe encore à une modélisation qui les rendrait *prédictibles* : un exemple connu est la théorie cinétique des gaz en thermodynamique qui constitue encore aujourd'hui un objet de recherche intense en mathématiques et en sciences physiques.

Les outils et modèles développés ont traditionnellement été apportés par les mathématiques, par exemple sous la forme d'équations analytiques, algèbre matricielle... Avec le développement au XX$^{\text{ème}}$ siècle de la *théorie des graphes*, de nouvelles méthodes d'étude des systèmes complexes ont vu le jour en formalisant les interactions comme des *réseaux* dont les éléments forment les sommets et les arêtes représentent la présence, le type, la force, ou autres paramètres, de l'interaction.

Ces réseaux spécifiques sont nommés *complex networks* (Newman, 2003), littéralement *réseaux complexes*, ou plus fréquemment en français *graphes de terrain*, car ils sont utilisés afin de modéliser une situation réelle, « de terrain ». Ils ont l'avantage de présenter un modèle simple, accessible aux scientifiques non spécialistes en mathématiques : chaque sommet est représenté par un disque et chaque arête par une ligne joignant deux disques. De nombreuses visualisations sont donc envisageables (Scott, 2012).

Ces graphes de terrain sont utilisés dans tous les domaines de la science cités précédemment (Wasserman & Faust, 1994, Knoke, 2014), et bien d'autres, motivant en conséquence le développement de méthodes visant à les analyser, en particulier concernant les interac-

tions à très large échelle dont il résulte de très grands graphes de terrain : à la décennie des *big data*, l'échelle du milliard de sommets est en voie d'être dépassée.

Ils ne peuvent être analysés « à la main », comme le furent les premiers réseaux sociaux par exemple (Moreno & Jennings, 1938), en premier lieu pour une raison évidente de taille, mais également de compréhension des processus à l'œuvre : les interactions ayant lieu à différentes échelles, du voisinage entre éléments à une échelle globale très large pouvant impliquer des milliards d'éléments simultanément, il est très difficile voire impossible de les appréhender.

Une tâche très courante d'analyse des graphes de terrain, qui a engendré une littérature prolifique ces vingt dernières années, est la *détection de communautés* (Fortunato, 2009). Il s'agit de trouver dans un graphe des ensembles d'éléments interagissant plus particulièrement entre eux qu'avec le reste du graphe, formant ainsi les dénommées *communautés*. Cette notion de communauté est héritée de la sociologie mais revêt un sens beaucoup plus général dont il n'existe pas de définition universelle. On comprend aisément que la forme des communautés diffère selon les domaines : communautés sociologiques, protéines interagissant spécifiquement et fréquemment, acteur économiques liés... on pourrait définir une variété de communautés par graphe étudié. Cette solution ne serait bien évidemment pas satisfaisante, et une partie du travail lié à la détection de communautés dans des graphes de terrain est de poser une définition à la fois suffisamment générale pour qu'elle puisse être appliquée de façon générique à un grand nombre de graphes, mais tout de même particulière car aucune définition ne peut recouvrir l'ensemble des graphes de terrain.

Dans ce contexte, la thèse présentée dans ce manuscrit focalise son étude sur des réseaux particuliers : les réseaux mobiles opportunistes sociaux (Chaintreau et al., 2005), qui seront détaillés dans le chapitre 1. Ces réseaux sont des réseaux de télécommunications sans fil formés spontanément, d'où le terme « opportuniste », lorsque des objets mobiles communicants, tels que des smartphones ou les nombreux objets connectés de *l'Internet des Objets*, se situent à une distance rendant possible un échange direct d'information (Conti & Giordano, 2014). On considère le cas où ces objets sont déplacés par des humains, pour la plupart de manière passive étant donné qu'ils les portent sur eux : il en résulte des réseaux dynamiques, à la structure induite par le réseau social sous-jacent, en constante évolution car les positions des objets déterminant l'existence de ces liens dépendent d'êtres humains et de leur comportement. Ces particularités posent des contraintes pour effectuer une détection de communautés en temps réel et ce contexte spécifique a motivé les travaux présentés dans cette thèse.

Travaux de thèse

Afin de tenir compte des contraintes des réseaux mobiles opportunistes sociaux, les travaux de thèse présentés dans ce manuscrit sur la détection de communautés se placent

dans le paradigme récent et actif des méthodes orientées-sommet, en alliant le traitement de graphes Think-Like-a-Vertex (McCune et al., 2015) aux méthodes de détection de communautés basées sur des leaders ou des graines (Kanawati, 2014a) détaillées dans le chapitre 2 : celles-ci présentent en effet des propriétés de décentralisation qui autorisent des implémentations parallèles et distribuées appropriées au cadre applicatif considéré.

Dans ce contexte, nous proposons d'une part un principe global de fonctionnement original que nous mettons en œuvre et déclinons dans trois algorithmes dédiés à trois configurations différentes de la tâche de détection de communautés : l'algorithme VOLCAN d'abord, considère le cas de référence des communautés disjointes dans un graphe statique. Nous l'étendons ensuite avec l'algorithme LOCNeSs au cas des communautés recouvrantes, qui autorisent un sommet à appartenir à plusieurs communautés simultanément : cette généralisation donne plus de flexibilité à la détection et la rend plus appropriée au cadre applicatif considéré. Nous examinons également le cas des graphes dynamiques, c'est-à-dire dont les sommets et les arêtes évoluent au cours du temps, auquel est consacré l'algorithme DynLOCNeSs. Chacun des algorithmes est associé à une implémentation décentralisée et fait l'objet d'une étude expérimentale qui permet d'évaluer la qualité des résultats qu'il fournit et de les comparer aux méthodes de l'état de l'art.

Nous considérons également, dans un cas particulier de réseau mobile ad-hoc spontané et décentralisé issu d'une application réelle de vêtements intelligents et communicants, une tâche de cheminement permettant d'identifier des interlocuteurs. Nous proposons une stratégie de recommandation utilisant la structure communautaire, à l'aide de l'algorithme SWAGG.

Organisation du manuscrit

Le chapitre 1 introduit des notions de base (définitions, outils mathématiques...) ainsi que des précisions sur le contexte applicatif ciblé, nécessaires à la compréhension de la suite du manuscrit. La partie I décrit alors les algorithmes proposés pour effectuer la détection de communautés sur des graphes statiques : le chapitre 2 présente un état de l'art des méthodes existantes, le chapitre 3 décrit l'algorithme VOLCAN détectant des communautés disjointes, puis le chapitre 4 l'algorithme LOCNeSs dédié aux communautés recouvrantes.

La partie II traite du cas des graphes dynamiques et suit le même schéma, en proposant tout d'abord un état de l'art dans le chapitre 5 suivi, dans le chapitre 6, d'une description de l'algorithme de détection de communautés dans des graphes dynamiques que nous avons proposé, DynLOCNeSs.

Enfin, le chapitre 7 présente l'application réelle liée à un réseau mobile opportuniste social, nécessitant une méthode de détection de communautés : l'orientation de personnes

participant à un événement social et munies de dispositifs mobiles communicants, dont les stratégies sont évaluées à l'aide de l'algorithme SWAGG.

Nous concluons ce manuscrit en résumant les contributions de la thèse et en présentant les perspectives qu'elle ouvre.

Plusieurs annexes utiles à la compréhension des expériences menées figurent en fin de manuscrit, elles concernent les critères d'évaluation (annexe A) et les jeux de données (annexe B) utilisés ainsi que des détails de preuves mathématiques (annexe C).

Chapitre 1

Contexte et application considérés

Sommaire

Ce chapitre introduit les définitions et notations des principaux concepts manipulés dans cette thèse : la section 1.1 contient quelques rappels sur les graphes en tant qu'objets mathématiques, la section 1.2 les considère en tant qu'objets réels, sous la forme de graphes de terrain. La section 1.3 introduit la tâche au cœur des travaux présentés dans cette thèse, la détection de communautés dans les graphes, qui est décrite en détail dans des chapitres ultérieurs ainsi que les algorithmes existants pour la réaliser. La section 1.4 décrit le type de graphes de terrain particulier considéré dans la thèse, les réseaux mobiles opportunistes, et présente leurs spécificités ainsi que les défis qu'ils soulèvent.

1.1 Rappels et notations à propos des graphes

Cette section rappelle quelques définitions classiques de la théorie des graphes, telles qu'elles peuvent par exemple être trouvées dans des ouvrages de référence (Gondran & Minoux, 1995, Newman, 2003, Barabási, 2016), et introduit les notations utilisées dans la thèse. La première sous-section donne quelques définitions générales décrivant ce qu'est un graphe, en tant qu'objet mathématique, la seconde présente quelques éléments topologiques caractéristiques des graphes et la troisième quelques caractéristiques numériques.

1.1.1 Définitions élémentaires

Nous présentons ici des définitions générales d'éléments mathématiques utilisés dans l'ensemble du manuscrit.

Graphe

Un *graphe* G est défini comme un couple $G = (V, E)$ composé d'un ensemble de *sommets* V (*vertices* en anglais) et d'un ensemble d'*arêtes* E (*edges*) reliant certaines paires de sommets de V. Formellement, $E \subseteq V \times V$. Nous considérons le cas de graphes finis, c'est-à-dire ayant un nombre fini de sommets et d'arêtes et notons $n = |V|$ et $m = |E|$.

Incidence, degré, adjacence, voisinage

Le nombre d'arêtes *incidentes* à un sommet v, c'est-à-dire telles que v constitue une de ses extrémités, est appelé *degré* de v, noté $d(v)$ ou d_v. Deux arêtes incidentes à un même sommet sont dites *adjacentes*.

Les sommets avec lesquels v partage une arête sont appelés les *voisins* (*neighbours*) de v et sont notés $\Gamma(v)$. Deux sommets reliés par une arête sont donc voisins, on les dit également *adjacents*. On peut représenter le graphe G sous la forme d'une matrice d'adjacence A de taille $n \times n$, où chaque $A_{i,j}$ vaut 1 ou 0 selon qu'il existe ou non une arête entre les sommets i et j.

Sous-graphe

Un graphe $H = (V_H, E_H)$ est un *sous-graphe* de $G = (V_G, E_G)$ si et seulement si $V_H \subseteq V_G$ et $E_H \subseteq E_G$. G est alors appelé le *super-graphe* de H.

H est dit *induit* par V_H si $\forall u, v \in V_H, (u, v) \in E_G \Leftrightarrow (u, v) \in E_H$, autrement dit on définit H uniquement par l'ensemble de ses sommets V_H et il contient implicitement toutes les arêtes du super-graphe G incidentes à deux extrémités de ces sommets.

Orientation, pondération, complétude

Les arêtes peuvent avoir un sens, amenant par exemple à distinguer pour l'arête (u, v) les sens $u \to v$ et $v \to u$. Dans ce cas on les appelle *arcs* et le graphe G est dit *orienté*. On distingue alors pour un sommet v un arc entrant (aboutissant à v) d'un arc sortant (issu de v). Par exemple l'arc $(u \to v)$ est un arc entrant pour v et sortant pour u. Le degré entrant d_{in} de v (resp. sortant d_{out}) est ainsi le nombre d'arcs entrants (resp. sortants) de v.

Une arête (u, u) joignant un sommet à lui-même est appelée *boucle* (*loop* ou *self-loop*).

Les arêtes peuvent également se voir affecter un poids –entier ou réel, positif ou négatif– et le graphe est alors dit *pondéré* (*weighted*).

Un graphe est dit *simple* s'il est non-orienté, non-pondéré et sans boucle.

Un graphe simple G est dit *complet* si tout couple de sommets distincts est relié : $\forall (u, v) \in V^2, u \neq v \implies (u, v) \in E$. Un graphe complet, non-orienté et fini vérifie $m = \frac{n(n-1)}{2}$.

Dans toute la thèse, sauf mention contraire, nous considérons des graphes simples et finis.

1.1.2 Chaînes, chemins, cycles, circuits, cliques

Nous donnons ici quelques définitions supplémentaires utiles pour comprendre le manuscrit. On considère un graphe $G = (V, E)$.

Définition Une *chaîne* est une suite d'arêtes successivement adjacentes.

Formellement, une chaîne S de taille p reliant u à v est une suite de p arêtes de E, $S = \{e_i, i = 0..p-1\}$ avec $e_0 = (u, s_1), e_1 = (s_1, s_2), ..., e_{p-2} = (s_{p-2}, s_{p-1}), e_{p-1} = (s_{p-1}, v)$. Chaque arête $e_i \in S, 0 \leq i < p$ est donc incidente à un sommet auquel est également incidente l'arête e_{i+1}.

Une chaîne est dite *élémentaire* si elle ne passe pas deux fois par le même sommet. Tout graphe comportant au moins deux sommets possède au moins une chaîne élémentaire.

Une chaîne est appelée *walk* en anglais, une chaîne élémentaire *trail* ou *path*[2].

2. La notation anglophone de *path* est ambiguë. Le mot *path* seul désigne généralement une chaîne dans laquelle aucun sommet n'appartient à plus de deux arêtes, i. e. elle ne passe pas deux fois par le

Définition Un *chemin* est une chaîne dans un graphe orienté, autrement dit une suite d'arcs successivement adjacents : formellement, un chemin T de taille p reliant u à v est un ensemble d'arcs $T \subseteq A$, $T = \{(u \to t_1), (t_1 \to t_2)...(t_{p-2} \to t_{p-1}), (t_{p-1} \to v)\}$. Chaque arc $a_i \in T, 0 \leq i < p$ est donc entrant sur un sommet pour lequel l'arc t_{i+1} est sortant.

Un chemin peut être élémentaire au même sens qu'une chaîne s'il ne passe pas deux fois par le même sommet.

Un chemin est appelé *directed path*[2] en anglais.

Définition Un *cycle* est une chaîne dont les sommets de départ et d'arrivée sont identiques. Formellement, une chaîne S est donc un cycle si :

$$S = \{(u, s_1), (s_1, s_2)...(s_{p-2}, s_{p-1}), (s_{p-1}, u)\}$$

Un cycle peut être élémentaire au même sens qu'une chaîne s'il ne passe pas deux fois par le même sommet.

Un cycle est appelé *closed walk* en anglais, l'appellation *cycle* désignant généralement un cycle élémentaire.

Définition Un *circuit* est un cycle orienté, i.e. un chemin dont les sommets de départ et d'arrivée sont identiques. Formellement, un chemin T est donc un circuit si :

$$T = \{(u \to s_1), (s_1 \to s_2)...(s_{p-2} \to s_{p-1}), (s_{p-1} \to u)\}$$

Un circuit peut être élémentaire au même sens qu'une chaîne et qu'un cycle s'il ne passe pas deux fois par le même sommet.

Un circuit est dit *minimal* s'il n'est inclus dans aucun autre.

Un circuit est appelé *directed cycle* en anglais, l'appellation *circuit* étant ambiguë car pouvant référer à un *directed cycle* aussi bien qu'à une *closed walk* permettant les répétitions.

Définition Une *clique* c d'un graphe $G = (V, E)$ est un sous-ensemble de sommets dont chacun est connecté à tous les autres dans G :

$$c \subseteq V, \forall u \in c, \exists v \neq u \in c \ / \ (u, v) \in E$$

c est donc un sous-graphe induit complet de G. On appelle k-clique une clique de k sommets (k entier). Une 3-clique est communément appelée *triangle*.

même sommet. Le mot *trail* est parfois utilisé pour une chaîne ne contenant pas deux fois la même arête. La mention explicite *directed* devant l'un des termes sus-mentionnés indique de manière non-ambiguë l'utilisation dans un contexte orienté.

Une propriété, importante dans la suite de ce manuscrit car liée au fonctionnement des algorithmes décrits dans la thèse et dont la preuve peut être trouvée dans un ouvrage de référence (Gondran & Minoux, 1995), décrit la présence de cycles dans un graphe :

Propriété 1.1.1 (Présence de cycles)**.** *Soit* $d_{\min}$ *le degré minimal de* G*, c'est-à-dire* $d_{min} = \min_{v \in V} d_v$*.*

Un graphe simple pour lequel $d_{min} \geq 2$ *possède au moins un cycle de longueur au moins égale à* $d_{\min} + 1$*.*

Exemple Un graphe simple $G = (V, E)$ de degré minimal $d_{\min} = 2$ possède au minimum $n = 3$ sommets. Une configuration où le graphe comporte trois sommets, chacun de degré 2, n'admet une répartition d'arêtes qu'en triangle, c'est-à-dire un cycle de longueur $d_{\min} + 1 = 3$

1.1.3 Connexité

Définition Une *composante connexe* de G est un sous-graphe H induit de G tel qu'il existe une chaîne reliant tout couple $(u, v), u \neq v$ de sommets de H.

Définition Un graphe simple non-orienté est dit *connexe* s'il forme une seule composante connexe, c'est-à-dire qu'il existe une chaîne entre tout couple $(u, v), u \neq v$ de ses sommets.

1.1.4 Nombre isopérimétrique et conductance

Cette section introduit deux notions essentielles, utiles pour comprendre le fonctionnement des méthodes de traitement des graphes orientées sommet (cf chapitre 2) : le nombre isopérimétrique et la conductance [3] d'un sous-graphe.

On considère dans la suite de cette sous-section un sous-graphe non vide $S_G = (V(S_G), E(S_G))$ du graphe $G = (V, E)$ tel que $|V(S_G)| \leq \frac{1}{2}|V|$ et l'on note $\overline{S_G} = G \setminus S_G, V(\overline{S_G}) = V \setminus V(S_G)$.

Coupe d'un graphe

Nous rappelons rapidement la notion de *coupe* d'un graphe, en réalité très riche et très étudiée (Gondran & Minoux, 1995) :

Définition Une *coupe* d'un graphe est une partition de ses sommets V en deux sous-ensembles V' et $\bar{V}'$, définie par l'ensemble des arêtes reliant un sommet de V' à un sommet de $\bar{V}'$.

La taille d'une coupe est le cardinal de cet ensemble.

3. Les noms sont repris d'équivalents en géométrie euclidienne/riemannienne/spectrale.

Coupe autour d'un ensemble de sommets

En particulier, pour formaliser la notion précédente, nous utilisons la notation $\partial V(S_G) = \{(u,v) \in E | u \in V(S_G), v \in V(\overline{S_G})\}$, pour représenter l'ensemble des arêtes ayant un sommet dans S_G et l'autre en-dehors de S_G. ∂V est donc bien un ensemble d'arêtes et non de sommets.

Nombre isopérimétrique

Définition Le *nombre isopérimétrique* (aussi *constante de Cheeger*) de S_G, noté $h(S_G)$, est le rapport du nombre d'arêtes « entre » S_G et $\overline{S_G}$ et du nombre de sommets de S_G (Bollobás, 1988, Mohar, 1989), soit formellement :

$$h(S_G) = \frac{|\partial V(S_G)|}{|V(S_G)|}$$

Le nombre isopérimétrique possède la propriété suivante : sa valeur pour un graphe G est égale au minimum des nombres isopérimétriques de tous les sous-graphes de G de taille inférieure à la moitié du graphe originel :

$$h(G) = \min_{S_G \subset G, |V(S_G)| \leq \frac{1}{2}|V|} h(S_G)$$

Le nombre isopérimétrique d'un sous-graphe apporte un certain nombre d'informations, c'est par exemple une bonne mesure pour estimer sa connectivité (cf. annexe C, p. 193) et un indicateur sur la possibilité de réaliser une coupe de très petite valeur contenant uniquement ce sous-graphe. Un certain nombre de propriétés sont exposées par Mohar (1989).

Conductance

La *conductance* ϕ (Bollobás, 1998, Mihail et al., 2003) est proche du nombre isopérimétrique mais fait appel à un dénominateur différent, basé sur la notion de volume (noté *vol*) d'un ensemble de sommets :

Définition Le *volume* d'un ensemble de sommets S la somme de leurs degrés :

$$vol(S) = \sum_{v \in S} d(v)$$

On peut remarquer que pour un graphe $G = (V, E)$, le volume est $vol(V) = 2|E| = 2m$.

On peut maintenant exprimer la *conductance*.

Définition La *conductance* ϕ de l'ensemble de sommets $V(S_G)$ est définie par :

$$\phi = \frac{|\partial V(S_G)|}{\min(vol(V(S_G)), vol(\overline{V(S_G)}))}$$

De même que pour le nombre isopérimétrique, la conductance d'un graphe est le minimum des valeurs de conductance de tous les sous-graphes de taille inférieure ou égale à la moitié du nombre de sommets de ce graphe.

Discussion sur la conductance et le nombre isopérimétrique

Trouver le ou les sous-ensembles d'un graphe minimisant la conductance, ou le nombre isopérimétrique, est très difficile. En effet un algorithme glouton nécessite d'énumérer tous les sous-graphes possibles (jusqu'à $n/2$) ce qui est impossible à réaliser en temps raisonnable, mis à part pour des graphes de très petite taille, car le problème est NP-complet (Gondran & Minoux, 1995).

En pratique on mesure donc la conductance d'ensembles déjà constitués, comme des coupes ou des décompositions/partitions (cf. section 2.2, p. 41, particulièrement 2.2.3, p. 43, et travaux de Leskovec et al. (2008)).

En détection de communautés, la conductance est souvent préférée au nombre isopérimétrique (Schaub et al., 2017) car elle rend mieux compte de la connectivité entre le sous-graphe considéré et le reste du graphe. Souvent, ce sous-graphe est d'ailleurs une coupe du graphe et la conductance est utilisée pour mesurer la connectivité liée au flot entre les deux.

1.1.5 Types de graphes

Il existe plusieurs types de graphes décrivant la manière dont sont organisés leurs sommets ou arêtes. Il est utile de connaître certains de ces types, car ils possèdent des propriétés spécifiques permettant une utilisation particulière, à l'instar des arbres (arbre de recherche par exemple). Lorsqu'une topologie de graphe spécifique est identifiée, on peut la nommer et l'étudier plus particulièrement. Le graphe est alors dit *structuré*. Dans le cas contraire il est dit *quelconque*.

Cette sous-section décrit quelques topologies de graphes structurés remarquables, avant de discuter de modèles de représentation enrichis.

Topologies simples, graphes structurés

Nous présentons dans un premier temps des topologies simples formant des graphes structurés : homogène, hiérarchique, polarisée. Ces types de graphes sont représentés figure 1.1.

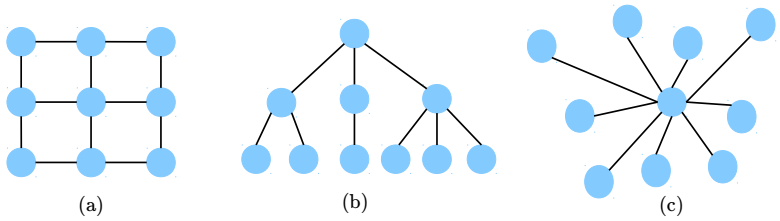


FIGURE 1.1 – Topologies homogène (a), arborescente (b) et polarisée (c)

Homogène ou régulière

Les graphes homogènes reproduisent un schéma régulier. Le schéma le plus connu est celui de la grille (*grid* ou *mesh* en anglais, représentée figure 1.1 (a)). Dans cette topologie, tous les sommets sont connectés en pair-à-pair à leurs voisins dans la grille. Il est important de différencier une topologie régulière (décrite ici) d'un graphe régulier : un graphe dont tous les sommets ont le même degré.

Dans les réseaux de communication, elle est utilisée pour former des réseaux sous le nom de *grid*. La topologie réseau nommée *mesh* existe également, mais diffère dans le sens où tous les nœuds sont connectés en pair-à-pair à tous les autres (formant un graphe complet).

Arborescente ou hiérarchique

Un graphe hiérarchique (voir figure 1.1 (b)) peut être découpé en « niveaux » ou couches hiérarchiques, c'est notamment le cas des arbres.

Les réseaux adoptant cette technologie sont centralisés, l'information remontant jusqu'au plus haut niveau du graphe (« sommet » ou « racine » pour un arbre) avant de transiter.

Polarisée

Un graphe polarisé (voir figure 1.1 (c)) est tel que tous les sommets sont rattachés à un seul et même sommet central, ou éventuellement une clique centrale, appelé(e) pôle.

L'architecture de réseau correspondante est évidemment très centralisée. Elle est utilisée par exemple pour les bornes d'accès de téléphonie mobile.

Modèles de représentation enrichis

On constate que le seul cadre de la théorie des graphes classique ne suffit généralement pas à capturer la complexité de situations réelles : les graphes de terrain ne sont, à de rares exceptions près, pas des graphes structurés. Détecter des communautés sur de tels graphes

n'a pas de sens, l'organisation des sommets et arêtes ne reflète a priori pas de différences de densité liées à la présence de communautés.

De nombreux travaux ont donc apporté de nouveaux outils mathématiques de théorie des graphes, qui ne sont pas des topologies, afin de traiter des situations plus riches et complexes. On peut citer par exemple les graphes bipartis (Asratian et al., 1998) et les graphes multi-couches, aussi appelés multiplexes ou multi-dimensionnels (Kivelä et al., 2014).

Dans le cadre de la thèse, nous considérons uniquement des graphes de terrain, tels que décrits dans la section suivante. En effet, ils ne peuvent pas être structurés car on souhaite y détecter des communautés.

1.2 Graphes de terrain

La section précédente a présenté quelques rappels sur les graphes en tant qu'objets mathématiques. Cette section s'intéresse à des graphes particuliers, dont il n'existe pas de définition formelle universelle et dont le processus de formation est souvent difficile à modéliser. Appelés *graphes de terrain*[4] ou, en anglais, *complex networks*, ils sont issus d'observations empiriques, comme détaillé dans la prochaine sous-section.

Nous en proposons tout d'abord une définition informelle dans la section 1.2.1 ainsi que plusieurs exemples de leur utilisation dans la section 1.2.2. Nous discutons de leurs caractéristiques topologiques particulières dans la section 1.2.3, p. 15 ainsi que des problématiques soulevées par leur exploitation dans la section 1.2.4. La section 1.2.5, p. 20 présente brièvement des modèles de génération de tels graphes, permettant de reproduire une ou plusieurs propriétés topologiques particulières connues.

1.2.1 Définition informelle

Un graphe de terrain est un graphe possédant des propriétés statistiques et topologiques non triviales (voir section 1.2.3, p. 15), par opposition aux topologies décrites dans la sous-section précédente. Ils sont généralement la représentation de réseaux étudiés dans le « monde réel » et tirent leurs propriétés particulières de l'organisation même de ces réseaux. De fait leur processus de génération sous-jacent est, le plus souvent, inconnu, comme illustré dans les exemples ci-dessous.

Dans ce manuscrit, afin de distinguer l'objet mathématique de « l'objet réel », comprendre la situation représentée, nous utilisons le terme de graphe, associé à ceux de sommets et arêtes, pour le premier, et les termes réseau, nœuds et liens pour le second. Dans la littérature, certains auteurs choisissent de n'utiliser qu'un seul de ces ensembles de termes, d'autres les utilisent indifféremment et de façon interchangeable[5].

4. Le terme *réseaux complexes*, traduction littérale de l'anglais, est parfois rencontré.

5. Note : le terme anglais pour graphe est *graph* et celui pour réseau est *network*. On note ainsi la bizarrerie de la traduction du groupe nominal *complex network* par *graphe de terrain*.

Il faut noter que le formalisme du graphe simple n'est pas le seul utilisé et qu'il n'est pas toujours le plus adapté pour décrire ces objets réels. D'autres modèles enrichis existent, tels que ceux évoqués 1.1.5, p. 11.

1.2.2 Exemples de graphes de terrain

La flexibilité du concept mathématique de graphe permet de modéliser un grand nombre de situations relationnelles et/ou interactives (Bornholdt & Schuster, 2006). Le problème même ayant donné naissance aux graphes, celui des sept ponts de Königsberg formulé et résolu par Euler (1741), est un graphe de terrain : le graphe est introduit afin de modéliser la topographie particulière de la ville de Königsberg [6].

Les situations modélisables par des graphes de terrain sont nombreuses et variées (Costa et al., 2011), mais de manière générale on compte des champs plus contributeurs à l'analyse des réseaux et des graphes de terrain dont :

- la biologie : réseaux d'interactions neuronales, protéine-protéine... (Alon, 2003)
- la physique, particulièrement la mécanique statistique : interactions entre particules, ondes... (Dörfler & Bullo, 2014)
- l'économie et les transports : réseaux de flux financiers, commerciaux, aériens, routiers, touristiques... (Chowdhury et al., 2000, Knoke, 2014)
- la sociologie : réseaux sociaux réels (lieu d'étude, de travail) (Wasserman & Faust, 1994), réseaux sociaux virtuels (facebook, Twitter), réseaux de collaboration (co-publication scientifique) (Barabási et al., 2002, Newman, 2004a), réseaux politiques (Knoke, 1994) mais également des réseaux de relations entre animaux (Sueur et al., 2011).
- l'informatique, particulièrement les télécommunications : réseaux mobiles (Daly & Haahr, 2007), Internet (Newman, 2003)...

Il existe bien entendu de multiples applications plus anecdotiques, formalisant sous forme de graphe des objets d'étude dont la structure n'est pas proche d'un réseau. Par exemple, Melo et al. (2016) construisent un graphe à partir de signaux audio et utilisent la détection de communautés afin de catégoriser des pièces de musique.

Dans le cadre de cette thèse, nous considérons des graphes de terrain particuliers, basés sur une hybridation de réseaux sociaux et de réseaux de communication : des réseaux mobiles opportunistes sociaux comme décrits dans la section 1.4, p. 30.

Exemple développé

Nous proposons de développer un exemple de réseau social réel qui servira à plusieurs reprises dans la suite du manuscrit : un laboratoire de recherche composé de plusieurs

6. Aujourd'hui Kaliningrad, dans le territoire éponyme de la fédération de Russie.

équipes. Nous retenons comme interactions leurs relations professionnelles : certains chercheurs travaillent ensemble et publient des articles en commun. Sous forme de graphe, ces interactions sont modélisées de la façon suivante : chaque sommet représente un chercheur et une arête entre deux sommets le fait que ces chercheurs publient ensemble.

1.2.3 Caractéristiques fréquentes

L'une des principales caractéristiques des graphes de terrain est leur topologie particulière, qui pourrait de prime abord sembler multi-polaire mais qui tient parfois du graphe quelconque (cf. section 1.1.5, p. 11). En l'absence de définition formelle, ils semblent néanmoins caractérisés par certaines propriétés que l'on retrouve conjointement, plus ou moins fréquemment : cette section évoque six de ces propriétés principales, qui ont trait principalement à la densité et à la distribution des degrés, puis décrit des discussions récentes sur leur universalité. Ces propriétés sont détaillées dans de nombreux ouvrages de référence, par exemple Newman (2003); Boccaletti et al. (2006).

1.2.3.1 Faible diamètre, petit-monde

Définition Le *diamètre* d'un graphe non orienté est la longueur de la plus longue chaîne existant entre deux sommets de ce graphe.

Les graphes de terrain sont connus pour avoir un diamètre généralement faible, en particulier dans le cas des réseaux sociaux (Watts & Strogatz, 1998, Boccaletti et al., 2006). Cette propriété est liée à un effet dénommé *small-world*[7], d'après le terme introduit par Milgram (1967) : celui-ci a réalisé, dans les années 1960 aux Etats-Unis, une expérience afin de valider une hypothèse selon laquelle le diamètre des réseaux sociaux a tendance à être faible.

Dans cette expérience, une personne choisie aléatoirement parmi un ensemble présélectionné devait faire parvenir une lettre à une autre personne sélectionnée de la même manière, mais située dans une zone socio-géographique éloignée. Pour cela, elle devait d'abord la transmettre à une personne de son choix qui devait à son tour la transmettre à une personne de son choix avec pour objectif d'atteindre la personne cible. Le postulat derrière ce mécanisme est que chaque expéditeur va tenter d'envoyer la lettre à une de ses connaissances dans une zone géographique la plus proche possible de la cible.

Le degré de séparation est le nombre de personnes par lesquelles a transité la lettre avant d'atteindre sa destination. Milgram a observé que dans les cas étudiés lors de son expérience, ce degré était d'environ 6. Bien que cette expérience ait été critiquée pour ses nombreux biais, entre autres le fait que seules 15% des lettres aient effectivement atteint leur cible, la faible taille moyenne des chemins entre deux personnes dans un réseau social

7. Petit-monde en français, mais le terme anglais reste largement prévalent

a été corroborée par différentes études depuis (Watts & Strogatz, 1998, Backstrom et al., 2012).

1.2.3.2 Densité locale forte mais globale faible

Définition La *densité (globale)* D_G d'un graphe G simple non orienté est définie comme

$$D_G = \frac{2m}{n(n-1)}$$

La densité est bornée entre $[0, 1]$, en effet un graphe ne comportant aucune arête a bien entendu une densité de 0, et un graphe complet, simple et non orienté, possède $\frac{n(n-1)}{2}$ arêtes, sa densité est donc de 1.

Un graphe dont la densité est très forte est dit *dense*, au contraire si sa densité est faible il est dit *épars*, *creux* ou *parcimonieux* (*sparse* en anglais). Cependant, la question des valeurs de densité correspondant à ces adjectifs est ouverte : ces valeurs ne sont pas universellement fixées et sont à définir selon les graphes étudiés. Il est généralement considéré que $0 < D_{\text{sparse}} \ll 0.5 \ll D_{\text{dense}} \leq 1$.

Densité locale

La densité d'un graphe ne dépendant que de son nombre de sommets et d'arêtes, on peut calculer la densité d'un sous-graphe quelconque, que l'on appelle densité *locale* de ce sous-graphe, par opposition à la densité *globale* du graphe entier.

Densité des graphes de terrain

La plupart des graphes de terrain, en particulier les réseaux sociaux (Wasserman & Faust, 1994), ont une densité globale faible, mais localement forte : leur densité globale se rapproche de celle d'un graphe épars mais il est possible d'y identifier des sous-graphes denses. Cette propriété constitue la base intrinsèque de la présence des *communautés* (voir section 1.3, p. 23).

D'une manière générale, il est considéré que la densité est très inégale dans les graphes de terrain (Newman, 2003).

1.2.3.3 Coefficient de clustering fort

Le *coefficient de clustering* (Wasserman & Faust, 1994) d'un ensemble de sommets d'un graphe mesure à quel point le voisinage de ces sommets est connecté et donc à quel point ces sommets sont susceptibles de former des *clusters* (*regroupements* ou *amas* en français). Il existe une définition globale, appliquée à l'intégralité d'un graphe, et une version locale, appliquée à un sous-graphe.

FIGURE 1.2 – Triplet non fermé (à gauche) et fermé, c'est-à-dire un triangle (à droite)

Coefficient global

Le coefficient de clustering global mesure à quel point les ensembles de trois sommets connectés entre eux (triplets connectés) sont fermés, c'est-à-dire forment des triangles comme nous l'illustrons sur la figure 1.2. Plus un graphe contient de triangles, plus les sommets sont considérés comme « regroupés ».

$$CC_{global} = 3 \times \frac{|triangles|}{|triplets\ connectes|}$$

Le facteur 3 est dû au fait qu'un triangle est formé de trois triplets connectés. Ainsi, un graphe constitué uniquement de triangles ou de cliques plus grosses a un coefficient de clustering global de 1.

Coefficient local

Le coefficient local a une définition différente puisqu'il se place du point de vue d'un sommet : il mesure à quel point le voisinage de ce sommet est proche de former une clique (Watts & Strogatz, 1998). Soit $E(\Gamma(v)), v \in V$ l'ensemble des arêtes du sous-graphe induit par le voisinage de v. Le coefficient de clustering local du sommet v, pour un graphe simple non orienté, est défini par :

$$CC_{local}(v) = \frac{2|E(\Gamma(v))|}{d_v(d_v - 1)}$$

Il est aisé de comprendre que, lorsque le voisinage de v forme une clique, ce coefficient vaut 1 et réciproquement. Il peut être considéré comme l'indicateur de « cliquicité » du voisinage de v, il est lié à la densité de ce voisinage. Ainsi, un voisinage dense a nécessairement une haute valeur de coefficient clustering local et réciproquement.

1.2.3.4 Sans échelle et queue lourde

Un graphe est dit *sans échelle* ou invariant d'échelle (*scale-free* en anglais) si sa distribution de degrés suit une loi de puissance, au moins asymptotiquement. Formellement, pour un sommet v quelconque d'un graphe, alors :

$$P(d_v = k) \sim k^{-\gamma}$$

où γ est un paramètre d'échelle strictement positif dont la valeur influe sur la structure du graphe.

La prévalence du caractère sans échelle des graphes de terrain est actuellement très discutée (cf. sous-section 1.2.4, p. 20). S'il est généralement admis que la distribution des degrés des graphes de terrain n'est pas uniforme, tous les graphes de terrain ne sont pas sans échelle.

Queues de la distribution des degrés

Une propriété qui a reçu moins de publicité que l'invariance d'échelle est la *queue* (ou *traîne*) de la distribution des degrés du graphe. Toutes les notions de probabilités et statistiques suivantes sont détaillées dans des ouvrages de référence, par exemple celui de Foss et al. (2011) pour les queues lourdes.

En statistiques, la queue d'une distribution correspond à la concentration de valeurs éloignées de la valeur centrale. Cette concentration est liée à un paramètre de la forme de la distribution nommé *coefficient d'aplatissement* ou *kurtosis*, qui est défini comme le moment d'ordre quatre de la variable centrée réduite.

On utilise souvent une version corrigée de ce coefficient (*excès d'aplatissement*), afin de bénéficier des propriétés suivantes : la queue de la loi normale a alors un coefficient nul (elle est dite mésokurtique), une queue plus légère un coefficient négatif (platikurtique) et une queue plus lourde un coefficient positif (leptokurtique). On parle de queue *lourde* lorsqu'elle n'est pas exponentiellement bornée[8].

Les graphes de terrain ont des distributions de degrés à queue leptokurtiques, généralement lourdes. On peut qualifier visuellement la courbe de cette queue de « verticale » : très peu de sommets de très hauts degrés et beaucoup de sommets de très faibles degrés.

1.2.3.5 Composante connexe géante

La plupart des graphes de terrain sont soit connexes, soit constitués d'une composante connexe géante, rassemblant une écrasante majorité de leurs sommets (Newman, 2003).

En effet, on constate empiriquement que la plupart des graphes de terrain ont au moins une composante connexe contenant plus de 50% des sommets et il n'est pas exceptionnel que cette composante contienne plus de 95% des sommets (Amaral et al., 2000, Kleinberg, 2002, Newman, 2003). Intuitivement, il est facile de lier cette propriété avec l'effet petit-monde (cf. section 1.2.3.1, p. 15) : en effet, par définition, s'il existe une chaîne entre la majorité des nœuds d'un réseau, ils font partie de la même composante connexe.

8. Par exemple, la courbe d'une exponentielle est leptokurtique, son coefficient corrigé est de 6, et le coefficient corrigé d'une distribution lognormale de variance 1 est supérieur à 110. La plupart des distributions de degrés dans les graphes de terrain ont des courbes de coefficient corrigé supérieur à 6 : elles sont sous-exponentielles.

1.2.3.6 Assortativité et homophilie : club des riches, attachement préférentiel

Un phénomène appelé assortativité (*assortativity*), lié au processus de génération d'un réseau, caractérise la tendance de ses nœuds à préférer établir des connexions avec des nœuds similaires ou compatibles selon la nature du réseau. C'est une propriété relative à l'homophilie en sociologie : la tendance à préférer des individus de nature similaire (McPherson et al., 2001, Kossinets & Watts, 2009). La manifestation la plus souvent retenue de cette assortativité dans la formation d'un réseau est la suivante :

Définition Un nœud préfère établir de nouvelles connexions avec des nœuds de degré similaire, favorisant l'agglomération de sommets et l'apparition de *hubs*, des regroupements de sommets de très hauts degrés.

L'assortativité est liée au phénomène empirique, fréquemment constaté dans les réseaux sociaux et de communication par exemple, dit du « club des riches » (*rich club*) (Colizza et al., 2006). Il réfère à la propension qu'ont les sommets de très hauts degrés d'un graphe de terrain à être très liés entre eux. L'étude des hubs contenus dans un réseau révèle des caractéristiques interprétables selon sa nature : présence d'un groupe d'influenceurs à fort impact pour un réseau social, bonne résilience et tolérance aux pannes pour un réseau de communication etc. (McAuley et al., 2007)

Un phénomène lié est l'attachement préférentiel, par lequel on distribue une quantité de ressources donnée à un nombre fini d'entités en fonction de la quantité de cette ressource qu'elles possèdent déjà, processus prosaïquement dénommé « the rich get richer ». Il a été décrit par Yule (1925) en systématique (biologie), par Merton (1968) sous le nom de Matthew Effect et Price (1976), en sociologie, en économie et dans bien d'autres domaines, faisant intervenir des êtres vivants la plupart du temps (Rigney, 2010).

Cet attachement produit la distribution des degrés en loi de puissance, dans les réseaux dits sans échelle. Il a été mesuré dans les réseaux sociaux et popularisé par l'article de Barabási et Albert (1999), qui ont proposé le terme, ainsi qu'un modèle de formation de réseau suivant ce principe, basé sur l'étude de la croissance de l'Internet, mais également constaté dans les réseaux sociaux et d'autres types de réseau (voir section 1.2.3.4, p. 17).

La combinaison de l'assortativité et de l'attachement préférentiel crée, dans les réseaux concernés, des formes en « grappes » agglomérées, où les sommets ont des voisins de degrés essentiellement similaires dont quelques uns de degrés nettement plus élevés, et où les sommets de très hauts degrés sont tous liés entre eux. Ces réseaux sont donc petit-monde et sans échelle, deux caractéristiques fréquemment (mais pas toujours, voir section 1.2.4 ci-dessous) combinées.

1.2.4 Universalité des propriétés

Des travaux précurseurs, tels que ceux deFaloutsos et al. (1999) concernant le world-wide-web, appuyés par ceux de Barabási et Albert (1999), ont suggéré que les propriétés précédentes sont largement répandues dans de nombreux types de graphes de terrain.

Toutefois, des études récentes (Clauset et al., 2009, Labatut, 2014) montrent qu'il faut en réalité questionner les postulats fréquemment admis que les graphes de terrain sont petit-monde ou sans-échelle. Cela peut être vrai pour certains types de réseaux (par exemple, les réseaux sociaux) mais ne saurait décrire universellement tous les graphes de terrain. L'échelle à laquelle les graphes sont analysés est également cruciale, en particulier lorsque les paramètres étudiés en sont dépendants, comme la distribution des degrés. Il a ainsi été observé que certains graphes présentés par Barabási et Albert (1999) dans leur article ne semblent pas être sans échelle lorsque l'on considère une échelle étendue des valeurs de $P(k)$ plus large que celle présentée. De récents travaux (Shalizi & Thomas, 2011, Han et al., 2014b) ont également montré que l'homophilie, que l'on pensait très répandue dans les réseaux sociaux, est en réalité beaucoup moins présente ou possiblement confondue avec d'autres phénomènes.

On peut prouver que les graphes suivant un même modèle de génération, et donc modélisant un même type de réalité comme des interactions entre protéines, possèdent les mêmes caractéristiques. On peut également prouver que la présence combinée de plusieurs de ces propriétés en caractérise d'autres, comme l'ont fait Seshadhri et al. (2012) pour l'invariance d'échelle. Souvent, ces propriétés sont partielles ou moins fortes que les « grandes » propriétés décrites précédemment. A contrario, une étude minutieuse doit être réalisée pour les graphes de terrain dont les mécanismes de formation sont peu ou mal connus.

Ainsi, les réseaux de transports urbains ferrés (métro, tram, train urbain), bien qu'étant également des graphes de terrain, ont généralement un coefficient de clustering local faible et ne sont pas sans échelle : construits afin d'assurer une desserte uniforme du territoire urbain qu'ils couvrent et d'y faciliter l'accès aux transports dans toutes les zones géographiques, les gares sont réparties, étalées sur ce territoire. La densité plus faible peut aussi être expliquée par des contraintes spatiales influençant le choix de l'emplacement des gares et le tracé des lignes de chemins de fer, ainsi que le graphe résultant.

1.2.5 Modèles

L'absence d'informations ou de connaissance sur les processus qui sont à l'origine des graphes de terrain observés rend leur étude difficile. Etant donné l'existence, même discutée, de propriétés communes, une approche classique consiste à proposer des modèles permettant de générer artificiellement des graphes ayant des caractéristiques identiques. Cette section décrit quelques uns de ces modèles, en distinguant l'approche aléatoire de l'approche par attachement préférentiel.

Graphes aléatoires

Un graphe aléatoire est, comme son nom l'indique, un graphe généré par un processus aléatoire. En ce sens, la relation des graphes de terrain avec les graphes aléatoires est ambiguë. Si, sémantiquement, le concept de graphe *de terrain* existe afin de le dissocier des graphes générés purement artificiellement, ils peuvent être approchés par des modèles mathématiques de génération (Albert & Barabási, 2002). En effet, la plupart des réseaux réels sont formés par un processus non-déterministe, qui admet de l'aléatoire, mais souvent très compliqué d'où le qualificatif de *complexe*. Les modèles proposés pour expliquer la formation des graphes de terrain sont ainsi nécessairement spécifiques, réducteurs et simplificateurs. L'étude des graphes de terrain a néanmoins largement recours à ces modèles afin de mener des études, simulations, et évaluations d'algorithmes.

Erdős-Rényi

Nous commençons par présenter les modèles d'Erdös et Rényi (1959), introduisant les graphes aléatoires. Bien que trop simples pour permettre de modéliser des systèmes complexes, ils sont à la base de l'étude des propriétés des graphes.

Le premier modèle prend en entrée un graphe G dont on connaît l'ensemble des sommets V de cardinal n dont l'ensemble des arêtes est initialement vide, $E = \emptyset$. Ce graphe peut contenir au maximum $m_{\max} = \frac{n(n-1)}{2}$ arêtes. On se munit d'une probabilité p, chacune des $m_{\max}$ arêtes potentielles étant effectivement ajoutée à E avec une probabilité p.

La présence d'une arête constitue une variable de Bernoulli indépendante. Si $p = 1$, G est un graphe complet. Le nombre d'arêtes m suit une loi binomiale de coefficient $\mathrm{C}^{2p}_{m_{\max}}$ et la distribution des degrés suit une loi de Poisson lorsque n tend vers l'infini, ils ne sont donc pas sans échelle.

Le second modèle, lié au premier, consiste à choisir un graphe de façon aléatoire uniforme parmi les graphes ayant un nombre de sommets n et d'arêtes m donnés. D'autres variantes existent, utilisant des lois de probabilités plus complexes (Bollobás, 2001).

Malgré leur simplicité, les modèles d'Erdős-Rényi ont permis la réalisation de nombreux travaux, notamment sur la connexité et les composantes connexes géantes, qui ont plus tard été adaptés aux graphes de terrain.

Watts-Strogatz

Le modèle de Watts et Strogatz (1998) a été introduit afin de modéliser l'effet petit-monde. Cependant, les graphes formés selon ce modèle ne montrent pas de fort coefficient de clustering local et ne sont pas sans-échelle (leur distribution converge également vers une loi de Poisson, comme Erdős-Rényi).

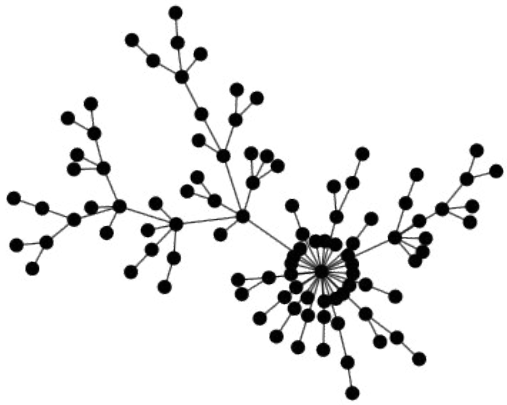


FIGURE 1.3 – Graphe d'attachement préférentiel généré par Graphstream (Dutot et al., 2007), tiré du site `http://graphstream-project.org/`.

On fixe d'abord un nombre de sommets n, un degré moyen k pair et une probabilité β tels que $1 \ll \ln(n) \ll k \ll n$, qui correspondent à des valeurs fréquemment observées dans les réseaux sociaux par exemple.

On construit ensuite une grille régulière où chacun des n sommets est connecté à $\frac{k}{2}$ voisins, répartis de chacun de ses côtés dans la grille.

Ensuite, pour chaque couple de sommets $v_i, v_j \in V$, chaque arête $(v_i, v_j), i < j$ est remplacée avec une probabilité β par une arête $(v_i, v_{j'}), j' \neq j$ n'existant pas déjà.

Barabási-Albert

Le processus de formation proposé par Barabási et Albert (1999) est lié à la présence de phénomènes tels que l'assortativité et l'attachement préférentiel (cf. section 1.2.3.6, p. 19).

On considère un graphe initial $G_0 = (V_0, E_0)$ de taille initiale $n_0 = |V_0|$. On ajoute ensuite des sommets tour à tour. A l'itération i (la taille du graphe est $n_i = |V_i|, n_i > n_0$ et $m_i = |E_i|$), un nouveau sommet est ajouté et connecté à $n_i \leq n_0$ sommets existants, où chaque sommet $v \in V$ est choisi avec la probabilité p_v suivante :

$$p_v = \frac{d_v}{\sum_{u \in V} d_u} = \frac{d_v}{2m_i}$$

La distribution des degrés est bien en loi de puissance, avec $P(d_v = k) \sim k^{-3}$

Dorogovtsev et Mendes (2002) ont proposé une variante produisant également des graphes sans échelle, avec un paramètre d'échelle différent.

1.2.6 Algorithmique des graphes de terrain

De nombreuses tâches d'analyse des graphes de terrain suscitent des travaux de recherche : compression, visualisation (Auber et al., 2007, Jünger & Mutzel, 2012) ou détection de communautés (voir section 1.3, p. 23) dont il est question dans la thèse.

Les propriétés spécifiques des graphes de terrain sont importantes et utiles à prendre en compte lors de la conception d'un algorithme, car elles confèrent souvent de grands avantages, comme la réduction de la complexité moyenne (Fortunato, 2009). Longtemps soupçonnées car constatées empiriquement, elles commencent à être prouvées (Brach et al., 2016), ouvrant la voie à un type d'algorithmique les exploitant.

L'avantage est bien sûr d'offrir des vitesses d'exécution (très) élevées, garanties pour des graphes possédant certaines caractéristiques données, mais incertaines pour d'autres types de graphe. Ainsi, la détection de communauté est un problème NP-complet (Newman, 2003), mais la complexité temporelle d'algorithmes de détection P et même NP atteint très rarement le pire cas théorique. C'est le cas de l'algorithme de Louvain, qui s'avère très rapide alors que sa complexité théorique est élevée : dans le pire des cas, il faut réaliser $\mathcal{O}(m^3)$ tests lors de la phase de déplacement/regroupement des sommets (Guillaume, 2012).

Les algorithmes que nous proposons, VOLCAN et LOCNeSs, décrits dans les chapitres 3, p. 66 et 4, p. 88 respectivement, s'incrivent dans ce type d'algorithmique : l'existence des voisinages de sommets qu'ils recherchent est prouvée dans des graphes possédant les conditions de Gleich et Seshadhri (2012) (cf. section 2.5.1.2, p. 57), mais est non garantie dans d'autres types de graphe.

1.3 La tâche de détection de communautés

Une caractéristique présentée en introduction et liée aux propriétés précédemment énoncées est la présence de *communautés* dans les graphes de terrain. La notion de communautés est complexe et aucune définition universelle n'est reconnue (Fortunato, 2009). Néanmoins, il est possible de décrire la notion intuitive de communautés, ainsi que la tâche de détection qui est au cœur de la thèse et sera reprise, détaillée, formalisée et discutée dans le chapitre 2.

Nous présentons ici quelques éléments introductifs, en premier lieu des définitions puis un positionnement par rapport à la fouille de graphe. Nous exposons des motivations possibles pour réaliser cette tâche d'analyse sur un graphe, ainsi que différentes manières d'évaluer la qualité des résultats produits.

1.3.1 Définitions

La plupart des grands réseaux d'interaction (mais pas tous) sont globalement peu denses, cependant leur densité n'est pas uniforme. Ils ont une structure *modulaire* : le

graphe de terrain qui leur est lié est composé de *sous-graphes localement denses* (voir section 1.2.3.3, p. 16). Les réseaux sociaux, biologiques sont souvent très modulaires, alors que les réseaux de transport urbain (métro par exemple), ne le sont que peu voire pas du tout (cf. section 1.2.4, p. 20).

De façon intuitive, on appelle *communautés* des sous-graphes dont la densité interne (liens entre leurs sommets) est très supérieure à la densité externe (liens avec le reste du graphe).

Un ensemble de communautés $C = \{c_1, c_2..., c_k\}$ pour le graphe G est un ensemble de k sous-graphes tels que $\forall i \in [\![1, k]\!], c_i \subset G$ et

$$\bigcup_{i=1}^{k} c_i = G$$

Illustrons cette définition informelle par un exemple : considérons le réseau social évoqué dans l'exemple de la section 1.2.2, p. 14 : les chercheurs collaborent entre eux afin d'effectuer des travaux et publier des résultats. On peut définir qu'un ensemble de co-auteurs réguliers de publications forme une communauté. De même, les chercheurs sont généralement répartis en équipes ou départements et chaque équipe peut également définir une communauté (aspect multi-niveaux, hiérarchique, des communautés).

Cependant, il faut faire attention aux limitations de la considération du seul graphe du laboratoire, en effet certains chercheurs collaborent peut-être uniquement avec des chercheurs appartenant à d'autres laboratoires, ainsi ils semblent n'appartenir à aucune communauté de co-auteurs au sein de leur laboratoire. Cette appartenance existe bien, mais n'est pas décelable car non contenue dans les données considérées (les communautés dans les réseaux de co-auteurs sont d'ailleurs constituées indépendamment de leur institution de rattachement). Ainsi, une partition en communautés n'est utile qu'à la lumière d'une interprétation minutieuse, réalisée par un expert dans le champ scientifique d'où les données considérées proviennent par exemple.

Clique et quasi-clique

On pourrait vouloir associer les communautés à des cliques (cf. section 1.1.2) dont la densité est 1, cependant la notion de clique est trop stricte pour correspondre à la réalité des communautés rencontrées dans les graphes de terrain (Fortunato, 2009, p. 10). La notion de *quasi-clique* est souvent utilisée à la place : un sous-graphe dont la densité est forte mais qui n'est pas une clique (Matsuda et al., 1999). Formellement, étant donné $\gamma \in [0, 1]$ on définit une γ-quasi-clique comme un sous-graphe non vide $H = (V_H, E_H), H \subset G$ tel que, en notant $q = |V_H|$:

$$|E_H| \geq \gamma \frac{q(q-1)}{2}$$

En d'autres d'autres, une γ-quasi-clique est un sous-graphe possédant un pourcentage γ des arêtes d'une clique de même taille.

Sens fort et sens faible

Radicchi et al. (2004) ont proposé une distinction entre communautés *au sens fort* et *au sens faible*. Considérons un sous-graphe $H \subset G$, ainsi qu'un sommet $v \in V(H)$. Notons $d_v^{in}(H)$ le nombre d'arêtes reliant le sommet v à d'autres sommets appartenant à H et $d_v^{out}(H)$ le nombre de sommets reliant v à des sommets n'appartenant pas à H. Radicchi et al. donnent alors les définitions suivantes :

Définition Le sous-graphe H est une *communauté au sens fort* si

$$\forall v \in V(H), d_v^{in}(H) > d_v^{out}(H)$$

Définition Le sous-graphe H est une *communauté au sens faible* si

$$\sum_{v \in V(H)} d_v^{in}(H) > \sum_{v \in V(H)} d_v^{out}(H)$$

Dans une communauté *forte*, chacun des sommets a plus de liens dans sa communauté que vers le reste du graphe, alors que dans une communauté *faible*, le total du nombre de liens dans cette communauté est supérieur à celui du total des liens vers le reste du graphe. Il est évident qu'une communauté forte est également une communauté faible, mais que la réciproque n'est pas vraie.

De nombreuses autres définitions de communautés ont été proposées (Wasserman & Faust, 1994, Fortunato, 2009) mais elles se rapprochent généralement de l'une ou l'autre des définitions présentées ci-dessus.

1.3.2 Détection de communautés et fouille de données

La recherche des sous-graphes constituant les communautés est une tâche particulière de fouille de graphe, nous discutons dans cette section du lien entre détection de communautés et fouille de données. Nous commençons par faire un point sur le vocabulaire utilisé puis sur les défis communs auxquels les deux tâches font face. Nous évoquons finalement les aspects communs aux différentes méthodes.

Vocabulaire

Le terme de *communauté(s)*, de par son étymologie (du latin *communis*, commun, lui-même de *munus*, relatif à quelque chose de public) et son origine (réunion ou association de personnes ayant des intérêts communs), revêt un sens connoté « réseaux sociaux », on peut même dire qu'il a directement été importé des sciences sociales, où il est utilisé depuis

longtemps (Moreno & Jennings, 1938). On le rencontre d'ailleurs dans les travaux de Mark Newman et ses collaborateurs (Girvan & Newman, 2002) sur la détection algorithmique de communautés sociologiques dans des réseaux de co-auteurs, qui ont fortement contribué à populariser la discipline au début des années 2000.

Néanmoins, il est maintenant utilisé de manière générique pour désigner les ensembles identifiés dans n'importe quel graphe de terrain (Fortunato & Hric, 2016). Certains auteurs utilisent le terme *clusters*, qui est hérité de la fouille de données, et est de ce fait sémantiquement plus neutre.

De même, l'expression *détection de communautés* (*community detection*[9]) semble être principalement utilisée pour désigner la recherche de sous-ensembles denses dans les graphes de terrain, alors que certains auteurs lui préfèrent des termes plus neutres dont le sens est lié à la fouille de données : *network/graph clustering* ou *partitioning* (Schaeffer, 2007,Buluç et al., 2016) et *(stochastic) blockmodelling* (Karrer & Newman, 2011) par exemple. Ces mots-clés référencent la quasi-totalité des publications sur le sujet, mais il est à noter que *graph clustering* et ses dérivés liés à la fouille de données étant antérieurs à *community detection*, les articles les plus anciens sont référencés uniquement par ces termes. Une partie de la communauté scientifique dont l'existence est antérieure à la « vague » ayant porté la détection de communauté depuis le début des années 2000, principalement liée à l'analyse de graphes et à la combinatoire, n'utilise pas les termes *community détection*. De façon analogue, les ensembles trouvés par les algorithmes sont dénommés *clusters* ou *communities* selon la terminologie choisie.

Enfin, on remarque que le terme de *détection* est employé de manière abusive par rapport à son sens premier. Concrètement, il renvoie à la nature non supervisée de la tâche : on applique un algorithme à un graphe dans lequel des communautés peuvent être trouvées ou non. Ce terme convient donc bien à la tâche telle qu'elle existe dans les domaines spécifiques, en sociologie par exemple, d'où il provient : on peut constater la présence ou l'absence de communautés.

Cependant, il est moins bien adapté à la réalité de la détection algorithmique sur graphe. De façon pragmatique, de nombreux algorithmes opèrent un partitionnement ou une décomposition du graphe : en ce sens un ensemble de communautés est *toujours* renvoyé par l'algorithme, charge à l'expert analysant les résultats d'évaluer l'existence réelle et la pertinence de ces communautés (cf. section 1.3.4, p. 29). Il se peut très bien que ces résultats ne contiennent aucune communauté, au sens de la définition attendue par l'expert.

Ainsi, la détection de communautés peut être considérée comme une tâche de clustering des nœuds d'un graphe incluant des contraintes supplémentaires, principalement liées à la sémantique des clusters trouvés.

9. Anecdotiquement, le curieux *community clustering* est aussi parfois rencontré.

Défis

Les défis rencontrés face à la détection de communautés (Fortunato & Hric, 2016) sont pour partie communs à ceux relatifs à la fouille de données et ont trait à :

— la taille des données. En effet, les graphes de terrain étudiés, en particulier les réseaux sociaux et biologiques, dépassent maintenant fréquemment le million de sommets et parfois même le milliard.

— la nature des données. La nature des différentes situations modélisées par les graphes de terrain, ainsi que leur interprétation, sont des informations très importantes à prendre en compte pour détecter des communautés pertinentes, puis les analyser, avec l'aide d'un expert.

— la complexité des données. Bien que l'analyse d'un graphe puisse être relativement simple du fait du faible nombre de caractéristiques considérées (au minimum, sommets et arêtes), plusieurs découpages en communautés peuvent être sémantiquement valables pour un même graphe. Cette problématique est ouverte : elle nécessite une connaissance extérieure, comme pour la nature évoquée au point précédent, afin d'aider à trancher les différents partitionnements possibles, voire optimiser le temps de recherche. Il est à noter que la structure du graphe peut également être enrichie avec des attributs, une variation au cours du temps, ou tout simplement des modèles plus riches tels que les graphes multiplexes, complexifiant là encore la tâche.

Méthodes

Un graphe pouvant également être représenté sous forme matricielle (adjacence, incidence, cf. section 1.1.1, p. 6), ou d'une collection de vecteurs, pour les graphes attribués par exemple, on peut utiliser de nombreuses méthodes de partitionnement de données existantes (voir section 2.2, p. 41).

Néanmoins, certains critères classiques utilisés pour mesurer la qualité d'un cluster, tels que l'homogénéité ou la pureté, ne sont pas toujours adaptés pour mesurer la qualité d'une communauté, a fortiori dans des cas particuliers tels que les communautés recouvrantes, qui sont évoquées dans le chapitre 4, ou hiérarchiques (Almeida et al., 2011). Ainsi, des méthodes de partitionnement classiques peuvent se révéler inadaptées.

D'autre part, les graphes de terrain étant peu denses leurs représentations matricielles, liées à leurs arêtes, sont très creuses. Ainsi, les méthodes classiques de clustering utilisant des matrices se révèlent souvent algorithmiquement coûteuses (Newman, 2003, Fortunato, 2009), particulièrement sur de très grands graphes comme nous l'avons souligné dans le paragraphe précédent. Fortunato (2009) estime que la détection de communauté est, dans la plupart des cas, une tâche NP-complète.

Enfin, il est très important de noter que plusieurs méthodes appliquées à un même graphe peuvent donner plusieurs résultats différents, tous pertinents selon un (ou plusieurs)

aspect(s) donné(s). Le choix de la méthode à appliquer est donc conditionné par les facteurs cités précédemment, ainsi que de nombreux autres (Schaub et al., 2017). De même que pour le clustering, il est généralement admis qu'aucune méthode ne satisfera tous les critères attendus, en terme de performance (rapidité, passage à l'échelle...) que de résultats (qualité structurelle, pertinence... des communautés) (Kleinberg, 2003).

La complexité de la tâche, NP-complète (Fortunato, 2009), nécessite donc le développement de méthodes spécifiques. On distingue généralement deux manières de procéder :

— en donnant une définition de ces communautés et en concevant un algorithme pour les trouver dans G
— en concevant un algorithme utilisant des mécanismes ou propriétés du graphe pour cibler des ensembles qui sont ensuite interprétés comme des communautés.

Des caractéristiques plus précises, présentées dans le chapitre suivant section 2.3, p. 45, permettent de décrire et organiser les différentes méthodes et familles de méthode de détection.

1.3.3 Motivations

De même que pour le clustering de données en général, trouver la structure communautaire d'un réseau est bien évidemment d'abord utile pour le comprendre, en extraire des informations (Newman, 2012). Nous citons ici trois exemples de motivations à la base de la détection de communautés.

Aider à l'analyse d'un réseau

Détecter des communauté peut permettre, par exemple, d'isoler des zones du graphe à l'origine de fonctions importantes du réseau associé. Ainsi Sueur et al. (2011) recherchent des groupes sociaux au sein d'une population de primates afin de mieux comprendre leur comportement. Cela peut également permettre de découvrir des informations enfouies. Par exemple, la recherche visant à identifier des terroristes sur les réseaux sociaux est très active (Krebs, 2002, Ouellet, 2016). Les graphes de terrains visés étant généralement très grands, la taille des données étant un défi cité précédemment, il est n'est pas possible de réaliser l'analyse de tels réseaux « à la main ».

Etablir des modèles de prédiction

Appréhender les caractéristiques des réseaux permet d'en construire des modèles prédictifs d'évolution (prédiction de sommets ou de liens (Lü & Zhou, 2011)). Ces modèles sont très demandés dans le cadre d'applications commerciales par exemple : caractérisation de profils (d'acheteurs, de visiteurs, de clients...). Différents exemples d'application sont cités section 1.2.2, p. 14.

Spécifiques

Bien évidemment, parmi les nombreux domaines de recherches utilisant l'analyse des réseaux complexes, certaines applications ont des besoins spécifiques en terme d'algorithmique (cf. section 1.2.2, p. 14).

Par exemple, cette thèse s'attache à présenter des méthodes permettant de détecter des communautés opérant sur des réseaux mobiles ad-hoc spécifiques, les OMSN, décrits dans la sous-section 1.4.4, p. 33.

1.3.4 Evaluation

L'évaluation de la qualité des communautés est un point clé de la détection, lié à la complexité et la nature du graphe considéré, évoquées précédemment. Parmi les différents découpages possibles d'un graphe en communautés, il faut pouvoir identifier les plus sensés et pertinents. Pour cela, différentes méthodologies sont utilisées (Danon et al., 2005), parmi lesquelles :

— l'appel à la connaissance d'un expert du domaine pour évaluer « à la main » un échantillon des communautés trouvées (l'échantillonnage étant une problématique en soi (Leskovec & Faloutsos, 2006))
— l'utilisation de métriques explorant les caractéristiques intrinsèques des communautés (taille, répartition des degrés...) (Almeida et al., 2011)
— la comparaison avec un modèle de référence (Yang & Leskovec, 2015). La majorité des algorithmes de détections étant *non-supervisés*, c'est-à-dire qu'ils identifient les communautés dans un graphe qu'ils analysent pour la première fois, si une partitions en communautés de référence existent (les données sont *annotées*), on peut les comparer aux communautés trouvées. Ces données annotées sont souvent désignées sous le nom de *vérité terrain* (*ground truth* en anglais), bien que le terme ne fasse pas consensus, en particulier quand ces annotations n'ont pas été réalisées directement « sur le terrain ».

On note que les algorithmes par optimisation de critère fournissent des communautés optimales pour le graphe donné, au sens du critère utilisé.

Un point très important car il fait l'objet de nombreuses discussions et de nombreux travaux dans la communauté scientifique de la détection de communautés sur graphe porte sur l'adéquation entre les modèles de communautés (que l'on recherche ensuite sur graphe), les communautés optimales selon une fonction objectif, et autres communautés théoriques d'une part, et la réalité des communautés dans les graphes de terrain d'autre part. De nombreux auteurs mettent en garde contre la trop grande confiance accordée aux communautés théoriques par rapport aux formes très diversifiées des communautés rencontrées dans les graphes de terrain (Hric et al., 2014, Yang & Leskovec, 2015).

1.4 Contexte applicatif : routage dans des réseaux mobiles opportunistes sociaux

Dans cette dernière section, nous abordons la notion du « terrain » des graphes de terrain pour lesquels les algorithmes proposés dans cette thèse ont été conçus. Une connaissance du contexte d'application particulier envisagé est en effet utile afin de mieux comprendre la motivation de nos travaux.

Les réseaux considérés sont des réseaux mobiles apparus récemment combinant de nombreuses caractéristiques que nous décrivons dans une première sous-section, avant d'exposer les défis rencontrés dont nous détaillons un point, le routage, ainsi qu'un exemple de leur exploitation. Nous discutons enfin de la faisabilité et de l'intérêt de la détection de communautés dans un tel réseau.

1.4.1 Un cas (très) particulier de graphe de terrain

Les algorithmes présentés dans cette thèse sont destinés à exploiter des graphes de terrain d'une catégorie particulière. Ces graphes sont induits par des réseaux mobiles opportunistes sociaux (Hui et al., 2005, Lee et al., 2013, Conti & Giordano, 2014), qui ont la particularité d'être des réseaux *dynamiques* (car constitués d'entités mobiles sans-fil), *ad-hoc*, *opportunistes*, mais dont la structure hérite de celle d'un réseau social.

Définitions

Nous commençons par définir les termes introduits concernant les réseaux mobiles : ad-hoc et opportuniste.

Définition Un *réseau mobile ad-hoc*, désigné par son acronyme anglais *MANET* (Mobile Ad-hoc NETwork), est un réseau constitué d'entités mobiles, sous-entendues communiquant entre elles sans-fil la plupart du temps, sans infrastructure propre, s'auto-configurant ainsi de manière continue et autonome (Macker & Corson, 1998). Cela signifie que les échanges entre entités s'effectuent en pair-à-pair, ce dont doit tenir compte le routage du trafic. Le nom MANET provient d'un groupe de travail de l'Internet Engineering Task Force [10] chargé de standardiser des protocoles de routage basés sur IP, afin de pouvoir réaliser du routage sans-fil en tenant spécifiquement compte de la dynamique des nœuds du réseau.

Définition Un *réseau mobile opportuniste* est un réseau ad-hoc formé automatiquement dès que les entités sont à portée de communication les unes des autres.

10. `https://datatracker.ietf.org/wg/manet/charter/`

Exemple de réseau mobile opportuniste

Un exemple type d'un tel réseau est un réseau de téléphones mobiles communicant sans fil en pair-à-pair : ces téléphones sont des dispositifs portés par des personnes et la structure du réseau dépend de la position géographique de ces personnes (porteurs), rendant possibles ou non les communications. Un réseau social sous-jacent (relations entre les porteurs), influençant le fait qu'elles puissent se trouver au même endroit à un instant donné, permet donc d'anticiper cette structure.

Le réseau mobile opportuniste formé hérite donc plus ou moins directement de la structure du réseau social et des caractéristiques qui s'y rapportent. Un tel réseau est dénommé Opportunistic Mobile Social Network (OMSN), mais il a également été étudié sous d'autres noms, par exemple PSN (Pocket Switched Network) par Hui et al. (2005), ou encore SSN (Spontaneous Smartphone Network) par Aloi et al. (2014). Ses principales caractéristiques sont :

— *mobile et sans-fil* : ce réseau est donc mobile, les dispositifs évoluant dans l'espace conjointement à leurs porteurs. Ils communiquent entre eux sans fil (Wi-Fi direct, Bluetooth...)
— *ad-hoc* : ce réseau est formé spontanément, sans architecture centralisée. Il est généralement à sauts multiples (les paquets passent d'intermédiaire en intermédiaire), bien que dans de rares cas où tous les porteurs sont proches géographiquement, des sauts uniques soient possibles.
— *tolérant aux délais* : les caractéristiques ci-dessus impliquent de potentielles interférences ou ruptures de communications fréquentes entre dispositifs, dont résulte une latence moyenne élevée.
— *topologie mesh* : de même, les caractéristiques précédentes font que ces réseaux adoptent le plus souvent une topologie mesh (cf. topologie régulière, section 1.1.5, p. 12), la plus efficace pour réduire la latence et les ruptures de communication.

1.4.2 Défis

Les défis concernant les OMSN sont similaires aux principaux défis relatifs aux MANETs et aux réseaux tolérants aux délais (Cao & Sun, 2013, Conti & Giordano, 2014).

En premier lieu figure le *routage*, c'est-à-dire la transmission et l'orientation des paquets réseau afin qu'ils parviennent de leur origine à leur destination. Dans un OMSN, il doit tenir compte de la topologie mesh, la latence, la rupture de routes... En particulier, la nature ad-hoc et tolérante aux délais impose de multiples contraintes. Un routage naïf, par exemple par inondation (*flooding*), se heurte à de nombreux inconvénients (Tseng et al., 2002) liés au coût des nombreux messages échangés : grande utilisation de la bande passante et grand nombre de messages (souvent redondants) reçus par les nœuds. De ces deux inconvénients dérivent de multiples autres : recevoir beaucoup de messages

a un coût énergétique non négligeable pour un dispositif mobile, des messages peuvent circuler très longtemps après avoir atteint leur cible –leur durée de vie (*TTL*, Time To Live) étant élevée pour respecter la tolérance aux délais–, les performances du réseau se trouvent donc dégradées par rapport à leur potentiel. De plus, l'évolution du réseau liée aux déplacements des porteurs implique non seulement de possibles ruptures de communication, mais également une déconnexion possible d'une partie du réseau. Ainsi, un chemin de bout-en-bout entre deux nœuds peut très bien, d'un instant à l'autre, non seulement changer mais tout simplement cesser d'exister. La livraison d'un message n'est donc jamais garantie à tout instant.

La *sécurité* est également souvent citée, en effet les protocoles de sécurisation des échanges actuels s'appliquent plus difficilement aux MANETs, quand ils ne sont pas tout simplement incompatibles (Di Pietro & Domingo-Ferrer, 2013).

1.4.3 Exemple réel de réseau OMSN : FireChat

Parmi différentes applications existantes des OMSN dans le monde réel (Lepp, 2015), nous choisissons de présenter ici un exemple qui a beaucoup fait parler de lui dans les média : FireChat [11].

FireChat est une application pour équipement numérique mobile, type smartphone/tablette, permettant à ses utilisateurs de communiquer localement en échangeant des messages textuels de pair à pair. Les dispositifs à portée de communication, via sans-fil Wifi ou Bluetooth, forment un réseau ad-hoc décentralisé *mesh*. Chaque utilisateur connecté à ce réseau peut échanger des messages avec n'importe quel autre utilisateur également connecté, sans passer par une infrastructure centralisée, a fortiori un opérateur téléphone : c'est donc bien un MANET.

Les principaux avantages de ce type d'application sont évidemment la possibilité de s'affranchir d'un opérateur téléphonique, pour des raisons de coût, mais surtout confidentialité et de disponibilité vis-à-vis de l'infrastructure d'un opérateur.

Ainsi, l'application FireChat a été très utilisée en Irak en 2014, lorsque le gouvernement irakien a restreint l'accès à Internet (Hern, 2014), permettant de contourner cette restriction de la disponibilité d'accès à un réseau global. Elle a également été utilisée à Hong-Kong, en 2014 toujours, lors des manifestations contre le pouvoir chinois (Bland, 2014), qui contrôle et censure la plupart des communication électroniques. Les protestataires pouvaient ainsi bénéficier d'une meilleure confidentialité de leurs échanges électroniques.

Des applications du même type sont également utilisées en cas de catastrophe naturelle mettant à mal le fonctionnement infrastructures de communication traditionnelles (téléphone fixe, mobile, Internet), telles que des séismes, typhons et tsunami. Elles permettent aux personnes des zones sinistrées de communiquer entre elles afin de s'assurer qu'elles sont indemnes, de prévenir les secours, rechercher les personnes disparues... alors que la

11. `https://www.opengarden.com/firechat.html`

circulation dans ces zones est très difficile voire impossible. Ces tâches sont cruciales en particulier dans les premières heures suivant la survenue de la catastrophe.

Les principaux inconvénients sont les inconvénients généraux des réseaux ad-hoc décentralisés, en premier lieu le routage en pair à pair dans un réseau dynamique et très volatile (voir ci-dessus 1.4.2, p. 31). D'autres inconvénients sont liés aux dispositifs mobiles et technologies utilisées : portée limitée et haute consommation d'énergie du Bluetooth et du Wi-fi Direct par exemple.

1.4.4 Un cas (très) particulier de détection de communautés

Tel que nous l'avons décrit précédemment, le routage est une tâche difficile dans les réseaux mobiles ad-hoc. Une des motivations de nos travaux est d'utiliser la caractéristique sociale des OMSN afin d'améliorer ce routage. En effet, différentes études ont montré que les propriétés de l'organisation sociale humaine pouvaient être mises à profit à cet effet (Chaintreau et al., 2005, Chaintreau et al., 2007, Costa et al., 2008, Karamshuk et al., 2011).

Cette organisation humaine prend différentes formes et connaître les zones denses où se regroupent les personnes, où tous les dispositifs mobiles se donc retrouvent à portée, facilite l'échange des paquets réseaux. Ces regroupements denses correspondent dans le graphe afférent à des communautés. Il existe également des algorithmes utilisant d'autres propriétés, tel que SimBet avec la centralité d'intermédiarité (Daly & Haahr, 2007).

Les premiers travaux utilisant la détection de communautés afin d'améliorer le routage ont été proposés par Hui et al. (2011) avec le protocole BUBBLE Rap. Le principe est de transférer un paquet réseau vers un sommet de chaque communauté connue en s'aidant de mesures de centralité, jusqu'à ce qu'un sommet de la communauté du destinataire soit atteint. Ce sommet transmet alors le paquet aux aux membres de la communauté jusqu'à ce que le destinataire soit atteint.

La méthode de détection de communautés utilisée est adaptée à partir de travaux réalisés sur les réseaux tolérants aux délais (une sur-classe des MANETs) (Hui et al., 2007).

Les expériences menées montrent des gains de performances significatifs, diminuant la surcharge du réseau (en terme « d'inondation » de paquets) par rapport à des méthodes existantes, tout en gardant un taux d'acheminement de paquets satisfaisant (Orlinski & Filer, 2013).

Ces travaux ont été poursuivis afin de proposer des méthodes de détection dynamiques, capables de produire une structure communautaire en temps réel (Chan et al., 2009).

1.5 Bilan

Nous avons, dans ce chapitre, passé en revue des notions élémentaires de théorie des graphes utiles pour comprendre le reste du manuscrit. Nous avons présenté un type de graphes particulier, les graphes de terrain, pour lesquels nous avons introduit des définitions, des modèles ainsi que des exemples.

Nous avons également introduit une tâche particulière pour l'exploitation de ces graphes de terrain : la détection de communautés. Cette tâche est au centre des travaux présentés dans la thèse.

Nous avons finalement présenté un cas d'application de détection de communautés lié à la thèse : le routage dans des réseaux mobiles opportunistes.

La prochaine partie traite de la détection de communautés sur graphe statique, en proposant un état de l'art des méthodes de détection existant et en présentant deux contributions : les algorithmes VOLCAN et LOCNeSs, réalisant une détection de communautés, respectivement disjointes et recouvrantes, sur un graphe statique.

Première partie

Détection statique de communautés orientée-sommet

Introduction

Nous avons vu dans les chapitres introductifs précédents ce qu'est la tâche de détection de communautés dans des graphes, les difficultés qu'elle représente, et le contexte particulier dans lequel nous souhaitons l'appliquer : les réseaux mobiles décentralisés sociaux.

Cette première partie considère le cas de référence défini par des graphes statiques, c'est-à-dire entièrement déterminés par un ensemble de sommets et d'arêtes, tous deux fixes et constants.

Le chapitre 2 dresse un état de l'art des méthodes existantes, en en distinguant cinq familles principales qui illustrent, de par leur variété, la complexité et la diversité d'interprétation de la tâche. Il détaille en particulier le cas des méthodes orientées-sommet, cadre dans lequel nos propositions s'inscrivent.

Le chapitre 3 présente l'algorithme que nous avons proposé, nommé VOLCAN, en mettant l'accent sur les principes généraux qui structurent l'intégralité de nos propositions, pour ce cas des communautés statiques disjointes comme pour les suivants, présentés dans le chapitre 4 et la partie II.

Le chapitre 4 s'intéresse au cas étendu des communautés recouvrantes, c'est-à-dire qui s'affranchissent de la contrainte d'intersection vide des communautés, autorisant des sommets à appartenir simultanément à plusieurs d'entre elles : il correspond à un modèle plus réaliste des communautés présentes dans le monde réel, en particulier pour les réseaux sociaux dont les OMSN que nous considérons dans la thèse. Il présente l'extension de l'algorithme VOLCAN pour ce cas, nommée LOCNeSs, qui suit les mêmes principes généraux d'approche orientée-sommet.

Chapitre 2

Etat de l'art : détection de communautés disjointes

Sommaire

Dans ce chapitre, nous dressons un état de l'art de la détection de communautés disjointes dans des graphes statiques. Nous commençons par formaliser dans une première section la tâche en elle-même, en détaillant les formes que peuvent prendre les communautés dans un graphe, ainsi que les méthodes connexes d'analyse de graphe, puis nous discutons dans une troisième section des différentes caractéristiques des méthodes elles-mêmes, permettant de les décrire et les classer afin d'établir une grille de lecture pour mieux appréhender l'état de l'art. La quatrième section décrit les principales familles de méthodes classées par type de détection et la cinquième section détaille en particulier la famille dite « orientée-sommet ». Les informations contenues dans ce chapitre peuvent être complétées par la lecture de différents ouvrages dédiés (Newman, 2010, Barabási, 2016).

2.1 Communautés disjointes vs. recouvrantes

Nous avons introduit dans le chapitre précédent (section 1.3) la notion de communautés dans un graphe de pair avec celle de leur détection de façon générale. Nous en distinguons ici deux types, les communautés disjointes et recouvrantes.

L'ensemble des communautés dans un graphe peut être défini sous la forme d'ensemble de sous-graphes (cf. section 1.3, p. 23). Comme il s'agit de sous-graphes induits, ils sont totalement déterminés par leur ensemble de sommets ou par leur ensemble d'arêtes. Dans la suite de ce manuscrit nous utilisons la représentation la plus répandue, comme ensemble de sommets.

L'ensemble de communautés $C = \{c_1, c_2..., c_k\}$ est donc défini comme une partition de l'ensemble V des sommets du graphe :

$$\forall i \in [\![1, k]\!], c_i \subset V$$

$$\bigcup_{i \in [\![1,k]\!]} c_i = V$$

De façon générale, la décomposition d'un ensemble de données en sous-ensembles est dite disjointe si elle est telle qu'un élément est présent dans exactement un sous-ensemble. Les intersections entre sous-ensembles sont donc vides.

Dans tout ensemble de données, en particulier dans les graphes de terrain, il peut être naturel de ne pas produire une décomposition en ensembles disjoints : il existe de multiples justifications au fait qu'un élément puisse appartenir à plusieurs ensembles simultanément. Une telle décomposition est dite *recouvrante*, ou *chevauchante* (*overlapping* en anglais). En fouille de données, le partitionnement en ensembles non-disjoints existe depuis longtemps (Jardine & Sibson, 1968, Shepard & Arabie, 1979, Ozawa, 1985).

Il en est de même pour la détection de communautés dans les graphes, on distingue ainsi les communautés disjointes et *recouvrantes* ou *chevauchantes* :

Définition Un ensemble de *communautés disjointes* d'un graphe G est une partition disjointe de ses sommets $C = \{c_1, c_2..., c_k\}$, c'est-à-dire telle que :

$$\begin{gathered} \forall i, c_i \subset V \\ \bigcup_i c_i = V \\ \forall c_i, c_j \in C, i \neq j, c_i \cap c_j = \emptyset \end{gathered}$$

Définition Le cas recouvrant n'impose pas la même contrainte : un ensemble de *communautés recouvrantes* (appelé *cover* en anglais) d'un graphe G est une partition de ses sommets $C = \{c_1, c_2..., c_k\}$ telle que :

$$\begin{gathered} \forall i, c_i \subset V \\ \bigcup_i c_i = V \\ \exists c_i, c_j \in C, i \neq j, c_i \cap c_j \neq \emptyset \end{gathered}$$

Ce chapitre présente un état de l'art des méthodes de détection de communautés *disjointes*, les méthodes traitant du cas recouvrant, qui constituent souvent des extensions des méthodes pour communautés disjointes, sont présentées section 4.2, p. 92.

2.2 Méthodes de fouille de graphes précurseures

Si le concept de détection de communautés dans les réseaux est apparu plutôt récemment (début des années 2000), la décomposition algorithmique d'un graphe en sous-graphes est une tâche qui a suscité de nombreux de travaux de recherches depuis plusieurs décennies.

Cette section présente les méthodes de fouille de graphe qui reposent sur des définitions contraintes de communautés dans le cadre de la théorie des graphes, ainsi que des méthodes issues du domaine de la fouille de données, qui peuvent être transposées pour l'identification de communautés (Gondran & Minoux, 1995, Kleinberg & Tardos, 2005). Elles constituent les précurseures des méthodes décrites dans les sections suivantes.

2.2.1 Enumération des cliques

De par leur complétude, les cliques ont logiquement formé de bons modèles pour les communautés. Le principal inconvénient est que l'énumération des cliques ou la recherche de cliques maximales dans un graphe sont des problèmes NP-difficiles, voire NP-complets dans certains cas. Un algorithme célèbre est celui de Bron et Kerbosch (1973), listant

toutes les cliques maximales d'un graphe, qui peut être utilisé comme base pour détecter des communautés.

2.2.2 Décomposition de graphe

La *décomposition de graphe* est, stricto sensu, la scission d'un graphe en plusieurs composantes. Cependant, le terme « décomposition de graphe » est utilisé pour recouvrir différents types de méthodes. Nous présentons ici les trois principaux : en sous-graphes d'arêtes disjointes, modulaire et en k-coeurs.

Décomposition en sous-graphes d'arêtes disjointes

Cette décomposition produit un ensemble de sous-graphes tels qu'aucun d'entre eux ne partage d'arête avec un autre (*edge-disjoint subgraphs*). C'est un problème NP-complet très étudié (Dor & Tarsi, 1997).

Soit $\mathcal{G} = \{S_1, ..., S_k\}, S_i \subset E$ une telle décomposition d'un graphe $G = (V, E)$, elle vérifie :

$$\bigcup_i E(S_i) = E$$
$$\forall i \neq j, E(S_i) \cap E(S_j) = \emptyset$$

Décomposition modulaire

La scission d'un graphe en sous-graphes de sommets disjoints est généralement appelée *décomposition modulaire*, où les sous-ensembles de sommets sont appelés *modules* (également : ensembles autonomes, ensembles homogènes, intervalles et ensembles partitifs) (Gallai, 1967, Montgolfier, 2003, Habib & Paul, 2010).

La définition formelle est semblable à la décomposition en sous-graphes d'arêtes disjointes, en considérant des sous-ensembles de sommets disjoints respectant les contraintes suivantes : chaque module S_i doit former une composante connexe et tous ses sommets doivent partager les mêmes voisins dans $\overline{S_i}$.

Soit $\mathcal{G} = \{S_1, ..., S_k\}, S_i \subset V$ une telle décomposition d'un graphe $G = (V, E)$, elle vérifie :

$$\bigcup_i V(S_i) = V$$
$$\forall i \neq j, V(S_i) \cap V(S_j) = \emptyset$$
$$S_i \text{ est connexe}$$
$$\forall i, \forall u, v \in S_i, \Gamma(u) \cap \overline{S}_i = \Gamma(v) \cap \overline{S}_i$$

La décomposition de graphes, particulièrement la décomposition modulaire, est un champ d'étude à part entière en algorithmique des graphes qui dispose de familles de méthodes dédiées, produisant des *arbres de décomposition*. Le premier algorithme, de complexité polynomiale, a été proposé par James et al. (1972). D'autres, linéaires, ont depuis été introduits (Cournier & Habib, 1994, McConnell & De Montgolfier, 2005, Tedder et al., 2008).

Décomposition en k-cœurs

Un k-cœur d'un graphe G est un sous-graphe connexe maximal dans lequel tous les sommets sont de degré au moins k. Garantir un degré k tout en ne cherchant pas à énumérer les cliques est intéressant, car un cœur dense (de k élevé) contient un certain nombre de quasi-cliques (cf. section 1.3.1, p. 23) tout en nécessitant des calculs de complexité algorithmique moindre. Les k-cœurs vérifient une structure d'imbrication, les cœurs les plus denses étant évidemment inclus dans les cœurs les moins denses.

Une décomposition en k-cœurs peut donc permettre de trouver les communautés dans un graphe (Dorogovtsev et al., 2006, Giatsidis et al., 2011).

2.2.3 Partitionnement, clustering de graphe

D'autres partitions de l'ensemble des sommets d'un graphe peuvent être obtenues en leur appliquant des algorithmes de partitionnement (ou clustering) (Jain et al., 1999), dont nous présentons ici trois types : les partitionnements par critère, par recherche de flots et coupes et deux types de partitionnements plus spécifiques : hiérarchique et spectral.

Partitionnement par critère

Il s'agit d'utiliser un critère déterminant quelles arêtes couper afin d'obtenir des sous-graphes. On supprime ainsi des arêtes, itérativement ou non, jusqu'à déconnecter des composantes, produisant des sous-graphes disjoints.

On peut citer en exemple l'algorithme de Kernighan et Lin (1970), utilisé pour concevoir des circuits électroniques et intégrés VLSI. Il scinde un graphe en deux sous-graphes disjoints, en minimisant la somme des poids des arêtes supprimées.

Pour les graphes de terrain, les *centralités* sont très utilisées, dont la centralité d'intermédiarité (*betweenness*, Freeman (1977)), qui mesure la propension d'un sommet à se trouver sur le plus court chemin entre deux autres sommets. On peut citer l'article de Girvan et Newman (2002), qui ont introduit la *edge betweenness* pour partitionner un graphe : on calcule cette edge betweenness pour toutes les arêtes du graphe, puis on retire celle de plus haute betweenness et on répète ces étapes itérativement (on peut ne calculer la betweenness que des arêtes qui ont été affectées par une suppression) jusqu'à obtenir un niveau de partitionnement souhaité.

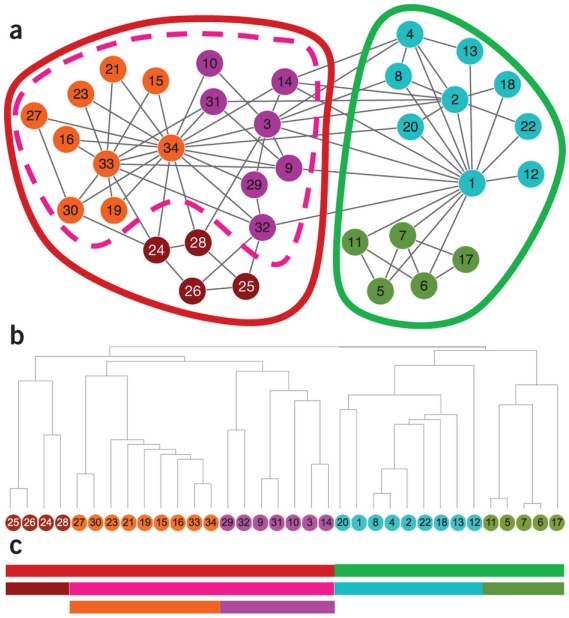


FIGURE 2.1 – Graphe et structure modulaire hiérarchique (a), représentée sous forme de dendrogramme (b) avec le code couleur des ensembles (c). Au plus haut niveau, il y a deux communautés (rouge et verte), au plus bas il y en a 5 (couleurs des sommets) Crédit : Shi et al. (2013)

Partitionnement par recherche de flots et coupes

Les flots et coupes sont des notions centrales en théorie des graphes et en recherche opérationnelle : la coupe autour d'un ensemble de sommets a été rappelée dans la section 1.1.4, p. 9, le flot est la somme des informations qui peuvent circuler dans le graphe, il est lié aux « capacités » (généralement les poids) des arêtes. Un théorème lie la recherche d'un flot maximum à celui d'une coupe minimale. Ces notions étant riches et complexes, nous renvoyons à l'ouvrage de Gondran et Minoux (1995) pour plus de détails sur le sujet.

Les communautés possédant la propriété d'être constituées d'ensembles de sommets ayant une faible connectivité entre eux (cf. annexe C.1, p. 193), on peut rechercher des communautés en réalisant des coupes minimales selon un critère de connectivité. Il existe notamment des méthodes basées sur la conductance : on part en général d'une coupe du graphe de valeur la plus faible possible (ou autre partition optimisée selon un critère voulu) et on en améliore la conductance (Lang & Rao, 2004). Leskovec et al. (2008) ont établi qu'une combinaison de cette méthode avec l'algorithme de partitionnement METIS (Karypis & Kumar, 1998) produisait des communautés de bonne qualité.

Partitionnements hiérarchique et spectral

Le regroupement (ou clustering) hiérarchique consiste à mesurer la similarité ou la dissimilarité entre les éléments de données puis à les rapprocher hiérarchiquement : on constitue

un arbre appelé dendrogramme dont chaque nœud non-terminal est le regroupement des nœuds sous lui, qui ont le même niveau de (dis)similarité (Johnson, 1967, Karypis et al., 1999). Le partitionnement hiérarchique d'un graphe en communautés et le dendrogramme associé sont illustrés figure 2.1.

Un autre type de clustering populaire est le clustering spectral, basé sur les valeurs propres de la matrice laplacienne des données considérées, par exemple avec l'algorithme de Shi et Malik (2000). On utilise pour cela une représentation matricielle du graphe, telle qu'une matrice d'adjacence. Ce type de partitionnement est néanmoins coûteux et difficile à exécuter sur de grands graphes (Newman, 2013).

Il est également possible de réaliser un partitionnement spectral basé sur la conductance, dans ce cas on utilise les bornes fournies par l'inégalité de Cheeger (Cheeger, 1969, Jerrum & Sinclair, 1989), bornant la conductance aux valeurs propres de la matrice laplacienne du graphe ; on réalise ensuite un partitionnement spectral en utilisant cette matrice.

2.3 Caractéristiques des méthodes existantes

Dans cette section, nous présentons quelques caractéristiques choisies décrivant les méthodes existantes de détection de communautés (Fortunato, 2009). Le choix de ces caractéristiques est relatif aux algorithmes proposés dans la thèse, permettant d'établir une grille de lecture et de classification des différentes méthodes présentées dans la section 2.4, p. 48.

2.3.1 Définition des communautés recherchées

Comme nous l'avons souligné dans le chapitre 1, il n'existe pas de consensus sur la définition des communautés, ce qui conduit à une grande variabilité des résultats obtenus par les différentes méthodes existantes. Ainsi, chercher des cliques ou des quasi-cliques, des communautés fortes ou faibles (cf. section 1.3.1, p. 23) ne donne pas les mêmes résultats. Les algorithmes peuvent donc d'abord être distingués par les propriétés structurelles des communautés qu'ils identifient, même si l'établissement de celles-ci n'est pas aisé.

2.3.2 Localité vs globalité

L'opposition local vs global recouvre plusieurs caractéristiques différentes, elle est en cela parfois trompeuse. Nous nous intéressons ici aux deux niveaux principaux : celui des communautés et celui de leurs critères d'évaluation.

Communauté globales vs. communautés locale

Une structure communautaire globale est une partition de l'ensemble des sommets comme nous l'avons présenté jusqu'ici : chaque sommet est affecté et appartient à une communauté (ou plusieurs, pour les communautés recouvrantes).

Une communauté locale, quant à elle, est une communauté relative à un sommet ou un ensemble de sommets, identifiée à l'aide d'un processus d'exploration à partir de ce ou ces sommets. Un point clé de cette exploration est, par exemple, d'identifier la *frontière* de la communauté : les arêtes entre les sommets qui la composent et ceux qui lui sont extérieurs.

Dans les réseaux sociaux notamment, la multiplicité de petites communautés nommées *ego networks* (centrées sur un utilisateur), établie depuis longtemps en sociologie (Freeman, 1982), a motivé de nombreux travaux sur les communautés locales (Leskovec et al., 2009).

Une différence entre une communauté locale et une communauté appartenant à la partition entière d'un graphe se situe au niveau de l'indépendance de leur existence : en effet on peut rechercher une communauté locale seule, sans s'intéresser au reste du reste du graphe. Dans ce cas, seule une partie des sommets et arêtes est considérée. Il existe des méthodes détectant uniquement des communautés locales, s'arrêtant à la frontière de celles-ci (Clauset, 2005), sans explorer entièrement le graphe. A l'opposé, une communauté « globale » n'existe pas indépendamment des autres, elle résulte du partitionnement du graphe entier.

Critère global vs critère local

Un critère global évalue la qualité de la totalité d'une structure communautaire. Il nécessite pour cela des informations au-delà d'une seule communauté, souvent sur l'intégralité du graphe. L'avantage est qu'il permet de considérer une vision d'ensemble : une communauté localement optimale pouvant ne contribuer que faiblement à l'optimalité de la structure communautaire du graphe entier, le critère permet d'optimiser les différentes communautés locales afin d'atteindre un optimum global. Un critère global très connu est la modularité (Newman, 2004b), détaillée section 2.4.1, p. 48.

Un critère de qualité local, au contraire, est généralement entendu comme un critère évaluant un sous-graphe en tant que tel, sans requérir d'informations sur le reste du graphe. On peut citer en exemple la densité interne d'une communauté, qui ne nécessite que le nombre de sommets et d'arêtes dans le sous-graphe considéré, au contraire de la modularité. L'avantage des critères locaux est qu'ils peuvent être calculés rapidement et indépendamment sur chaque communauté. Les méthodes les exploitant sont donc plus facilement parallélisables. De nombreux critères globaux ont donc été adaptés en critères locaux, ainsi une version locale de la modularité a été proposée par Clauset (2005).

2.3.3 Agglomératif vs divisif

Les qualificatifs agglomératif et divisif sont appliqués aux méthodes de constitution d'ensembles de tous types pour décrire leur stratégie : regroupement ou partitionnement (respectivement).

Les méthodes agglomératives peuvent être décrites comme *bottom-up* : on considère tout d'abord chaque élément de base (dans le cas d'un graphe, un sommet) et on opère des regroupements jusqu'à obtenir des ensembles d'une qualité jugée suffisante.

A l'inverse, les méthodes divisives sont qualifiées de *top-down* : on considère l'ensemble des données, ici le graphe dans son ensemble, puis on cherche où « couper » (supprimer une arête dans un graphe) pour former des sous-ensembles d'une qualité suffisante.

Au fur et à mesure des étapes d'agglomération ou de division, on peut donc obtenir différents « niveaux » de communautés, que l'on peut voir sur l'exemple figure 2.1, p. 44. Ces niveaux sont représentables par un *dendrogramme*, également visible sur la figure.

2.3.4 Répartition des calculs, centralisation/décentralisation

Les méthodes existantes, au delà de différences sur les définitions des communautés en elles-mêmes, peuvent aussi être différenciées selon des critères algorithmiques de de répartition des calculs. En effet, au fur et à mesure que les graphes de terrain étudiés deviennent de plus en plus grands, la détection ne peut plus être effectuée sur une seule machine, même puissante.

La tâche de distribution d'un algorithme existant étant complexe, certains travaux s'attachent à proposer des versions parallélisables d'algorithmes éprouvés (Montresor et al., 2013), alors que d'autres cherchent à produire des méthodes nouvelles, directement utilisables dans une architecture parallèle (Leung et al., 2009, Martelot & Hankin, 2014).

Si l'algorithme utilise en entrée des informations *centralisées*, il doit accéder à une mémoire *centrale* uniquement, alors que si ces informations sont *décentralisables* elles peuvent être réparties en ensembles indépendants, que l'on peut également distribuer dans la mémoire de chaque processus léger parallèle (*thread*). Il est alors beaucoup plus facile de distribuer les calculs car les différents processus peuvent travailler indépendamment, sans perdre de temps à accéder à un information centralisée, nécessairement partagée, ou à partager eux-mêmes des données entre eux.

La centralisation n'est pas assimilable à la répartition : un algorithme peut être à la fois centralisé et réparti. Dans ce cas, un programme central se charge par exemple de répartir les calculs sur les nœuds disponibles et de collecter et agréger les résultats, à l'instar du framework *MapReduce* de Google (Lee et al., 2012).

Une méthode globale est plus efficace dans une architecture centralisée, en effet s'il y a besoin d'accéder à une information sur nombre de zones du graphe, il est plus simple et rapide d'aller les chercher dans une mémoire centrale (si tant est que la taille du graphe

permette de le stocker en mémoire centrale (McCune et al., 2015)). A l'inverse, une méthode locale, si elle utilise des informations n'excédant pas le voisinage proche d'un sommet, sera plus facilement décentralisable, bien qu'une implémentation décentralisée soit complexe, comme nous le montrons avec VOLCAN section 3.3, p. 74.

2.4 Principales familles de méthodes de détection disjointe

Dans cette section, nous dressons un état de l'art des méthodes récentes (postérieures à 2000) destinées à trouver des communautés dans des graphes, plus précisément dans cette section des communautés disjointes. On considère ces méthodes en-dehors du champ de la fouille de graphe, car répondant à des contraintes sémantiques spécifiques, comme nous l'avons décrit dans les considérations à propos du vocabulaire, section 1.3.2, p. 25.

Cette famille de méthodes étant elle-même très vaste, nous proposons de distinguer cinq grandes catégories, correspondant aux cinq sous-sections suivantes, pour lesquelles nous présentons des algorithmes. Une famille plus spécifique au cœur de la thèse, les méthodes orientées-sommet, est présentée dans la section 2.5.

Les méthodes présentées sont, sauf indication contraire, *centralisées* (cf. sous-section précédente : l'implémentation proposée ne répartit ni l'information, ni les calculs.

La lecture de cette section peut être complétée par celle des états de l'art de Fortunato (2009); Papadopoulos et al. (2012); Bedi et Sharma (2016); Zhao (2017).

2.4.1 Optimisation d'un critère de qualité

Détecter des communautés en optimisant un critère est la première famille de méthodes, en termes de chronologie et d'importance. En effet elle dérive naturellement des travaux de fouilles de graphe menés en recherche opérationnelle et combinatoire (cf. sections 2.2, p. 41 et 2.3.3, p. 47).

Ces méthodes reposent sur le choix d'un critère mesurant la qualité d'une partition (ou d'une communauté dans le cas local) et consiste à optimiser ce critère. Nous détaillons ici le critère le plus populaire, la *modularité*. Il en existe bien sûr de nombreux autres comme par exemple la *surprise* (Aldecoa & Marín, 2011) qui comble certaines lacunes comme la limite de résolution (voir ci-dessous), l'information circulant dans le graphe optimisée par la méthode InfoMap (Rosvall & Bergstrom, 2008) ou le critère WCC (Weighted Community Clustering), basé sur la distribution de triangles dans le graphe (Saltz et al., 2015). On peut noter pour ce dernier que les auteurs proposent également une implémentation parallèle de leur algorithme d'optimisation (global), qui le rend très rapide.

Nous présentons ensuite quelques méthodes utilisant la modularité pour déterminer une structure communautaire dans un graphe.

Modularité

Le critère de modularité a été introduite par Newman et Girvan (2004) pour améliorer les résultats des méthodes de partitionnement de graphe existantes et pour pallier la lenteur de ces méthodes, comme celles basées sur les centralités (Girvan & Newman, 2002). L'utilisation de la modularité est généralement très liée à la détection de communautés, bien qu'elle ne soit à la base qu'un critère mathématique dont le but est de représenter l'état modulaire d'un réseau (d'où son nom). L'état modulaire, distinct de la décomposition modulaire introduite page 42, se comprend comme le fait que le réseau soit organisé en *modules* reliés entre eux, c'est-à-dire en sous-ensembles de nœuds plus connectés entre eux que vers le reste du graphe, ce qui correspond également à la définition intuitive de communauté donnée section 1.3.1, p. 23. Enfin, il ne faut pas confondre le critère de modularité présenté dans ce paragraphe, qui est un critère numérique, avec la propriété d'un graphe à présenter un état modulaire, également dénommée modularité.

La modularité a immédiatement suscité un fort engouement dans le domaine de l'analyse des réseaux, ainsi que dans les domaines faisant usage des graphes de terrain tels que la biologie moléculaire. C'est une mesure incontournable pour exploiter des réseaux, bien qu'elle souffre de plusieurs inconvénients (Lancichinetti & Fortunato, 2011). Parmi ceux-ci on peut citer la difficulté de la maximiser efficacement, une tâche NP-complète dans la plupart des cas (Brandes et al., 2007), ainsi que la limite de résolution (Fortunato & Barthélemy, 2007) : la modularité ne capture pas, ou mal, les plus petites communautés. En particulier, lorsqu'on maximise itérativement la modularité, ce qui permet d'obtenir une structure hiérarchique (voir le schéma figure 2.1, p. 44), la modularité favorise les regroupements et donc les communautés d'une certaine taille. De nombreuses autres limitations ont également été soulevées.

La modularité, calculée pour une partition en communautés C d'un graphe G, représente la proportion des arêtes de G se trouvant à l'intérieur de chacune des communautés $c \in C$ et corrigée par rapport au hasard, c'est-à-dire comparé à la probabilité que ces arêtes soient naturellement présentes dans un modèle de référence (*null model*). Par défaut ce dernier est un graphe aléatoire comportant le même nombre d'arêtes et de sommets que le graphe G (Newman & Girvan, 2004).

Ainsi, la modularité mesure à quel point la probabilité de la répartition des arêtes s'éloigne du modèle de référence pour atteindre une structure modulaire (forte densité dans les communautés, faible entre elles). Plus les différences de densité sont importantes, plus la modularité est haute. Elle est bornée entre -1 et 1.

Soient un graphe non-orienté $G = (V, E)$ et un ensemble de communautés disjointes C défini comme une partition de V. L'écriture la plus simple de la modularité de la partition

C du graphe G, notée $Q(C, G)$, est :

$$Q(C, G) = \sum_{c \in C} (e_{cc} - a_c^2)$$

où e_{cc} utilisé dans la formule est la fraction des arêtes reliant les sommets de la communauté c à des sommets de la communauté c'.

$$e_{cc'} = \frac{|\{(u, v) : (u, v) \in E, u \in c, v \in c'\}|}{|E|}$$

et a_c est la fraction des arêtes qui ont au moins une extrémité dans la communauté c :

$$a_c = \frac{|\{(u, v) : (u, v) \in E, u \in c\}|}{|E|} = \sum_{c' \in C} e_{cc'}$$

Une définition plus développée mais équivalente est la suivante :

$$Q(C, G) = \frac{1}{2|E|} \sum_{u \in V} \sum_{v \in V} \left[A_{uv} - \frac{d_u d_v}{2|E|} \right] \delta(c_u, c_v)$$

où :

— A_{uv} est la valeur de la matrice d'adjacence pour (u, v).
— δ est le delta de Kronecker, qui vaut 1 si ses deux arguments sont égaux et 0 sinon.

Optimisation de la modularité

Algorithme glouton de Newman

Newman (2004c) a d'abord proposé une méthode gloutonne de maximisation de la modularité, agglomérative et globale. Bien qu'efficace, cette méthode est inapplicable sur de très grands graphes étant donnée sa complexité en $\mathcal{O}(n^2)$ dans le pire des cas.

Algorithme de Louvain

Cet algorithme (Blondel et al., 2008b), agglomératif et global lui aussi, réalise la maximisation de modularité en deux phases. La première réalise les fusions des paires de sommets, ou de communautés, qui apportent un gain de modularité. La seconde transforme le graphe obtenu en un graphe dont les sommets sont les clusters du graphe initial et une arête pondérée relie deux tels sommets avec pour poids la somme des poids des arêtes entre les deux clusters.

Ces deux phases produisent un nouveau niveau hiérarchique de découpage en communautés. L'algorithme s'arrête lorsqu'aucune fusion de la première phase n'améliore plus (significativement) la modularité.

L'algorithme de Louvain offre ainsi l'avantage de proposer une structure hiérarchique des communautés, avec différents niveaux de granularité : plus le niveau hiérarchique est

élevé, plus les sommets ont été regroupés et plus les communautés sont donc de grande taille.

Bien que la complexité de cet algorithme ne puisse pas être établie analytiquement et ne soit donc pas connue avec exactitude, il a été constaté empiriquement qu'il s'exécute très rapidement sur des graphes de terrain, un effet de l'algorithmique des graphes de terrain présentée section 1.2.6, p. 23 et donne des résultats de très bonne qualité (Guillaume, 2012).

Autres méthodes basées sur la modularité

La parallélisation d'algorithmes de détection de communauté par optimisation de modularité a été explorée, notamment par Riedy et al. (2012) : ils ont adapté et testé l'algorithme de Louvain sur 1288 processeurs simultanément, réussissant à traiter un graphe de 122 millions de sommets et environ 2 milliards d'arêtes en un peu plus de deux heures.

Clauset (2005) a proposé une adaptation de la modularité afin de pouvoir détecter une communauté locale, ainsi qu'un algorithme l'utilisant. D'autres versions locales de la modularité ont été proposées par Luo et al. (2008) ainsi que Chen et al. (2009).

2.4.2 Processus stochastiques

Le clustering de graphes et la détection de communautés basés sur des processus stochastiques sont souvent désignés par le terme anglophone de *stochastic blockmodeling* (Holland et al., 1983, Doreian et al., 2005).

Le principe général est de partir d'un modèle de communautés (le *blockmodel*) décrivant généralement des caractéristiques (*features*) relatives aux sommets, aux arêtes et aux relations entre sommets et arêtes et de trouver un partitionnement du graphe lui correspondant le mieux. De nombreuses méthodes ont été proposées à cet effet : certaines combinent des modèles de mélanges ou de vraisemblance avec un algorithme d'espérance-maximisation (Newman & Leicht, 2007), à l'aide de processus markoviens tels que les MCMC [12] (Hastings, 2006) ou procèdent par inférence statistique (Tang & Liu, 2009).

Par exemple, l'algorithme de Newman et Leicht (2007) cité précédemment définit un modèle de graphe sur la base d'observations sur des motifs de répartitions des connexions entre sommets dans des graphes de terrain et utilise la maximisation de la vraisemblance pour trouver une partition. D'autres approches ont, par exemple, généralisé la modularité afin de l'utiliser comme modèle (Reichardt & Bornholdt, 2006).

Le modèle des variables latentes (Everitt, 1984) est également très utilisé : la répartition en communautés est vue comme une structure latente et les algorithmes utilisés sont les mêmes que cités plus haut : MCMC (Hoff et al., 2002, Handcock et al., 2007), algorithme EM basé sur un maximum de vraisemblance (Handcock et al., 2007) ou sur un modèle de description plus complexe de la structure d'un graphe (Zanghi et al., 2010).

12. Monte-Carlo Markov Chains

2.4.3 Propagation de labels

La propagation de labels est apparue plus tardivement, avec les travaux de Raghavan et al. (2007). Le principe est simple : chaque sommet propage son identifiant unique (*label*), c'est-à-dire qu'il le communique à ses voisins, qui le communiquent en plus de leur propre identifiant à leurs voisins etc. Au bout d'un certain nombre de transmissions d'identifiants, une fois la stabilisation atteinte (les sommets ne reçoivent plus de nouveau label), l'algorithme s'arrête et chaque sommet connaît alors sa communauté : il s'agit de l'identifiant qu'il a reçu le plus fréquemment.

Cet algorithme repose sur la définition intuitive des communautés : la densité étant plus forte dans une communauté, un identifiant d'un sommet appartenant à celle-ci est plus propagé en son sein que vers les communautés extérieures.

Chaque sommet gérant la transmission des identifiants qu'il reçoit, la méthode pourrait être considérée d'une certaine façon comme orientée-sommet et citée dans la section suivante. Néanmoins, nous l'avons classée à part car elle ne procède pas à l'analyse du voisinage des sommets mais simplement à une inondation par propagation, incompatible avec une utilisation dans le contexte du chapitre 1.4.

La facilité d'implémentation de ce paradigme, la faible complexité ($\mathcal{O}(m)$) permettant une exécution très rapide, ainsi que les résultats empiriques intéressants ont entraîné de nombreux travaux et de nombreuses variantes ont vu le jour (Leung et al., 2009, Šubelj & Bajec, 2011, Cordasco & Gargano, 2012) améliorant l'équilibrage de la charge de propagation, la robustesse de l'algorithme, les performances en parallélisation etc.

2.4.4 Regroupement par similarité

Ces algorithmes se rapprochent, dans leur concept, d'algorithmes de partionnement de données (cf. section 2.2.3, p. 43), il existe des algorithmes de détection de communautés utilisant des mesures de proximité ou similarité afin de constituer les clusters. Par exemple, Leicht et al. (2006) ont exploré les similarités entre sommets et ont montré expérimentalement que des similarités exprimées en fonction de propriétés de sommets permettent d'identifier certaines propriétés telles que les affinités ou l'homophilie (cf. section 1.2.3.6).

L'algorithme de Zhang et al. (2009) est hybride entre l'optimisation d'un critère et le regroupement par similarité. Il repose sur la dynamique de propinquité pour détecter les communautés, un concept sociologique décrivant les mécanismes de formation de relations entre individus en se basant sur leur proximité physique ou psychologique. Les auteurs utilisent une définition adaptée pour les graphes, proche de l'attachement préférentiel et optimisent cette métrique par étapes successives.

Danisch et al. (2013a) mesurent la proximité d'un sommet avec tous les autres sommets du graphe selon différents critères tels que la *carryover opinion* Danisch (2015). En ordonnant ces proximités par ordre décroissant, on peut identifier la « frontière » de la

communauté locale, atteinte lorsque la courbe des valeurs de proximité ordonnées amorce une chute. L'identification d'une communauté autour d'un sommet particulier s'apparente aux communautés centrées-graines (les auteurs utilisent le terme *ego-centrées*), néanmoins la comparaison avec tous les sommets du graphe en fait une approche non locale, c'est pourquoi nous l'avons placée dans celle section et non dans cette consacrée aux méthodes orientées-sommet.

Les auteurs ont également apporté des améliorations à leur algorithme, notamment en utilisant plusieurs centres de calcul des proximités pour une même communauté (communauté multi-égo-centrées, Danisch et al. (2013b)).

2.4.5 Vers des méthodes orientées-sommet

Les différents avantages des méthodes utilisant de l'information locale (cf. section 2.3.2, p. 45), combinés avec les avantages des méthodes citées dans les sous-sections précédentes, les regroupements par similarité en particulier, ont fait naître une nouvelle famille de méthodes où la donnée de base considérée n'est plus un graphe ni un sous-graphe mais un seul sommet, avec son voisinage.

Cette famille encore relativement disparate voit régulièrement de nouveaux algorithmes apparaître, nous la nommons dans ce manuscrit « méthodes de détection de communautés orientées-sommet » et elle fait l'objet de la section suivante.

2.5 Les méthodes orientées-sommet

Cette section est dédiée à une présentation détaillée de la famille des méthodes orientées-sommet à laquelle se rattachent nos propositions, décrites dans les chapitres 3, 4 et 6. Relativement récentes, ces méthodes n'utilisent pas toutes les mêmes termes et mettent en œuvre des approches variées, même si elles partagent un même principe. Nous proposons ici une structuration de cette nouvelle famille en quatre catégories. Nous décrivons tout d'abord leurs principes généraux avant de détailler des éléments de théorie des graphes, en premier lieu la conductance, pertinents pour la description des méthodes venant ensuite, en particulier de leurs implications théoriques. Enfin, nous dressons un état de l'art des méthodes orientées-sommet en détection de communautés sur graphe, autour de trois techniques : marches aléatoires et conductance, centrées sur une graine et utilisant des sommets *leaders*.

2.5.1 Introduction aux méthodes orientées-sommet

Le concept *vertex-centric* (aussi *vertex-oriented* ou *node-centric*), que l'on traduit par orienté-sommet, est utilisé dans différents contextes pour décrire un paradigme selon lequel on se place du point de vue d'un sommet du graphe pour effectuer les opérations. Ce paradigme, simple à exprimer, s'incarne de différentes façons selon l'angle de traitement du

graphe abordé. Nous présentons ci-après deux de ces angles : l'implémentation et l'exécution parallélisée d'algorithme, avec le paradigme Think-Like-a-Vertex (TLAV), sous-section 2.5.1.1 et la conception d'algorithmes de détection de communautés, sous-section 2.5.1.2.

2.5.1.1 Méthodes de traitement décentralisées : paradigme Think-like-a-Vertex

Le concept orienté-sommet a été popularisé dans l'analyse de graphe parallèle et/ou décentralisée sous la dénomination *« Think(ing)-Like-a-Vertex » (TLAV)* (Tian et al., 2013). L'idée est de créer un sous-programme s'exécutant à partir d'un sommet d'un graphe, prenant par exemple en entrée uniquement un sommet donné du graphe et éventuellement son voisinage.

A partir de ce point, le programme parallélisé consiste à lancer le sous-programme sur chaque sommet du graphe. Plus précisément, McCune et al. (2015) proposent la définition suivante d'un programme suivant le paradigme TLAV (traduction libre de l'anglais) :

Définition Les *frameworks* Think-like-a-vertex sont des plateformes logicielles qui exécutent itérativement un programme dont l'entrée est un unique sommet (*vertex-program, VP*), sur chacun des sommets d'un graphe. Le VP est conçu selon le point de vue d'un sommet, nommé sommet principal, recevant en entrée les données relatives à ce sommet ainsi que celles relatives aux sommets adjacents et arêtes incidentes. Le VP est exécuté sur chaque sommet de l'ensemble des sommets du graphe, de façon synchrone ou asynchrone. L'exécution s'arrête après un nombre donné d'itérations, ou après que tous les VP ont convergé.

Un VP peut également prendre plusieurs sommets en entrée (*vertex-subgraph-centric*), qu'il traite les uns après les autres, nécessitant une phase préalable de partitionnement des sommets.

Le modèle de programmation orienté-sommet est moins expressif que les algorithmes conventionnels travaillant avec l'intégralité du graphe en mémoire mais passent facilement à l'échelle et multiplient les possibilités d'utilisation du parallélisme.

Caractéristiques

Selon McCune et al. (2015), les quatre principales caractéristiques d'une approche d'analyse de graphes TLAV sont :

- *Timing* : de quelle façon les sommets à traiter sont ordonnés par le planificateur de la machine (*scheduler*). Les possibilités sont : synchrone, asynchrone ou hybride.
- *Communication* : comment les informations sont transmises entre les différents processus de VP. Elle peut être par passage de messages, en mémoire partagée ou par messages actifs.
- *Modèle d'exécution* : c'est le modèle d'implémentation du VP qui définit également comment circule le flot de données, c'est-à-dire qu'il influe sur le mode de commu-

nication du point précédent. Le VP peut être implémenté en une, deux ou trois phases correspondant à autant de fonctions :

1. une seule fonction qui effectue l'intégralité du traitement sur le sommet principal,
2. deux fonctions *scatter* et *gather*, la première distribuant les données liées au sommet principal à ses voisins et la seconde collectant et agrégeant les résultats,
3. trois fonctions *gather*, *apply* et *scatter*, la première compilant des données transmises par un ensemble de sommets en entrée (ex. voisinage du sommet principal), la seconde modifiant en conséquence les données liées au sommet principal et la dernière propageant les changements.

Tous les traitements qui ne sont pas contenus dans le VP sont effectués en mémoire principale, a priori de façon non distribuée.

— *Partitionnement* : le partitionnement est la tâche cruciale consistant à séparer le graphe d'entrée en différents sous-graphes qui sont ensuite analysés en parallèle. Partitionner les sommets d'un graphe en $k \in \mathbb{N}^*$ ensembles distincts de façon à équilibrer la taille des ensembles et la charge de calcul est NP-complet. Un bon partitionnement permet d'économiser la communication entre processus et donc d'améliorer les performances globales du programme en temps d'exécution.
Les méthodes les plus efficaces sont des heuristiques de coupe de graphe ou de répartition sommet par sommet.
Un fait très intéressant est que les méthodes de détection de communautés ont récemment été popularisées pour le partitionnement de graphes visant à leur analyse distribuée : en effet les communautés étant empiriquement des ensembles denses moins connectés entre eux, elles forment de parfaits candidats à une bonne partition en sous graphe. En effet, la communication inter-processus étant majoritairement due aux arêtes liant deux sommets qui ne sont pas traités par le même VP, ce cas se réduit en affectant les sommets d'une même communauté à un même VP. En particulier, les algorithmes de détection décentralisées, premier lieu les méthodes par propagation de labels (Raghavan et al., 2007), ont été beaucoup adaptées pour le partitionnement.

Le premier outil implémentant le principe d'analyse de graphe TLAV, et sans doute le plus connu, est *Pregel*, introduit par Google (Malewicz et al., 2010). D'autres ont été proposés par la suite (Han et al., 2014a, McCune et al., 2015) parmi lesquels la populaire version libre *Giraph*[13] soutenue par la Fondation Apache.

Il existe également des frameworks *orientés-arête* qui utilisent des paradigmes d'implémentation plus ou moins proche de TLAV, une arête pouvant être considérée comme une paire de sommets.

13. `http://giraph.apache.org/`

Avantages et inconvénients

Le principal avantage de l'approche TLAV est de permettre une décomposition très fine des algorithmes, au sens où les tâches sont divisées au maximum, jusqu'à l'unité de base : le sommet. Elles permettent ainsi un passage à l'échelle aisé et très flexible, et une grande rapidité de traitement des très gros graphes.

En contrepartie, elles nécessitent d'adapter les algorithmes existants, ou d'en créer d'autres, afin de les implémenter. De plus, il n'est pas toujours possible de scinder un problème en sous-problèmes résolubles localement.

2.5.1.2 Algorithmes de détection de communautés

Récemment, les termes *ego-centered, vertex-centric (ou cent(e)red)* et *vertex-oriented* ont commencé à figurer dans des intitulés de méthodes de détection de communautés sur graphe. Avant cela, les méthodes étaient également référencées avec des mots-clés généraux qualifiant la détection tels que *local community detection*, *ego-nets detection*, ou plus spécifiquement basées sur des *leaders* ou des *seeds* (graines).

Pour nous, les méthodes orientées-sommet sont toutes les méthodes locales basant la détection du point de vue d'un sommet du graphe, c'est-à-dire que les algorithmes peuvent être exprimés sous la forme de programmes prenant en entrée un sommet et éventuellement son voisinage, de la même manière que les Vertex Program de TLAV. Cependant, nous n'affirmons pas que les méthodes présentées dans la suite de cette section sont directement implémentable dans une architecture TLAV, cette question dépassant le cadre de l'état de l'art de la thèse. Les trois sous-sections suivantes détaillent trois grandes familles de méthodes orientées-sommet : utilisant les marches aléatoires sur graphe (sous-section 2.5.2), centrées sur des graines (sous-section 2.5.3) et basées sur des sommets leaders (sous-section 2.5.4). Il est à noter que certains auteurs ne font pas réellement de distinction entre les deux. La classification que nous proposons peut donc se heurter à l'utilisation indifférente du vocabulaire par les auteurs des différents travaux dans le domaine.

Avantages et inconvénients

Les avantages et inconvénients de ces algorithmes sont principalement les mêmes que ceux des méthodes locales (Fortunato, 2009, Xie et al., 2013).

D'un côté, leur grande flexibilité permet des implémentations parallèles par exemple, facilitant ainsi le passage à l'échelle et l'analyse très rapide de très grands graphes, également une moins forte sensibilité à certains paramètres locaux tels que la limite de résolution qui impacte fortement certaines mesures globales telles que la modularité (Fortunato & Barthélemy, 2007). Un autre argument en faveur de cette flexibilité est leur complexité : elle est généralement liée au degré des sommets et à la taille des communautés, des valeurs bien

plus faibles que le nombre de sommets ou d'arêtes impactant la complexité des méthodes globales.

D'un autre côté cependant, on note une plus grande difficulté à capturer globalement les limites des zones de forte densité où se trouvent les communautés, et de manière générale une tendance à produire des résultats de moins bonne qualité, constatés en évaluation sur des benchmarks (Fortunato, 2009, Xie et al., 2013).

Enfin, le rôle très important du sommet central (centre, ego, graine, leader...) implique bien évidemment que les résultats de l'algorithme, ie. ses performances qualitatives, sont très dépendants du choix des sommets centraux, comme l'ont également discuté Gleich et Seshadhri (2012).

Aspects théoriques, travaux de Gleich & Seshadhri

Les méthodes de détection orientées-sommet sont relativement populaires de par les avantages cités ci-dessus, mais peu d'études théoriques se sont intéressées en profondeur à leur fonctionnement. Cependant, s'il est possible de conduire une analyse mathématique de critères de qualité globaux, ainsi que de leur processus d'optimisation, les effets de mesures locales de voisinages sont plus difficiles à lier aux résultats d'une détection sur un graphe entier, les études théoriques sur les méthodes locales étant très peu répandues (Zhao, 2017, p. 10). L'une des premières et des plus importantes est celle de Gleich et Seshadhri (2012). Les auteurs posent tout d'abord deux conditions sur le graphe considéré, qui doit posséder :

— une distribution des degrés en queue lourde D : il doit exister des constantes positives α_1, α_2 et $\gamma < 3$ telles que, pour tout $d \in D$, le nombre de sommets f_D de degré D respecte :

$$f_d \in [\alpha_1 n/d^\gamma, \alpha_2 n/d^\gamma]$$

— un fort coefficient de clustering κ
(généralement, $\kappa \in [0.9, 1[$, la valeur 1 étant atteinte pour un ensemble de cliques disjointes, ce qui ne nous intéresse pas ici)

Différents résultats sont ensuite établis dans le cas où ces conditions sont vérifiées par un graphe. Nous reproduisons ici un théorème important proposé, ainsi que sa démonstration, dans leur article (Gleich & Seshadhri, 2012), étendu par des résultats reproduits annexe dans l'C, p. 193, montrant l'existence d'une coupe dont la valeur de conductance est liée au coefficient de clustering du graphe considéré.

Théorème 2.5.1 (Existence d'une coupe de conductance connue)**.** *Soit G un graphe de fort coefficient de clustering κ dont la distribution des degrés est en queue lourde, selon la définition de Gleich et Seshadhri (2012). Il existe une coupe au voisinage d'un sommet de conductance au plus* $4(1-\kappa)/(3-2\kappa)$

Un autre théorème, que nous ne reproduisons pas ici, montre que sous la même condition de distribution des degrés, il existe de larges k-cœurs dans le graphe.

2.5.2 Méthodes utilisant des marches aléatoires

Une première famille de détection de communautés orientées-sommet exploite la notion des marches aléatoires sur graphe (Lovász, 1996), définie comme suit :

Définition Une *marche sur un graphe* $G = (V, E)$ est une séquence de sommets $v_0, v_1, v_2... \in V$ telle que v_{i+1} est un voisin de v_i pour i.

Une *marche aléatoire sur un graphe* $G = (V, E)$ est une marche sur G pour laquelle le successeur v_{i+1} d'un sommet v_i dans la séquence de la marche est choisi à l'aide d'une distribution de probabilités.

En particulier dans une *marche aléatoire uniforme*, le sommet successeur est choisi de façon équiprobable parmi les voisins de son prédécesseur.

Les marches aléatoires sur graphe ont de nombreuses propriétés et de nombreuses utilités. Elles sont par exemple utilisées pour l'exploration du Web et sont à la base du système *PageRank* de Google (Brin & Page, 1998).

De nombreux algorithmes ont proposé d'exploiter les marches aléatoires afin de détecter des communautés sur un graphe, en utilisant le lien entre le nombre de pas effectués menant à la convergence de la marche (*temps de mixage*) et la connectivité de sous-ensembles de sommets parcourus. Le principe sous-jacent est qu'une marche aléatoire reste plus facilement piégée à l'intérieur d'une communauté car les liens y sont plus nombreux, et met du temps à la quitter. Ce phénomène, explicable à l'aide de la connectivité des graphes de terrain et de la théorie des graphes expanseurs (cf. annexe C, section C.1, p. 193), est par exemple exploité par l'algorithme WalkTrap (Pons & Latapy, 2005). Cet algorithme est néanmoins global : il exploite un ensemble de probabilités de convergence de marches issues de chacun des sommets du graphe.

La première approche locale utilisant une marche aléatoire, que nous qualifions d'orientée-sommet, a été introduite par Spielman et Teng (2004), basée sur et étendant des travaux de Lovász et Simonovits (1993).

L'algorithme proposé, nommé *Nibble*, constitue un cluster local en lançant une marche aléatoire sur un sommet choisi lui-même aléatoirement. L'estimation du temps de mixage de cette marche aléatoire (résultats de Lovász et Simonovits (1993)), permet de découvrir une coupe approximativement identifiable à ce que nous avons précédemment nommé « communauté locale », de faible conductance. L'algorithme Partition, basé sur des exécutions répétées de Nibble, permet ensuite de réaliser le partitionnement du graphe entier.

Cet algorithme a été amélioré et étendu plusieurs fois, en particulier Andersen et Lang (2006) tentent d'identifier de « bons » ensembles de sommets de départ sur lesquels utiliser l'algorithme Nibble, afin de rechercher explicitement des communautés de faible conductance.

Andersen et al. (2006) proposent une version de leur algorithme basé sur la convergence de marches aléatoires utilisant des vecteurs de PageRank personnalisé, inspirés de

l'algorithme PageRank de Google (Brin & Page, 1998), d'où leur nom (Jeh & Widom, 2003). Cette méthode consiste à réaliser une marche aléatoire avec une probabilité de retour par téléport (*jump back*) et une probabilité d'arrêt, le PageRank personnalisé étant alors la probabilité qu'une marche commençant à un sommet s s'arrête effectivement à une cible t. L'effet de piège d'une marche aléatoire, évoqué précédemment et déjà utilisé par WalkTrap, permet de lier les valeurs de PageRank personnalisé aux contours locaux d'une communauté.

L'utilisation de PageRank personnalisé est plebiscitée pour la construction de communautés basées sur l'exploration du voisinage par marche aléatoire, offrant de bons résultats qualitatifs (Gleich & Seshadhri, 2012). D'autres procédés utilisant la convergence des marches existent et donnent également de très bon résultats (Kloster & Gleich, 2014).

Sur les bases établies par Andersen et Lang (2006), d'autres travaux, notamment ceux de Zhu et al. (2013); Orecchia et Zhu (2014) proposent des améliorations, principalement concernant la complexité des méthodes concernées. Si ces méthodes semblent efficaces, elles n'en demeurent pas moins relativement lentes pour analyser de grands graphes et difficiles à décentraliser totalement : en effet chaque sommet doit gérer une marche aléatoire dans son voisinage, ce qui occasionne beaucoup de communications avec ses voisins.

2.5.3 Méthodes centrées-graine et ego-centrées

L'idée principale de ces méthodes est de détecter une communauté locale autour d'un sommet ou d'un ensemble de sommets, constituant la *graine*. Elles se rapprochent des méthodes évoquées dans la sous-section précédentes, mais utilisent des mesures différentes des convergences de marches aléatoires pour analyser le voisinage des sommets.

On peut également détecter une structure communautaire globale sur tout un graphe en sélectionnant plusieurs graines et en regroupant ou agrégeant les communautés locales d'une manière définie.

La sélection des graines initiale est donc cruciale, puisque c'est d'elle que dépend la qualité des communautés détectées. La principale différences entre les nombreuses méthodes d'une même famille tient donc à la stratégie de sélection de ces graines.

Nous présentons ici quelques exemples de méthodes détectant des communautés en partant d'un sommet (graine), ou d'un ensemble de sommets, de façon locale, correspondant au concept exposé nommé *orienté-sommet*. Nous utiliserons, dans cette sous-section, le terme *graine* pour désigner le sommet ou l'ensemble de sommets de départ. Il peut parfois porter un autre nom selon les choix de l'auteur.

Nous présentons tout d'abord les méthodes utilisant l'expansion aux abords des communautés, puis d'autres types approches plus anecdotiques : hybride et par similarité.

Algorithme 1 Cadre général d'une méthode centrée leader/graine (Kanawati, 2014a)

Require: $G = (V, E)$, un graphe connexe
Ensure: $C \subset \mathcal{P}(V)$, des communautés sous forme d'ensembles de sommets
1: $C \leftarrow \emptyset$
2: $S \leftarrow$ calculer_graines(G), ensemble de leaders/graines
3: **for** $s \in S$ **do**
4: $c \leftarrow$ etendre_communaute(s)
5: add c in C
6: **end for**
7: fusionner(C)

Expansion

Le pseudo-code d'une méthode générique de détection par expansion est donné algorithme 1. C'est la méthodologie la plus souvent rencontrée. Des graines sont sélectionnées à l'aide d'une stratégie variant, puis une communauté est construite autour de chacune de ces graines.

Une des premières méthodes proposant cette approche est celle de Bagrow et Bollt (2005). Leur algorithme explore le voisinage d'un sommet donné en calculant deux métriques : le degré émergeant et le degré total émergeant. Ces calculs sont réalisés par propagation dans ce voisinage, le degré émergeant est calculé pour chaque voisin du sommet, puis pour chacun de ses propres voisins etc. Le degré total émergent est la somme des degrés émergents. Les sommets traversés font alors partie de la communauté. Cette propagation continue tant que la variation de degré total émergent est supérieure à un seuil défini expérimentalement.

Le choix de ce paramètre nest pas évident, il dépend de la densité des communautés, du graphe et de la forme des communautés souhaitées. De plus, le nombre de calculs de degré émergent à réaliser est très élevé et la propagation de ces calculs peut conduire à inonder un réseau dans lequel on implémenterait cette méthode. Enfin, les sommets de départ sont définis arbitrairement, bien qu'une stratégie de sélection fine de ces sommets améliore la qualité des résultats (Riedy et al., 2011).

Un autre exemple est l'algorithme GCE (Lee et al., 2010), qui opère une expansion à partir de cliques. L'idée d'utiliser une clique comme graine est intéressante, car elle correspond à l'idée intuitive et généralement vérifiée que le cœur d'une communauté est dense. Le principal désavantage est cependant le même que pour les autres méthodes centrées-graine : un graphe de terrain contient généralement de nombreuses cliques [14], il faut donc pouvoir identifier celles qui formeront les meilleures graines.

14. On rappelle également que trouver la clique maximale, ou toutes les k-cliques d'un graphe, est NP-complet.

Hybride

Selon les mêmes principes, la méthode YASCA de (Kanawati, 2014b) remplace l'expansion par une détection des communautés locales autour des graines, puis utilise de l'*ensemble clustering* (Strehl & Ghosh, 2003) afin de choisir les meilleures.

van Laarhoven et Marchiori (2016) opèrent un mélange de détection par optimisation locale de type apprentissage automatique et par convergence de marche aléatoire. Ils utilisent une expansion à partir d'une ou plusieurs graine(s), en optimisant localement le voisinage à affecter à la communauté l'aide d'une forme relâchée de conductance, rapprochée des k-moyennes à noyaux pondérés (Dhillon et al., 2007). L'optimisation proprement dite peut être réalisée de plusieurs façons : à l'aide d'un algorithme EM ou d'une descente de gradient projeté (PGD).

Similarité

L'approche de sélection de graines de Ngonmang et al. (2012) (voir sous-section 4.2.3) et des méthodes GMAC et iGMAC (Ma et al., 2014) allient l'utilisation de nouvelles métriques de similarité de voisinage et de compacité-isolation, afin de diriger l'analyse de ce voisinage vers les zones les plus « intéressantes », réduisant de ce fait la sensibilité aux choix de cette graine.

D'autres exemples, appliqués aux communautés recouvrantes, sont traités dans la section 4.2.3, p. 95.

2.5.4 Méthodes basées-leader

Ces méthodes sont basées sur l'observation empirique dans les graphes de terrain, particulièrement dans les réseaux sociaux, que certaines entités (ex. individus) exercent une plus grande influence et en attirent d'autres utour d'eux, formant les communautés (Blondel et al., 2008a). Les premières, entités influentes, sont nommées *leaders* (meneurs) et les secondes *followers* (suiveurs). De même que pour les graines, il est nécessaire d'identifier les leaders au sein du graphe, ce qui pose la problématique de leur sélection.

Les méthodes basées sur des graines et celles basées sur des leaders sont proches dans leurs concepts, elles différent principalement sur la formation de la communauté autour du sommet central (graine ou leader) : la formation autour d'une graine utilise un mécanisme, parmi ceux cités précédemment par exemple, alors qu'une méthode leader-follower, plus spécifique, cherche à identifier deux rôles distincts parmi les sommets et à réaliser des associations entre leaders et followers pertinentes.

L'idée est que les followers suivent les leaders, un sommet doit donc être identifié à l'un de ces rôles, ce qui nécessite d'établir l'*influence* d'un sommet sur son voisinage, alors qu'elle est beaucoup moins contrainte dans les méthodes centrées graine qui sélectionnent

Algorithme 2 Leader-Follower (Shah & Zaman, 2010)

Require: $G = (V, E)$, un graphe connexe
Ensure: $C \subset \mathcal{P}(V)$, des communautés sous forme d'ensembles de sommets
1: $C \leftarrow \emptyset$
2: **for** $v \in V$ **do**
3: calculer centralité de distance $D(v)$
4: **end for**
5: définir ensemble de leaders $\mathcal{L}$ et followers $\mathcal{F}$:

$$\begin{aligned} \mathcal{L} &= \{v \in V : \exists u \in V \ / \ (u,v) \in E, D(v) < D(u)\} \\ \mathcal{F} &= V \setminus L \end{aligned}$$

6: Soit $\{v_1, v_2, ..., v_{|\mathcal{L}|}\}$ les sommets de $\mathcal{L}$ ordonnés par leur $D(v_i)$ croissante.
7: Soit $M : V \to V \cup \{\star\}$ avec $M(v) = v, \forall v \in \mathcal{F}$
8: **for** $i \in [\![1, |\mathcal{L}|]\!]$ **do**
9: $F_{vi} = \{v \in \Gamma(v_i) \cap \mathcal{F} : M(v) = \star\}$
10: $\forall v \in F_{vi}$, affecter $M(v) = v_i$ et $C_{vi} = \{v_i\} \cup F_{vi}$
11: **end for**
12: **for** each $v_i \in \mathcal{L}$ avec $C_{vi} = \{v_i\}$ **do**
13: Supprimer C_{vi} et définir $H_{vi} = \{M(v) : v \in \Gamma(v_i) \cap \mathcal{F}\}$ puis
14: **if** $H_{vi} \neq \emptyset$ **then**
15: Soit u l'élément le plus répété dans H_{vi} (plusieurs possibilités → choix aléatoire)
16: **else**
17: Soient $I_{vi} = \{M(v) : v \in \Gamma(v_i)\}$ et u l'élément le plus répété dans I_{vi} (plusieurs possibilités → choix aléatoire)
18: **end if**
19: $M(v_i) = u$ et $C_u = C_u \cup \{v_i\}$
20: **end for**

les sommets à ajouter à une communauté principalement sur des critères topologiques, par exemple le positionnement d'un sommet par rapport à une zone du graphe.

Nous détaillons ici deux méthodes : Top-Leaders (Rabbany et al., 2010) et Leader-Follower (Shah & Zaman, 2010). D'autres méthodes sont évoquées dans le chapitre 4, concernant les communautés recouvrantes, section 4.2.3, p. 95.

Une idée intuitive du rôle que doit remplir un leader a été proposée par Blondel et al. (2008a) : il constitue une exception dans son voisinage, principalement en fonction de son degré, résumé par l'aphorisme « being rich among the poor, and vice versa ». Les auteurs proposent une caractérisation possible des leaders par rapport à la distribution statistique des degrés.

Sans directement utiliser ces définitions, la méthode Top-Leaders (Rabbany et al., 2010) fonctionne de la manière suivante : elle initialise un ensemble de k leaders, puis associe les $n - k$ autres sommets du graphe (followers) à un leader chacun en utilisant une mesure de centralité (les auteurs proposent la centralité de degré, d'autres étant possibles).

Le processus se répète de façon itérative : les leaders sont mis à jour, puis chaque follower peut changer de leader s'il se trouve proche d'un nouveau leader. Les itérations se répètent jusqu'à convergence, c'est-à-dire jusqu'à ce qu'il n'y ait plus de changement de leader pour tous les followers. Une communauté est alors formée par chaque leader et son ensemble de followers.

Le fonctionnement de cet algorithme est à rapprocher de celui de k-means (Jain et al., 1999) transposé à de la détection de communautés sur graphe et offre un bon compromis d'approche, les différents stratégies impliquées (sélection des leaders, affectation aux followers, mise à jour des leaders...) pouvant être sélectionnées indépendamment en fonction des contraintes du contexte d'application souhaité. L'implémentation peut par exemple être totalement locale et décentralisée, en choisissant des stratégies adéquates. Des travaux d'amélioration de k-means peuvent également être repris et adaptés pour cet algorithme.

Leader-follower (Shah & Zaman, 2010), présenté dans l'algorithme 2, p. 62, de son côté constitue l'ensemble des leaders à l'aide des centralités de distance des sommets du graphe. Les leaders sont les sommets dont la centralité de distance est moins que celle d'au moins un de leurs voisins. Les autres sont automatiquement considérés comme followers. Les communautés sont ensuite formées en associant les followers au leaders à l'aide d'une fonction d'appartenance.

2.6 Bilan

En conclusion, après avoir présenté quelques précisions nécessaires sur les définitions utilisées, nous avons présenté les méthodes classiques de fouilles de graphes précurseures des méthodes de détection de communautés. Nous avons introduit quelques caractéristiques permettant de classer ces dernières, puis nous avons présenté, de pair avec des exemples représentatifs, les principales familles de méthodes en lien avec la thèse, en particulier les méthodes orientées-sommet, famille dont fait partie l'algorithme que nous proposons dans le chapitre suivant, nommé VOLCAN.

Chapitre 3

VOLCAN : détection de communautés disjointes

Sommaire

> Robust community detection need not employ complicated algorithms
>
> *(Gleich & Seshadhri, 2012)*

Ce chapitre présente l'algorithme VOLCAN (**V**ertex-**O**riented **L**ocation of **C**ommunities based on **A**greement in **N**etworks), qui détecte des communautés dans un graphe, en utilisant une analyse locale du voisinage des sommets compatible avec une implémentation du type Think-Like-a-Vertex (cf. section 2.5.1.1, p. 54).

Le chapitre est organisé de la manière suivante : tout d'abord sont décrits les objectifs qui ont motivé le développement de VOLCAN, puis sont exposés les principes sous-jacents utilisés pour tenir compte de ce contexte, dans la section 3.1. Une description de l'algorithme est donnée dans la section 3.2, ainsi que de la mesure de préférence utilisée : l'*agreement*. Nous discutons ensuite l'aspect algorithmique en précisant les possibilités d'une implémentation totalement décentralisée dans la section 3.3. Une étude théorique est proposée section 3.4, comprenant une caractérisation des communautés trouvées ainsi que des discussions sur plusieurs propriétés usuelles : complexité, déterminisme et propagation. Enfin une étude expérimentale est présentée section 3.5.

Ces travaux ont fait l'objet de la publication Canu et al. (2015).

3.1 Objectifs et caractéristiques

La conception de VOLCAN a été motivée par une application de la détection de communautés ayant pour particularité d'anticiper, dans sa conception, une implémentation décentralisée, qui puisse être exécutée sur un réseau d'objets connectés mobiles, de type ad-hoc opportuniste, de type OMSN (cf. section 1.4, p. 30). En effet, les communautés détectées sur de tels réseaux peuvent être utilisées afin d'améliorer potentiellement le routage (voir section 1.4, p. 30).

Cette section présente tout d'abord les principes de fonctionnement retenus préalablement à la conception de VOLCAN, afin de lui permettre d'intégrer des propriétés *by design*, dans la sous-section 3.1.1, puis la forme des communautés qu'il vise à identifier, dans la sous-section 3.1.2.

3.1.1 Principes

L'exploitation d'un réseau OMSN impose de nombreuses contraintes, en particulier le coût de calcul algorithmique ainsi que la quantité de propagation des données sont prépondérants. Idéalement, dans une implémentation décentralisée d'un algorithme de détection de communautés pour ce type de réseaux, chaque dispositif mobile échange des informations minimales en utilisant des données fournies uniquement par ses plus proches voisins

(aspect local et décentralisé), et la structure communautaire est calculée en propageant le moins d'informations possible.

Avec VOLCAN, nous montrons qu'en analysant des voisinages immédiats de sommets de façon simple, il est possible de trouver des communautés de bonne facture, avec l'idée de minimiser le coût algorithmique, de limiter la propagation et d'obtenir une vitesse d'exécution rapide, par rapport à la plupart des algorithmes existants.

VOLCAN utilise pour cela le principe leader-follower (cf. section 2.5.4, p. 61), avec la particularité de ne pas imposer de choix de leaders, qui est le principal inconvénient de cette famille de méthodes. Nous détaillons ce point dans la section 3.2, p. 68 concernant la description de l'algorithme. La nouveauté apportée par VOLCAN par rapport aux méthodes existantes est sa conception basée sur le paradigme Think-Like-a-Vertex : l'algorithme est ainsi écrit comme un programme à exécuter par chacun des sommets, qui prennent en entrée uniquement des données sur leur voisinage. Il pourrait ainsi être implémenté dans un des outils de développement TLAV existants (McCune et al., 2015).

Plus précisément, VOLCAN dispose des caractéristiques suivantes, qui seront également détaillées dans la description de l'algorithme :

une détection par similarité locale afin de modéliser le principe d'assortativité et d'attachement préférentiel, VOLCAN utilise une mesure de préférence entre sommets de même voisinage (voir section 3.2.2, p. 69), assimilable à une mesure de similarité, afin de former des groupes de sommets à la base des communautés.

une implémentation décentralisable, ne nécessitant pas d'entité centrale qui assurerait la communication entre les différents dispositifs constituant le réseau : chaque dispositif communique avec son voisinage et un échange d'informations minimal suffit à constituer la structure communautaire.
En effet, dans un réseau ad-hoc, il n'est pas possible pour un nœud de recevoir directement des informations d'autres nœuds qui ne sont pas dans son voisinage immédiat. Dans ce cadre, découvrir une caractéristique globale telle que la structure communautaire se révèle plus complexe que dans un réseau à architecture centralisée.

la limitation de la propagation de messages : l'échange d'informations mentionné précédemment est contrôlé et réalisé avec parcimonie afin d'éviter l'inondation de messages dans le réseau lors de l'exécution.
Cet objectif est étroitement lié au précédent, en effet la propagation d'information dans un réseau pair-à-pair est coûteuse pour les dispositifs : chaque message diffusé doit être reçu et réémis par tous les nœuds du réseau, jusqu'à réception par l'ensemble des nœuds. Dans un réseau de taille n, lorsque chaque nœud émet un message (simultanément par l'ensemble des nœuds par exemple), le nombre total de messages émis est donc de l'ordre de $\mathcal{O}(n \times n)$, chaque nœud envoyant un message et relayant les $n-1$ autres provenant des autres nœuds du réseau.

3.1.2 Forme des communautés

VOLCAN est une méthode agrégeant des paires de sommets leaders avec leurs followers. Comme nous l'avons mentionné ci-dessus et le détaillons dans la section 3.2.1, VOLCAN associe les followers aux leaders à l'aide d'une mesure de préférence entre sommets σ dont le but est de cerner des zones localement denses dans le graphe. Toutes les mesures de similarité entre sommets d'un même voisinage (Cohen et al., 2012) peuvent être utilisées, cependant nous en proposons une nouvelle, nommée *agreement*, basée sur différents principes, propriétés et résultats détaillés dans la section afférente (3.2.2.2, p. 71). Nous considérerons par la suite une mesure de préférence :

$$\sigma : V \times V \to \mathbb{R}^+$$

avec $\sigma(u, v)$ exprimant la préférence du sommet u pour le sommet v.

Ainsi, nous ne restreignons pas les communautés à une définition donnée a priori, qui pourrait être contraignante. Parmi les deux types d'approches distingués section 1.3.2, p. 28, VOLCAN suit celle n'utilisant pas de définition de communauté a priori : nous proposons une méthode de recherche basée sur des préférences entre sommets et des propriétés des graphes de terrain et réalisons une caractérisation a posteriori des communautés trouvées, dans la section 3.4, p. 76.

3.2 Description de l'algorithme

Cette section décrit l'algorithme VOLCAN, ainsi que la mesure *agreement* qu'il utilise. Dans une première sous-section 3.2.1 nous présentons une vue d'ensemble de l'algorithme, puis nous présentons dans la sous-section 3.2.2 différentes mesures de préférence entre sommets, dont la mesure d'agreement que nous proposons. Nous exposons ensuite en détails les deux étapes de l'algorithme : choix du leader et agrégation par fusion, dans les sections 3.2.3 et 3.2.4 respectivement.

3.2.1 Vue d'ensemble

VOLCAN est composé de deux étapes principales, qui s'inscrivent dans le schéma d'une méthode de détection orientée sommet leader-follower, basée sur l'exploration du voisinage d'un sommet v. La principale différence avec les méthodes leader-follower existantes est que, dans VOLCAN, chaque sommet peut être à la fois leader *et* follower.

Ces étapes sont :

1. Sélection d'un voisin préféré pour chaque sommet v, son leader, noté a_v : on calcule la préférence de v pour chacun des sommets de son voisinage immédiat, et a_v est le voisin maximisant cette préférence.

Toute mesure de préférence peut être utilisée, mais l'efficacité globale de VOLCAN est améliorée en utilisant une nouvelle mesure d'*agreement* proposée (sous-section 3.2.2.2 ci-dessous), basée sur les voisins de plus haut degré des sommets. On peut alors définir un nouveau graphe, orienté, que nous nommons *graphe des préférences* $G_p = (V_p, \mathcal{A})$, formellement défini et exploité dans l'étude théorique de VOLCAN, section 3.4.1, p. 77.

2. Agrégation par fusion des communautés : chaque sommet forme sa propre communauté, puis pour chaque paire $(v,\ a_v)$, les communautés $C(v)$ et $C(a_v)$ sont fusionnées.

Ainsi, chaque sommet v choisit un leader dont il rejoint la communauté. Pour ce faire, il recense des leaders potentiels (candidats) et choisit parmi eux le sien, leurs communautés fusionnant ensuite. VOLCAN propose une procédure de choix des candidats par mesure de préférence, dont différents exemples sont donnés dans la section suivante 3.2.2, ainsi qu'une procédure de choix du leader décrite dans la section 3.2.3, p. 72.

3.2.2 Mesures de préférence entre sommets

Le principe de la méthode proposée repose sur une mesure de préférence σ entre un sommet et les sommets de son voisinage. Nous présentons tout d'abord quelques mesures existantes, puis la mesure que nous proposons pour correspondre le mieux possible à la tâche de détection de communauté, l'*agreement*.

3.2.2.1 Mesures de préférence existantes

Nous distinguons ici deux grandes familles de mesures de préférence entre sommets : celles basées sur les marches aléatoires et celles basées sur l'analyse du voisinage. Ces mesures sont référencées par Liben-Nowell et Kleinberg (2007) et Cohen et al. (2012).

Mesures basées sur des marches aléatoires

Comme nous l'avons vu dans le chapitre 2, section 2.5.2, p. 58, les marches aléatoires sur graphes sont des processus efficaces pour l'exploration de grands graphes. En analysant les probabilités de convergence d'une telle marche (temps de *mixage*, cf. section 2.5.2, p. 58) on peut en déduire des informations sur le voisinage d'un sommet. Entre autres, le temps de mixage en commençant à un sommet source s pour s'arrêter à un sommet cible t produit une valeur numérique pouvant être utilisée comme mesure de préférence.

On trouve donc comme mesures basées sur des marches aléatoires des estimations utilisées dans divers algorithmes cités dans le chapitre 2, section 2.5, p. 53, entre autres ceux de Spielman et Teng (2004) et Andersen et Lang (2006), Orecchia et Zhu (2014).

D'autres travaux (Andersen et al., 2006, Gleich & Seshadhri, 2012, Zhu et al., 2013) reposent sur les mêmes propriétés, mais utilisent le *PageRank personnalisé* à la place du temps de mixage (cf. section 2.5.2, p. 58).

Le problème de ces mesures est qu'elles reposent essentiellement sur des équations non locales à un voisinage de sommet, elles sont difficilement adaptables dans le contexte que nous considérons : l'estimation de la convergence d'une marche, et donc de son temps de mixage, dépend d'informations sur la topologie du voisinage du sommet source. Plus l'on souhaite une estimation précise, plus la taille de ce voisinage doit être large. Or, il est complexe de réaliser un calcul utilisant des informations d'une large portion de graphe dans un contexte totalement décentralisé en limitant la propagation. C'est pourquoi nous présentons également ci-dessous des mesures basées sur l'analyse du voisinage immédiat du sommet source.

Mesures basées sur l'analyse du voisinage immédiat

Liben-Nowell et Kleinberg (2007), Cohen et al. (2012) et Kanawati (2014a) recensent plusieurs mesures de similarité entre sommets facilement calculables et implémentables. Nous présentons les principales ci-dessous, en notant u et v deux sommets d'un graphe. La plus basique évalue le nombre de voisins communs :

$$\sigma_{ComN}(u,v) = |\Gamma(u) \cap \Gamma(v)| \tag{3.1}$$

D'autres utilisent des normalisations ou des exploitations plus complexes.

Le coefficient de Jaccard (1901) adapté pour les ensembles de sommets est défini de la façon suivante :

$$\sigma_{Jacc}(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{3.2}$$

Il compare le nombre de voisins communs à sa valeur potentielle maximum, définie par le cardinal de l'union des voisinages. Il permet de définir une mesure relative alors que le nombre de voisins est une mesure absolue.

La mesure d'Adamic et Adar (2003) adaptée aux ensembles de sommets s'écrit :

$$\sigma_{AA}(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log(|\Gamma(w)|)} \tag{3.3}$$

L'utilisation de l'inverse du logarithme donne un poids plus fort aux sommets partageant des voisins de faible degré. Ainsi, la mesure rapproche des sommets ayant des voisinages plus atypiques au sein de leur communauté.

Une version sans le logarithme, pénalisant moins les voisins communs de haut degré, a été proposée par Zhou et al. (2009a) :

$$\sigma_{AAb}(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{|\Gamma(w)|} \tag{3.4}$$

La mesure d'attachement préférentiel est la transcription, pour les voisinages de sommets, du principe de l'attachement préférentiel, évoqué dans le chapitre 1, section 1.2.3.6, p. 19. Sa valeur est le produit des degrés des deux sommets :

$$\sigma_{PA}(u,v) = |\Gamma(u)| \times |\Gamma(v)| = d_u \times d_v \tag{3.5}$$

Ainsi, un sommet préfère un voisin de (plus) haut degré. L'utilisation de cette mesure produit des agglomérats de sommets de hauts degrés, formant de larges *hubs* en particulier dans les graphes de terrain sans échelle (cf. section 1.2.3.6, p. 19).

3.2.2.2 Agreement

Nous avons choisi d'introduire une méthode de préférence entre sommets nommée *agreement*. Son but est de détecter des zones denses à l'aide d'information locale, à savoir les voisins de hauts degrés de chaque sommet. L'agreement ne prend que des valeurs entières positives ou nulle.

Nous constituons pour ce faire une liste S_v pour chaque sommet $v \in V$ du graphe. L'idée est que cette liste soit représentative des voisins de v avec lesquels il est le plus susceptible de former une communauté, on a donc $S_v \subset \Gamma(v)$.

Nous nous basons sur le principe lié à l'attachement préférentiel connu sous le nom de « phénomène du club des riches » (*rich-club phenomenon*, aussi *"richs get richer"*, décrit dans la section 1.2.3.6, p. 19), afin de calculer une préférence tenant compte à la fois des degrés des voisins du sommet considéré, mais également le nombre de ces voisins communs.

Constitution des listes S_v

Nous remplissons la liste S_v avec les k_v voisins de v de plus haut degré. Formellement, nous utilisons un paramètre entier k_v qui fixe la taille de S_v comme discuté ci-dessous et notons $\Gamma(v) = \{u_1, ..., u_{d_v}\}$. Nous utilisons également une fonction d'ordre f_v qui ordonne les voisins par degré décroissant, c'est-à-dire telle que $f_v(i)$ est l'indice du $i^{\text{ème}}$ voisin de plus haut degré de v. S_v est alors définie par :

$$S_v = \{u_{f_v(1)}, ..., u_{f_v(k_v)}\}$$

Si un choix est nécessaire parmi des sommets de même degré, un tirage aléatoire est effectué. Par exemple, si v est un sommet ayant 5 voisins $u_{1...5}$ de degrés respectifs 3, 2, 2, 2 et 1

Algorithme 3 Vertex Program : calcul de l'*agreement*, choix du leader

Require: sommet v
Ensure: sommet a_v, leader d'un sommet v
1: $a_v \leftarrow \arg\max\{\text{agreement}(u, v) | u \in \Gamma(v)\}$

et si k_v est fixé à 3, alors S_v est constitué de u_1 et de deux sommets tirés aléatoirement parmi les sommets $\{u_2, u_3, u_4\}$.

Calcul de l'*agreement*

Muni des listes S calculées précédemment, l'*agreement* est calculé entre deux sommets voisins u et v, $u \in \Gamma(v)$ par :

$$\text{agreement}(u, v) = |S_u \cap S_v| \tag{3.6}$$

Cette mesure de préférence est alors élevée si u et v ont de nombreux voisins en commun, en tenant de plus compte des degrés de ceux-ci : l'agreement n'est pas calculé comme le cardinal de l'intersection des voisinages $|\Gamma(u) \cap \Gamma(v)|$, sauf pour le cas $k_v = d_v$ décrit ci-dessous, mais effectue une sélection des voisins considérés, selon leurs degrés.

Valeur de k_v

Le paramètre k_v, dont la valeur peut différer pour chaque sommet v, a été introduit afin de ne pas fausser le calcul de l'*agreement*. En effet, une valeur $k_v = 1$ n'est pas souhaitable : v choisirait son voisin préféré a_v comme étant nécessairement son voisin de plus haut degré. Or, il n'est pas rare que ce voisin de plus haut degré fasse partie d'une communauté différente, particulièrement s'il est situé dans la bordure de cette communauté.

D'un autre côté, considérer l'intégralité d'un voisinage, i.e. $k_v = d_v$, est inutile et revient à définir $S_v = \Gamma(v)$. En nous situant dans les conditions de Gleich et Seshadhri (2012) [15] (cf. section 2.5.1.2, p. 57), la probabilité que les triplets existants entre v et ses voisins de haut degré soient fermés, entre d'autres termes soient des triangles, est très forte. Leurs valeurs sont, dans ce cas, moins différenciées et la pertinence du choix du leader peut en être impactée.

Expérimentalement nous fixons $k_v = \lceil d_v/2 \rceil$, une valeur que nous avons constatée comme expérimentalement satisfaisante.

3.2.3 Etape 1 : choix du leader

Cette étape a pour but d'apparier les sommets les plus susceptibles de former une communauté, action réalisée à l'aide de la mesure de préférence σ basée sur les voisinages

15. Distribution des degrés en queue lourde et haut coefficient de clustering.

Algorithme 4 Fusion des communautés

Require: ensemble de sommets V,
Ensure: ensemble de communautés $\mathcal{C}$
1: $\mathcal{C} = \{\{v\} | v \in \mathcal{V}\}$
2: **pour chaque** v non fusionné
3: fusion$(C(v), C(a_v))$

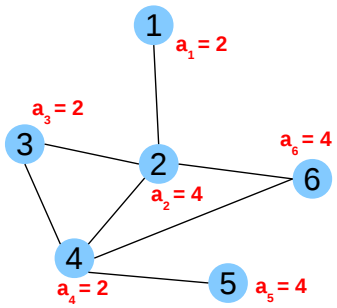


FIGURE 3.1 – Illustration du choix des leaders.

de ces sommets discutée précédemment. Chaque sommet v calcule les préférences avec tous ses voisins, $\sigma(v, u), u \in \Gamma(v)$. Le leader a_v associé comme sommet préféré à v est ensuite choisi comme le voisin maximisant σ :

$$a_v = \arg\max_{u \in \Gamma(v)} \{\sigma(v, u)\} \tag{3.7}$$

Si plusieurs valeurs sont retournés par l'arg max une seule est retenue tirée aléatoirement de façon uniforme.

VOLCAN exploite, par la définition du leader a_v, la mesure d'agreement présentée ci-dessus. Nous avons constaté expérimentalement que les mesures existantes listées précédemment ne donnent pas des résultats satisfaisants (voir section 3.5.1, p. 80). En particulier, la mesure directement basée sur l'attachement préférentiel est trop « forte » et constitue des agrégats massifs ne correspondant pas aux communautés réelles, parfois même une seule communauté pour tout le graphe.

L'algorithme 3, p. 72 formalise ce processus, qui s'inscrit dans le cadre d'une conception orientée-sommet de type TLAV et constitue un exemple de Vertex Program.

La figure 3.1 illustre le choix du leader : deux leaders sont présents dans le graphe, les sommets 2 et 4, choisis respectivement par les sommets 1, 3, 4 et 2, 5, 6. On remarque les sommets 3 et 6 ont choisi un leader différent, ayant pourtant le même voisinage constitué des sommets 2 et 4, tous les deux de degré 4.

Cet effet se produit dans deux cas de figure :

— avec un paramètre $k_v = 1$, chaque sommet doit choisir un de ses deux voisins à mettre dans sa liste S_v. Les deux voisins ayant le même degré, l'un des deux est

choisi aléatoirement pour constituer la liste (le sommet 2 pour la liste du sommet 3, le sommet 4 pour être celle du sommet 6), devenant le leader.

— avec un paramètre $k_v = 2$, la liste S_v est composée des deux sommets 2 et 4, mais étant de même degré un tirage aléatoire est effectué pour choisir celui qui devient effectivement le leader.

Le critère d'agreement étant le seul retenu pour élire un leader, chaque sommet est éligible au titre de leader d'un autre sommet. Un sommet peut donc être à la fois leader et follower. Ce principe permet de former des communautés, comme discuté dans la section 3.4.1, p. 76.

3.2.4 Etape 2 : agrégation par fusion

L'étape de fusion est similaire à la plupart des étapes de fusion dans les algorithmes de détection agglomératifs (Kanawati, 2014a) : initialement, chaque sommet v forme sa propre communauté, puis les communautés fusionnent deux à deux jusqu'à ce que tous les sommets v soient dans la même communauté que leur leader a_v. Ainsi, le sommet v fusionne avec son sommet leader a_v, puis ce sommet leader fusionne également avec son propre leader, etc., comme indiqué dans l'algorithme 4, p. 73.

Cette méthode de fusion conserve la vue locale de la communauté : en effet un sommet ne peut pas choisir sa communauté, en tout cas pas au-delà du choix du leader déjà opéré à ce stade. Sa communauté dépend de la hiérarchie des leaders au-dessus de lui, suivant le modèle d'attachement préférentiel. On vise ainsi l'obtention des cœurs de communautés denses, formés par des sommets de hauts degrés se préférant mutuellement, et de petites chaînes de leaders se préférant successivement, formant les *whiskers* constatés par Leskovec et al. (2008).

Un des avantages de la fusion est qu'elle n'est pas sensible à l'ordre : cette étape permet d'obtenir des communautés identiques à partir du même ensemble de préférences inter-sommets.

La section suivante décrit l'implémentation de cette fusion dans un contexte décentralisé. La section 3.4, p. 76 examine d'un point de vue théorique les propriétés que possèdent les communautés identifiées par ce processus.

3.3 Implémentation décentralisée

Cette section discute de l'implémentation de VOLCAN, afin de pouvoir l'utiliser dans un environnement réellement décentralisé du type TLAV (cf. section 2.5.1.1, p. 54). La sous-section 3.3.2 traite notamment du de l'étape de fusion.

3.3.1 Choix du leader dans un environnement décentralisé

Nous décrivons ci-dessous une proposition de protocole d'implémentation décentralisée de VOLCAN pour la première étape, le choix du leader :

— Initialisation – chaque sommet v, assimilé à un dispositif mobile communicant sans fil, découvre ses voisins : les autres dispositifs à portée de communication sans fil.
— Échange des degrés – chaque sommet v envoie à ses voisins son degré, c'est-à-dire le nombre de voisins qu'il a précédemment découverts. Il envoie donc au plus d_v messages et en reçoit théoriquement autant.
— Calcul de la liste S_v – chaque sommet v calcule sa liste S_v en utilisant les degrés de ses voisins.
— Échange des listes – chaque sommet v envoie sa liste S_v à ses voisins, générant à nouveau d_v messages.
— Calcul de l'*agreement* – chaque sommet v calcule son *agreement* avec chacun de ses voisins.
— Sélection de a_v – chaque sommet v sélectionne son leader a_v.

La sous-section suivante décrit une proposition d'implémentation pour la seconde étape de l'algorithme : la fusion.

3.3.2 Fusion décentralisée

Si les deux premières étapes sont faciles à décentraliser de par leur conception même, il n'en va pas de même pour la fusion. En effet, ce processus est itératif, impliquant de plus en plus de sommets du graphe à chaque étape (et donc de dispositifs dans le réseau mobile) : chaque sommet forme initialement sa propre communauté, puis fusionne avec celle de son leader. Les deux sommets ont donc besoin de savoir qu'ils sont maintenant dans la même communauté. Au fur et à mesure des fusions, le nombre de sommets impliqués croît très rapidement.

Nous proposons un mécanisme de propagation/rétropropagation reposant sur la propriété 3.4.1, p. 77, qui permet d'identifier les circuits dans le graphe des leader/follower aux communautés.

1. Identification des leaders : chaque sommet vérifie si au moins un de ses voisins l'a choisi comme leader.
2. Identification des circuits - propagation : chaque sommet qui est *uniquement follower*, c'est-à-dire tel qu'aucun autre ne l'a choisi comme leader, envoie un message *token* muni d'un identifiant unique, à faire suivre de sommet en sommet jusqu'à ce qu'un sommet indique avoir déjà reçu un token portant cet identifiant (boucle du circuit).

A ce stade, les sommets de chaque branche raccordée à un circuit possèdent un identifiant numérique unique, et les sommets appartenant à un circuit possèdent les identifiants de toutes les branches raccordées au circuit.

3. Identification des communautés - rétropropagation : un ou plusieurs sommet(s) appartenant à chaque circuit (car dernier destinataire d'un token) envoie(nt) un token portant l'identifiant de valeur minimale, parmi celles reçues à l'étape précédente, à son ou ses follower(s), qui ne retiennent désormais plus que cet identifiant et le rétropropagent. Un sommet ayant plusieurs followers rétropropage bien sûr à tous ses followers.

   Une fois tous les sommets sans follower atteints, chaque sommet ne connaît plus qu'un identifiant, qui est l'identifiant de sa communauté.

Cet algorithme converge mais son temps de stabilisation (y compris temps de chaque phase séparée) ne peut pas être connu des dispositifs, ainsi son implémentation nécessite de fixer des durées d'attente afin d'éviter d'être bloquée au cas où une (rétro)propagation échoue.

3.4 Etude théorique

VOLCAN n'est pas basé sur une définition a priori des communautés, mais sur un processus d'extraction (cf. section 1.3.2, p. 28). Aussi, l'analyse théorique consiste d'abord à caractériser les communautés identifiées, comme décrit dans la section 3.4.1. Les sections suivantes sont consacrées aux propriétés usuelles de complexité, déterminisme et stabilité ainsi qu'à la propagation.

3.4.1 Caractérisation des communautés détectées

Le processus d'identification mis en œuvre par VOLCAN, tel que décrit dans la section précédente, définit une communauté comme un ensemble de sommets constituant tous les préférés, c'est-à-dire clos par la relation de préférence. Nous fournissons ici une caractérisation de ces ensembles, définis selon les circuits du graphe des préférences décrit plus bas.

Rappel des notations : on considère un graphe $G = (V, E)$ simple complètement connecté, non-orienté sans boucle, non-valué. On suppose que G comporte une structure communautaire, ie. V peut être partitionné en un ensemble de communautés $C = \{c_1, ..., c_{|C|}\}, c_i \subset V$ et $\bigcup_i c_i = V$. On note $\Gamma(v) \subset V$ l'ensemble des voisins de $v \in V$. On note $a : V \to V$ la fonction qui à tout sommet v associe son sommet préféré a_v.

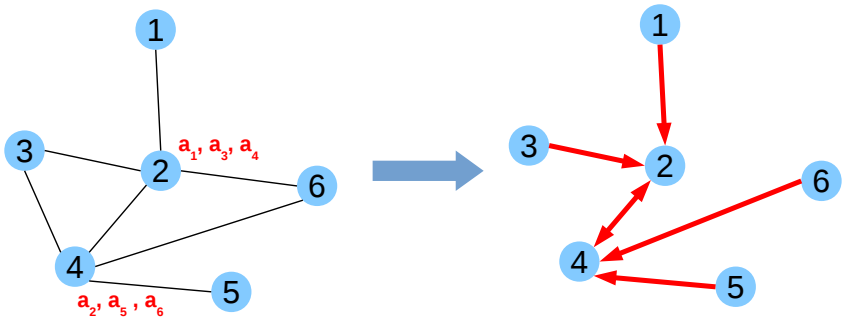


FIGURE 3.2 – Passage d'un graphe G et de préférences a_i (à gauche) vers un graphe orienté des préférences G_p (à droite)

Lien entre graphe des préférences et communautés formées

Dans cette sous-section, nous présentons le lien entre le graphe orienté créé par VOLCAN, notamment les circuits qu'il contient, identifiés à des communautés du graphe originel.

Graphe des préférences G_p

Etant donné un graphe $G = (V, E)$ analysé par VOLCAN, le graphe est préférences est le graphe orienté $G = (V, A)$ où A est défini comme $A = \{(v, a_v), v \in V\}$. Un tel graphe est illustré figure 3.2 : à gauche G et les deux sommets leaders identifiés, les numéros 2 et 4, constituant les $a(i)$) de leurs followers i, indiqués en rouge. A droite, le graphe orienté correspondant.

Deux propriétés sont à noter :

Propriété 3.4.1. *G_p possède au moins un circuit.*

Démonstration *G_p n'est pas un arbre puisque, par construction, chaque sommet dispose d'un arc sortant et G_p est fini.* ∎

Propriété 3.4.2. *Dans tout chemin de $G_p(v_1 \to ... \to v_{p+1})$ composé de $p \in \mathbb{N}^*$ arcs soit $p+1$ sommets, on a $\forall i \in [\![1, p+1]\!], v_i = a^{i-1}(v_1)$*

Démonstration *par récurrence immédiate puisque $\forall i, v_i = a(v_{i-1})$ par définition de G_p.* ∎

Structure des communautés

On peut alors caractériser les communautés identifiées par VOLCAN selon les circuits minimaux (cf. section 1.1.2, p. 8) de G_p :

Théorème 3.4.3. *Soit $c \in C$ de G une communauté de G identifiée par VOLCAN. Il existe un unique circuit minimal de G_p tel que*

$$c = V(l) \ \cup \ \{u \in V \mid \exists u' \in V(l) tel\ qu'il\ existe u_i \in V, (u \to u_{1+1}), ..., (u_p \to u') \subseteq A\}$$

Démonstration *On note branches(l) l'ensemble des sommets desquels partent une chaîne vers l'un des sommets de l. Formellement, branches(l) = $\{u \in V \mid \exists u' \in V(l)$ tel qu'il existe $u_i \in V, (u \to u_{1+1}), ..., (u_p \to u')] \subseteq A\}$.*

Soit $v \in c$.

Comme G_p est fini et que chaque sommet possède un unique arc sortant, il existe nécessairement un unique circuit minimal l auquel on peut arriver en partant de v. Soit $w \in V(l)$ tel qu'il existe une suite de sommets $v_1, ..., v_p$ telle que $\{sv \to v_1, ..., v_p \to w\} \subseteq A$. Cette suite est éventuellement vide si v appartient au circuit. On a alors $w = a^{p+1}(v)$, par la propriété 3.4.2.

D'après la procédure de fusion définie par VOLCAN, $w \in c$.

Pour tout $u \in V(l)$, on a $u \in c$. En effet, dans le circuit l, il existe une chaîne entre w et u car il existe q tel que $u = a^q(w)$ donc u appartient à la même communauté que w. On a ainsi que $V(l) \subseteq c$.

Pour tout $u \in$branches(l) en notant u' le sommet de $V(l)$ lié par une chaîne à u, il existe p tel que $u' = a^q(u)$ donc d'après la procédure de fusion de VOLCAN, u' et u sont dans la même communauté. Comme $u' \in V(l)$, $u' \in c$ d'où $u \in c$. On a ainsi que branches(l) $\subseteq c$.

Enfin, il n'y a pas d'autre sommet dans c : en effet si $u \notin V(l) \cup$branches(l) il n'existe pas de chemin entre u et w. Ces deux sommets ne sont donc pas affectés à une même communauté.

Théorème 3.4.4. *Dans G_p, $|C| = |L|$.*

Démonstration *D'après le théorème 3.4.3, à chaque communauté correspond un unique circuit (minimal). Réciproquement, si l est un circuit minimal, alors les sommets qui le composent sont affectés à une même communauté, comme montré dans la démonstration précédente : à tout circuit correspond donc une unique communauté.*

3.4.2 Complexité

La complexité temporelle de VOLCAN est, comme pour nombre de méthodes de détection, difficile à établir (Fortunato, 2009).

Le Vertex Program de VOLCAN (algorithme 3) réalise, pour un sommet v, une opération de tri sur ses voisins dont le coût dépend donc de son degré : elle est de complexité $\mathcal{O}(d_v \log d_v)$. v effectue également qu'une comparaison de sa liste S_v avec celles de ses voisins, de complexité $\mathcal{O}(d_v \cdot k_v)$ (k_v étant la taille de S_v). La complexité totale de la première étape peut ainsi être estimée à $\mathcal{O}(d_v^2)$, dans le pire cas.

La complexité de la fusion dépend de la taille des groupes d'interpréférences (ensembles déterminés dans la propriété 3.4.1, p. 77), donc des communautés, soit $\mathcal{O}(|\bar{c}|)$ en moyenne, où $|\bar{c}|$ est la taille moyenne d'une communauté. La complexité moyenne estimée au final est donc de $\mathcal{O}(\bar{d}^2 + |\bar{c}|)$ où $\bar{d}$ est le degré moyen du graphe.

3.4.3 Déterminisme, stabilité

Nous avons empiriquement constaté que les résultats de VOLCAN sont stables, au regard de l'évaluation par les critères de qualité choisis (section 3.5, p. 79). La fusion étant une action symétrique, l'étape d'agrégation ne dépend pas de l'ordre et les communautés produites avec un même ensemble de leaders sont identiques.

Le choix du leader, cependant, admet un non-déterminisme, en effet plusieurs voisins d'un sommet v peuvent maximiser l'*agreement*. Dans ce cas, a_v est choisi aléatoirement parmi eux avec une probabilité uniforme. La stabilité des communautés constatée expérimentalement (cf. section 3.5, p. 79) laisse néanmoins à penser que ce cas se présente rarement.

3.4.4 Propagation

Nous présentons ici une estimation du gain de VOLCAN en terme d'envoi de messages (assimilables à des paquets réseau). En effet, dans le contexte d'une exécution de cet algorithme dans un réseau décentralisé, chaque information nécessaire à un calcul et non disponible sur un dispositif est obtenue en la demandant à un autre dispositif qui la détient. Cette obtention nécessite l'envoi et la réception de messages via des paquets dans le réseau. Pour des raisons de simplification nous considérons ici les échanges en tant qu'entité unitaire, et non selon le nombre de paquets réseau les composant.

Chaque sommet envoyant uniquement des informations à ses voisins, qui lui en envoient en retour, on peut estimer à $2n\bar{d}$ messages de taille k le volume échangé, $\bar{d}$ étant le degré moyen du graphe. Tous les envois de messages dans l'algorithme ayant un volume analogue, le volume global peut être estimé en moyenne à $\mathcal{O}(n\bar{d})$.

A contrario, un algorithme à propagation de labels classique, par inondation, envoie un label de chaque sommet vers chaque autre du graphe, passant par chaque arête donc. On peut estimer le nombre moyen de messages échangés dans ce cas comme étant de l'ordre de $\mathcal{O}(nm)$. Pour un graphe de 5000 sommets avec $\bar{d} = 10$, le nombre total de messages échangés est dix fois inférieur pour VOLCAN.

3.5 Etude expérimentale

Nous présentons dans cette section une étude composée de cinq expériences visant respectivement à évaluer les effets des différentes mesures de préférence entre sommets, décrites dans la section 3.2.2, p. 69, mesurer les performances de VOLCAN, sa sensibilité

aux paramètres ainsi qu'à le comparer à des méthodes de l'état de l'art. Nous avons pour cela réalisé une implémentation de test en Java.

Nous utilisons comme jeux de données des graphes réels et artificiels (type Clauset et LFR), présentés en annexe B, p. 185, générés de pair avec une proposition de structure communautaire de référence, appelée vérité terrain. L'évaluation consiste en la mesure de l'adéquation des résultats de VOLCAN avec cette vérité terrain à l'aide des critères de qualité ARI et NMI, décrits en annexe A, p. 177.

3.5.1 Etude expérimentale des mesures de préférence

Nous proposons tout d'abord une expérience visant à comparer les performances de VOLCAN selon la mesure de préférence sélectionnée.

Protocole expérimental

Nous considérons l'algorithme VOLCAN, dans cinq variantes ne différant que par la mesure utilisée dans le calcul de la préférence entre deux sommets, en considérant les mesures décrites dans la section 3.2.2, p. 69 : le nombre de voisins communs ($ComN$), la mesure de Jaccard ($Jacc$), la mesure d'Adamic-Adar (AA), la mesure d'attachement préférentiel (PA) et l'agreement.

Ces variantes sont testées sur deux graphes différents : l'un généré par le benchmark LFR, avec pour paramètres $n = 1000$, $\mu = 0.1$, l'autre étant le graphe réel du club de karaté de Zachary (voir annexe B, p. 185).

L'expérience a été réalisée 100 fois pour chaque mesure sur chaque graphe, les moyennes des valeurs de NMI sont présentées.

Résultats obtenus

Les résultats obtenus sont présentés figure 3.3. On observe que VOLCAN muni de l'agreement produit une meilleure NMI qu'avec les autres mesures pour chacun des deux graphes. On remarque par ailleurs que les mesures des voisins communs, de Jaccard et d'Adamic-Adar sont particulièrement mal adaptées au club de karaté de Zachary. Nous pouvons donc conclure à la pertinence de l'agreement par rapport à d'autres mesures envisageables dans l'implémentation de VOLCAN.

3.5.2 Graphes artificiels

Nous présentons ici deux expériences de comparaison avec des approches de l'état de l'art dont le mode de fonctionnement est éloigné de LOCNeSs : elles sont basées sur une optimisation de critère. Nous commençons par la méthode locale de Clauset puis continuons avec les méthodes de Louvain et InfoMap, toutes les deux globales.

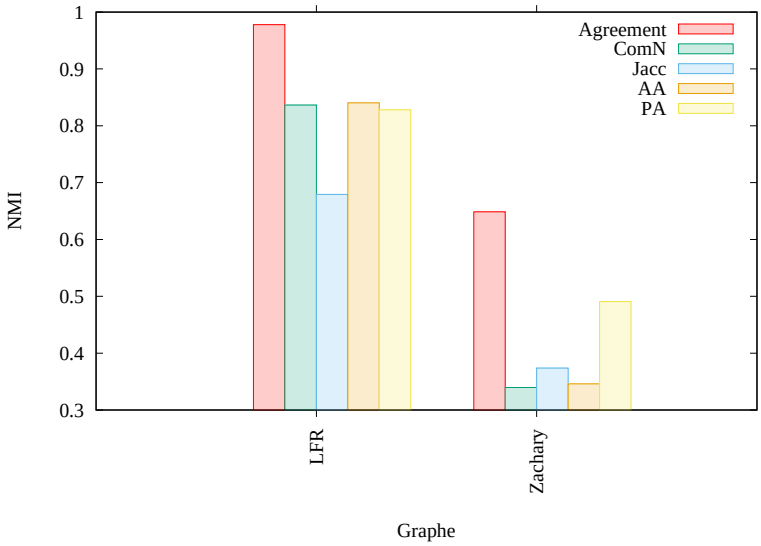


FIGURE 3.3 – NMI moyenne pour différentes mesures de préférence (couleurs) et différents graphes (abscisses)

3.5.2.1 Comparaison avec l'approche locale de Clauset

Nous commençons ici par une comparaison avec une autre approche locale de la détection de communautés, présentée par Clauset (2005), décrite dans la section 2.3.2, p. 45. Nous avons effectué des expériences similaires à celles présentées dans son article.

Protocole expérimental

Clauset génère des graphes en utilisant la méthodologie décrite en annexe B, section B.1.1, p. 186. Il fixe le nombre total de sommets à $n = 128$ sommets divisés en 4 communautés de taille égale, qui constituent une vérité terrain. Il choisit également de fixer un degré moyen $z = 16$ et de faire varier la proportion d'arêtes nommée z_{out}.

Nous comparons les performances de VOLCAN à celles obtenues par la modularité locale de Clauset sur un graphe entier. En effet, l'algorithme de Clauset sélectionne comme point de départ un sommet graine et construit une communauté par expansion, jusqu'à s'arrêter. Une exécution de l'algorithme ne détecte donc qu'une seule communauté.

Clauset exécute ainsi quatre fois son algorithme sur quatre graines choisies dans les quatre communautés de la vérité terrain, avant de comparer ses résultats à la vérité terrain.

Le comportement de VOLCAN étant différent, une seule instance de l'algorithme traite le graphe complet, nous avons dû agréger les résultats produits par plusieurs exécutions de l'algorithme de Clauset. Nous avons donc lancé cet algorithme quatre fois sur chaque graphe, sur quatre graines différentes choisies comme les sommets de plus haut degré de chacune des quatre communautés (tirage aléatoire si plusieurs possibilités dans la même communauté). Il est à noter que cela constitue un gros avantage donné à la méthode de

Clauset car cela facilite grandement sa détection, le choix d'une graine comme point de départ d'un algorithme local étant à la fois déterminant et problématique (voir section 2.5.3, p. 59).

Nous pré-traitons également ses résultats de la manière suivante : un sommet classé dans plusieurs communautés est gardé dans une communauté choisie aléatoirement uniformément. En effet, les communautés produites par VOLCAN sont disjointes et chaque sommet est affecté à une seule communauté. Les communautés recouvrantes sont traitées dans le chapitre 4. Chaque sommet non classé constitue sa propre communauté, de laquelle ils restera le seul membre.

L'expérience est menée pour z_{out} variant de 1 à 8, les résultats moyennés pour chaque valeur de z_{out}.

Résultats obtenus

Les résultats sont présentés sur la figure 3.4. On observe que, excepté pour $z_{out} = 3$, les résultats restent inférieurs à ceux de la méthode de Clauset. Les valeurs proches de 0 obtenues par l'ARI à partir de $z_{out} \geq 6$ signifient que la répartition en communautés proposée par VOLCAN pour ces valeurs est qualitativement proche d'une répartition aléatoire. Un graphe pour ces valeurs élevées de z_{out} n'exhibe cependant pas de structure communautaire très identifiable (visuellement ou algorithmiquement).

Cependant, bien que la NMI reste plus élevée pour la méthode de Clauset que pour VOLCAN, l'ARI montre que la partition en communautés trouvée par la méthode de Clauset n'est structurellement pas similaire à la partition originale. Ceci est dû à l'agrégation effectuée, en effet la méthode de Clauset a tendance à classer un nombre important de sommets dans plusieurs communautés, effet minimisé par le prétraitement que nous effecutons. Inversement, elle a aussi tendance à laisser beaucoup de sommets sans affectation, car elle s'arrête lorsqu'elle pense avoir trouvé la frontière d'une communauté. La façon dont nous considérons ces sommets sans affectation n'impacte que peu la NMI, mais beaucoup l'ARI qui y est plus sensible (voir les propriétés de la NMI et de l'ARI en annexe A).

Ainsi, le cœur des communautés est bien identifié, mais la précision décroît rapidement lorsque l'algorithme parvient aux frontières.

Au contraire, les performances de VOLCAN décroissent graduellement, mais l'ARI reste corrélé à la NMI, montrant la pertinence de la partition en communautés produite, comparée à l'originale. Au final, l'ARI est plus favorable à VOLCAN pour $3 \leq z_{out} \leq 5$, alors que la NMI est plus favorable à la méthode de Clauset.

On remarque également le fort écart-type pour cette méthode lorsque $2 \leq z_{out} \leq 3$, avant la chute de performances, alors que l'écart-type pour VOLCAN reste modéré. Nous interprétons ceci comme une instabilité notable de la méthode de Clauset lorsque le bruit, la réduction de la différence de densité intra/inter-communautaire par exemple quand $z_{out} \geq 2$, survient, menant à cette chute de performances.

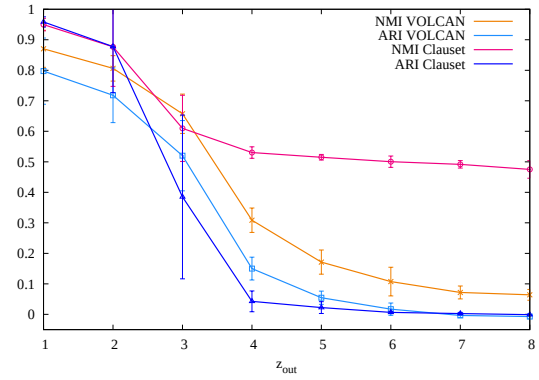


FIGURE 3.4 – ARI et NMI moyens pour toutes les communautés en fonction de la proportion d'arêtes inter-communautaires z_{out}, comparé à la méthode de Clauset.

3.5.2.2 Comparaison avec des approches globales

Dans cette expérience, nous comparons VOLCAN avec deux méthodes populaires de l'état de l'art : Louvain (Blondel et al., 2008b) et InfoMap (Rosvall & Bergstrom, 2008) (cf. section 2.4.1, p. 48). Il faut noter que les approches globales bénéficient d'un avantage certain dès le départ : affranchies des contraintes favorisant l'implémentation décentralisée imposées à VOLCAN, elles peuvent réaliser des calculs et comparaisons plus complexes et plus précis. En contrepartie, elles sont plus difficilement décentralisables.

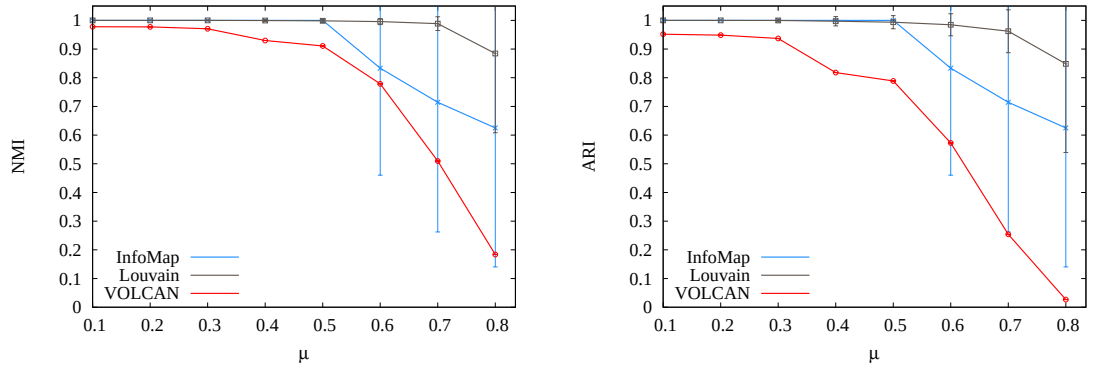
Protocole expérimental

Nous considérons un graphe de 1000 sommets créé par le générateur LFR (voir annexe B, section B.1.2, p. 187), pour des valeurs du paramètre μ entre 0.1 et 0.8. Nous mesurons la NMI et l'ARI. Les résultats, présentés figure 3.5, montrent les moyennes et écarts-types de ces mesures en fonction de μ.

Résultats obtenus

Nous constatons que la NMI est très élevée, supérieure à 0.9, jusqu'à $\mu = 0.5$. Louvain offre de meilleures performances ici que InfoMap, ce qui n'est pas toujours le cas (Hric et al., 2014), mais les deux algorithmes voient leur écart-type augmenter, particulièrement InfoMap, lorsque $\mu > 0.5$. Bien que la qualité des résultats produits par VOLCAN décroisse, son avantage est ici sa stabilité. On note en effet le très faible écart-type, qui fait que les performances moyennes de VOLCAN sont supérieures aux moins bonnes performances d'InfoMap pour $\mu \geq 6$.

Globalement, tant que la structure communautaire reste suffisamment bien définie et identifiable ($\mu \leq 0.5$), VOLCAN offre des performances raisonnables, y compris par rapport à des méthodes globales avantagées.



(a) NMI en fonction de μ

(b) ARI en fonction de μ

FIGURE 3.5 – Évaluation comparative de VOLCAN sur des graphes générés par LFR (1000 sommets)

3.5.3 Graphe réel : club de karaté de Zachary

Pour finir, nous présentons un aperçu des performances de VOLCAN sur un graphe tiré d'une situation réelle : le club de karaté de Zachary. Ce graphe et sa longue histoire sont détaillés en annexe B, section B.2.1, p. 189.

La figure 3.6 montre la partition obtenue par VOLCAN. Les valeurs de NMI et ARI obtenues sont de 0.65 et 0.67 respectivement. Trois sommets sont placés dans des communautés différentes de la répartition proposée par Zachary : #9, #14 and #20 (colorés sur la figure). Ces sommets sont à l'interface entre les deux communautés et tous connectés au sommet #34, un des sommets les plus influents dans sa communauté (on note également qu'il est de degré maximal). Les résultats de VOLCAN semblent donc cohérents.

D'autres expériences, plus complètes, ont été conduites en utilisant la version de VOLCAN étendue pour les communautés recouvrantes, LOCNeSs et sont présentées dans le chapitre 4, section 4.4, p. 99.

3.6 Bilan

Nous avons présenté dans ce chapitre un algorithme de détection de communautés sur graphe, du type orienté sommet tel que nous l'avons défini dans l'état de l'art au chapitre 2.

Cet algorithme utilise de l'information locale de façon décentralisée, dans le sens où les tâches sont effectuées en parallèle par chaque sommet sous la forme d'un Vertex Program. Il est ainsi facilement implémentable dans un contexte également décentralisé, tel qu'un réseau mobile ad-hoc opportuniste décrit dans le chapitre 1.

Nous avons examiné ses propriétés théoriques d'une part et réalisé une étude expérimentale d'autre part, démontrant l'intérêt du procédé algorithmique utilisé par VOLCAN.

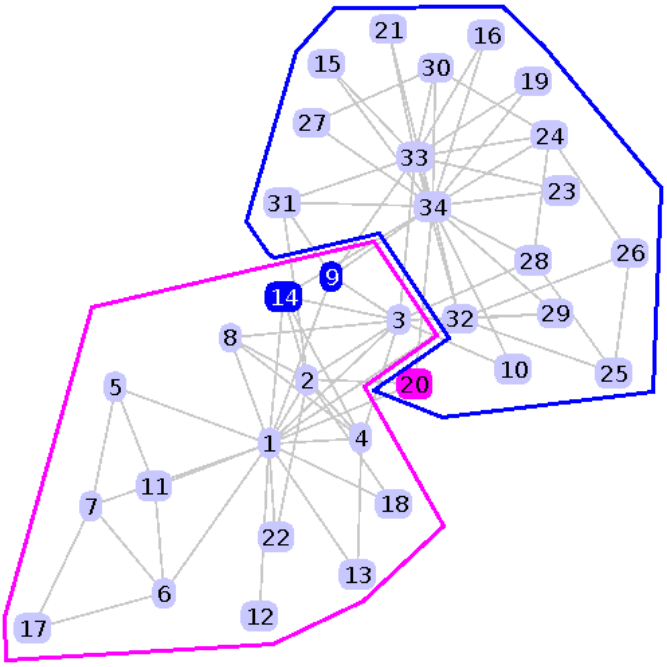


FIGURE 3.6 – Partition en communautés pour le club de karaté de Zachary. Les sommets colorés sont ceux différant de la partition proposée dans l'article de Zachary (1977).

Chapitre 4

LOCNeSs : détection de communautés recouvrantes

Sommaire

Ce chapitre s'intéresse à la problématique des communautés recouvrantes, alors que le chapitre précédent considérait les communautés disjointes. Il présente une extension de l'algorithme VOLCAN nommée LOCNeSs (*L*ocating *O*verlapping *C*ommunities in *Ne*twork *S*tructure*s*), visant à produire un découpage du graphe en sous-ensembles non-disjoints de sommets, tel qu'un sommet puisse appartenir à plusieurs communautés (multi-appartenance).

LOCNeSs applique les mêmes principes que VOLCAN, c'est-à-dire une détection orientée sommet à base de mesure de préférence locale, permettant une implémentation décentralisée.

Ce chapitre est organisé de la manière suivante : tout d'abord nous discutons de l'intérêt de détecter des communautés recouvrantes, avant de dresser un état de l'art des méthodes existantes, en particulier les méthodes orientées sommet. La section 4.3 décrit ensuite l'algorithme que nous proposons, LOCNeSs, et ses propriétés. La section 4.4 présente quatre expériences visant à valider l'approche proposée. Ces travaux ont fait l'objet de deux publications : Canu et al. (2016b) et Canu et al. (2016c).

4.1 Sommets multi-appartenants, communautés recouvrantes

Si une partition en clusters ou communautés disjointes peut être suffisamment pertinente pour l'analyse de certains graphes de terrain très segmentés, la question est moins évidente concernant les graphes encapsulant simultanément plusieurs phénomènes, plusieurs vérités.

Il est ainsi naturel d'estimer qu'un sommet puisse appartenir simultanément à plusieurs communautés, il est alors dit multi-appartenant. Ainsi, dans un réseau de co-publication, où chaque nœud correspond à un chercheur, un nœud multi-appartenant peut être interprété comme un chercheur ayant plusieurs thématiques de recherche, associées à plusieurs sous-ensembles de co-auteurs. De même, dans un réseau de communication, il est naturel d'associer les nœuds de *peering*, qui permettent l'échange de trafic, à plusieurs communautés, en l'occurrence les sous-réseaux entre lesquels ils font transiter les données. L'intersection des ensembles de nœuds de ces communautés sera donc non nulle, les sommets la composant étant dit multi-appartenants.

Cette section propose une analyse plus détaillée des communautés recouvrantes, en distinguant différents types de sommets multi-appartenants, puis en présentant deux types de critères, sémantiques et structurels, présidant à leur existence.

4.1.1 Définitions et types de sommets multi-appartenants

Derrière les termes *recouvrant* ou *multi-appartenant* se trouvent des situations correspondant à des définitions différentes. Nous présentons ici deux types de représentation des multi-appartenances et décrivons des formes empiriques de multi-appartenance. Nous

illustrons les définitions en reprenant l'exemple de la section 1.2.2, p. 14, le laboratoire de recherche.

4.1.1.1 Représentation des sommets multi-appartenants

Nous distinguons ici deux représentations possibles de sommets multi-appartenants : les fonctions d'appartenance des sommets et les intersections entre communautés.

La représentation par *fonction d'appartenance* est centrée sur les sommets : chacun dispose d'une fonction définissant son appartenance à chaque communauté. Cette appartenance peut être binaire, pour des appartenances strictes, ou graduée, pour des appartenances appelées *floues* dans la littérature (Xie et al., 2013) en référence à la théorie des sous-ensembles flous (Zadeh, 1965).

La représentation par *intersection entre communautés* se place au niveau des ensembles de sommets formant les communautés : une partie d'une communauté, sous forme de sous-graphe induit, peut être partagée avec plusieurs autres communautés. En reprenant l'exemple du laboratoire, si l'on construit des communautés par thème de recherche, certaines équipes peuvent être pluri-thématiques. A l'intérieur d'une même équipe, il peut y avoir des chercheurs mono-thématiques, mais aussi des chercheurs travaillant dans, ou à l'interface de, plusieurs thématiques différentes (qui parfois donnent naissance à des disciplines à part).

Dans un laboratoire d'informatique, on peut par exemple avoir des chercheurs travaillant en cryptographie, d'autres sur les réseaux de communication, et parmi ces deux ensembles, des chercheurs travaillant dans la cryptographie pour les réseaux de communication.

Si les fonctions d'appartenance peuvent modéliser toutes les situations de recouvrement, la taille de l'information nécessaire, n vecteurs de $n-1$ composantes dans le cas binaire par exemple, peut constituer un frein pour l'exploitation de grands graphes. Les intersections entre communautés, plus économes dans le cas où de larges parties de communautés sont partagées, sont cependant moins adaptées dans le cas de nombreuses multi-appartenances isolées (voir ci-dessous), car on considère alors une intersection (sous-graphe induit) par sommet multi-appartenant isolé.

4.1.1.2 Formes de multi-appartenance

Nous décrivons ici deux formes de multi-appartenances fréquemment rencontrées dans les graphes de terrain, inclusion hiérarchique, intersection substantielle et multi-appartenances isolées, que nous illustrons avec l'exemple du laboratoire de recherche.

L'*inclusion hiérarchique* est un cas particulier d'intersection où la multi-appartenance peut être interprétée comme une analyse multi-échelle, qui donne des informations plus ou moins détaillées sur la structure du réseau. Dans l'exemple du laboratoire, les équipes

de recherche peuvent être regroupées en départements (surtout si le laboratoire est gros), par exemple au LIP6 les chercheurs de l'équipe LFI [16] font également partie du département DAPA [17]. Par transition, ils sont donc membres des communautés LFI et DAPA, la première étant entièrement incluse dans la seconde.

Dans le cas d'un découpage selon les équipes du laboratoire, un chercheur peut appartenir à plusieurs équipes différentes, possiblement dans plusieurs laboratoires différents. C'est par exemple le cas des chercheurs associés ou invités. Ce cas est généralement ponctuel, mis à part les équipes partagées ou en collaboration qui seraient plutôt à classer dans la catégorie des intersections substantielles, nous le nommons *multi-appartenance isolée*.

Quelle forme pour LOCNeSs ?

En considération du contexte exposé section 1.4, p. 30, il paraît plus pertinent de détecter des multi-appartenances isolées afin de mettre en évidence des sommets particuliers, un point d'échange permettant de favoriser le routage, par exemple à l'interface entre deux communautés. Les propriétés que nous attendons de LOCNeSs sont donc d'identifier correctement des sommets susceptibles d'endosser un rôle particulier. En effet la constitution de leaders par similarité locale, par exemple, rendrait possiblement moins efficace la détection de zones substantielles d'intersection entre communautés, étant moins à même de détecter une différence nette de densité au sein de cette zone et considérant plutôt les sommets comme appartenant à une seule et même communauté.

4.1.2 Caractérisation des multi-appartenances

L'observation générale de communautés recouvrantes dans les graphes de terrain a donné lieu à nombre de travaux de recherche. Palla et al. (2005), par exemple, ont caractérisé la présence de recouvrement dans des réseaux divers tels que collaborations scientifiques, association de mots, et interactions protéine-protéine (Xie et al., 2013).

Il faut noter que si la multi-appartenance est intuitive et peut être illustrée par de nombreux exemples, sa définition souffre des mêmes indéterminations que la structure communautaire elle-même : il faut choisir le type de multi-appartenance que l'on souhaite détecter ou le définir, et conjointement le caractériser par rapport aux graphes considérés.

Nous proposons de retenir deux types de critères à cette fin, les critères *structurels* et *sémantiques*, successivement discutés ci-dessous.

Critères sémantiques : fonctions spécifiques de sommets

Ce premier type de critères a trait à l'objet même, aux fonctionnalités du réseau. En effet, il est très fréquent dans un réseau que certains nœuds endossent un rôle spécifique,

16. LFI : Learning, Fuzzy and Intelligent systems
17. DAPA : Données et APprentissage Artificiel

par exemple de relai ou de médiateur entre communautés pour des réseaux sociaux ou une interface d'échange de trafic dans les réseaux de communication.

Ainsi, dans l'exemple du laboratoire de recherche, si le but global de tous les membres du laboratoire est d'étudier des phénomènes et produire des connaissances, certains membres remplissent des fonctions spécifiques, au niveau de l'ensemble du laboratoire (directeur/directrice ou responsable d'équipe) ou à l'interface entre plusieurs équipes ou services (ingénieurs, techniciens, administratifs).

Lorsque ces rôles influent sur la structure du réseau, par exemple au niveau de la densité des arêtes incidentes à ces nœuds spéciaux, il est possible de les retrouver à l'aide d'un algorithme de détection traitant les multi-appartenances.

Critères structurels : cohésion globale du réseau

Si une partie du travail d'identification des sommets multi-appartenants est de trouver une correspondance structurelle à des critères sémantiques, il est également nécessaire de conserver une cohérence structurelle qui pourrait être mise à mal par l'utilisation de critères basés sur la sémantique pure.

Reid et al. (2013) ont par exemple observé, dans plusieurs types de graphes de terrain dont les réseaux sociaux, que de nombreuses cliques se situent entre plusieurs communautés naturelles (au sens de : annotées par des experts) distinctes, ce qui veut dire qu'une partie des arêtes de ces cliques sont des arêtes inter-communautaires. La plupart des méthodes de détection disjointes « brisent » ces cliques et perdent ainsi une précieuse information sur la structure naturelle du graphe de terrain.

Enfin, un point crucial à considérer dont le traitement met en défaut certains algorithmes de détection (Xie et al., 2013) est la proportion de sommets multi-appartenants dans un graphe : il ne paraît pas cohérent d'en détecter un grand nombre. En effet, l'un des intérêts de trouver la structure communautaire est qu'elle propose un découpage du graphe faisant ressortir les zones denses, qui portent une information en tant que telles. Si les communautés se chevauchent à un point où, par exemple, 50% des sommets de chaque communauté sont partagés avec d'autres communautés, la structure apparaît brouillée et perd en information [18].

Il existe d'autres modèles pour représenter une telle structure, on peut par exemple annoter chaque sommet pour faire figurer les proximités ou appartenances multiples à des sommets ou communautés. Il suffira alors de réaliser la détection en tenant compte des attributs (Bothorel et al., 2015).

Nous concluons ici qu'il est important, lors de la conception d'une méthode de détection, d'obtenir une adéquation entre les propriétés sémantiques des sommets à isoler et la

18. Remarque : nous considérons ici un cas classique où des communautés sont représentées comme un ensemble de sommets ou de sous-graphes, le cas des communautés hiérarchiques présente bien évidemment des ensembles inclus les uns dans les autres.

structure communautaire globale produite : la multi-appartenance doit être une information supplémentaire décrivant certains sommets et améliorant la compréhension du graphe, tout en préservant la cohérence de la structure communautaire.

4.2 Etat de l'art

De même que la détection de communautés disjointes dans les graphes est liée à la tâche de clustering (cf. section 2.2.3, p. 43), la détection de communautés recouvrantes est liée au partitionnement de données en sous-ensembles non-disjoints. Celui-ci prend souvent la forme de clustering pondéré, avec différents cadres formels de représentation des poids parmi lesquels on peut citer les cadres probabiliste (Basu et al., 2004), flou (Bezdek, 1981, Höppner et al., 1999) et évidentiel (Masson & Denoeux, 2008, Antoine et al., 2014, Antoine & Labroche, 2015, Zhou et al., 2015) pour en mentionner quelques uns. D'autres approches considèrent une affectation totale des points à plusieurs clusters (Ben N'Cir et al., 2015). On peut notamment citer l'extension de k-means au cas des clusters recouvrants (Cleuziou, 2008), également adapté au clustering de graphes (Bauman & Bauman, 2017). C'est l'approche majoritaire en détection de communautés et celle que nous mettons en œuvre dans nos propositions.

Nous évoquons ici tout d'abord les méthodes étendant ou complétant des approches existant pour le cas disjoint, telles que décrites dans le chapitre 2, puis des approches originales conçues uniquement pour traiter les communautés recouvrantes. Enfin, nous discutons des méthodes orientées-sommet. Xie et al. (2013) proposent un état de l'art plus complet de ces méthodes, celui de Fortunato (2009) traite également des communautés recouvrantes.

4.2.1 Méthodes adaptées d'approches pour communautés disjointes

Les approches décrites dans le chapitre 2, section 2.4, p. 48 ont été étendues à la détection de communautés recouvrantes. Nous présentons ici ces extensions pour deux des familles adaptées : l'optimisation d'un critère de qualité numérique et les méthodes par propagation de labels.

Optimisation

Les approches par optimisation ont constitué les premières sources des travaux sur la détection de communautés recouvrantes. A l'instar de leur contrepartie pour la détection de communautés disjointes, elles requièrent la définition de critères de qualité, cette fois adaptés au cas recouvrant. La modularité a ainsi adaptée pour évaluer des partitions non disjointes (Shen et al., 2009).

Plus spécifique, la méthode OSLOM (Lancichinetti et al., 2011) se détache pour ses résultats remarquablement précis, constatés expérimentalement (Xie et al., 2013). Le critère de qualité est une fonction de *fitness* optimisée localement, qui évalue la significativité statistique des communautés, selon le même modèle que la modularité (c'est-à-dire la probabilité comparée à un modèle de référence), mais en utilisant des propriétés du graphe différentes. L'approche locale utilisée permet d'éviter certains inconvénients tels que la limite de résolution (cf. section 2.4.1, p. 48). Cependant, le temps d'exécution de cette méthode est substantiellement plus élevé que celui de la plupart de ses concurrentes.

Propagation de labels

Plusieurs méthodes utilisant la propagation de labels ont été proposées. COPRA (Gregory, 2010) est principalement une adaptation de la méthode de Raghavan et al. (2007) (décrite dans la section 2.4.3, p. 52), elle l'étend en permettant à chaque sommet de conserver plusieurs labels à l'aide de coefficients d'appartenance dépendant du nombre de fois où ces labels ont été reçus. Elle conserve les labels dont le coefficient est supérieur à un seuil donné et les sommets appartiennent aux communautés de tous les labels qu'ils ont conservés une fois les phases de propagation terminées. Cette technique est reprise et améliorée par SLPA (Xie et al., 2011) qui introduit deux rôles différents : *speaker* et *listener*. Chaque sommet endosse l'un de ces rôles et deux phases de propagation, l'une des speakers vers les listeners et l'autre réciproquement, sont alternées. Ce procédé permet d'améliorer la qualité des résultats et le temps de convergence par rapport à COPRA.

Plus récemment, DLPA (He-Li et al., 2015) proposent une nouvelle amélioration par une approche hybride entre la propagation de labels et la coopération entre sommets telle qu'utilisée par exemple dans les systèmes multi-agents (voir page 95) : chaque sommet évalue la confiance qu'il porte en ses voisins selon un critère proposé dans l'article et sélectionne son ou ses labels via un procédé basé sur des « labels dominants » : le label d'origine de chaque sommet est son label dominant, à chaque nouvelle propagation le label dominant devient celui maximisant un coefficient d'appartenance, pondéré par la confiance. Une fois la convergence achevée la communauté principale d'appartenance est celle du label dominant et les communautés supplémentaires sont celles des labels conservés par les sommets, de façon similaire à COPRA. Les labels dominants et la confiance permettent de restreindre le nombre de labels en circulation pendant la propagation et ainsi de limiter d'une part la masse de propagation et, d'autre part, la prise en compte des multi-appartenances moins pertinentes.

4.2.2 Approches dédiées

Une autre famille de méthodes de détection de communautés recouvrantes repose sur des approches originales, dédiées à ce problème. Nous distinguons ci-dessous deux catégories

parmi celles existantes (Xie et al., 2013) : les méthodes qui exploitent la notion de cliques et celles qui se placent dans le cadre multi-agents.

Cliques

De nombreuses approches de détection de communautés recouvrantes sont basées sur l'utilisation de cliques. En trouvant et fusionnant les cliques d'un graphe, il est possible de former des communautés, un sommet compris dans plusieurs cliques situées dans des communautés différentes est alors multi-appartenant.

Différentes approches basées sur des cliques ont été proposées, par exemple en utilisant la *percolation* (Palla et al., 2005, Derényi et al., 2005, Palla et al., 2007). Le principe est de former des communautés en *percolant* (adaptation du sens utilisé en physique) des communautés dites *adjacentes*. Une communauté est définie comme une union maximale de k-cliques adjacentes les unes aux autres, c'est-à-dire de sous-graphes complets de $k \in \mathbb{N}$ sommets partageant $k-1$ sommets. Ainsi, on peut former une chaîne entre les paires de sommets de la communauté, en passant par des cliques adjacentes, justifiant le terme percolation.

Il faut noter que cette approche nécessite d'abord de fixer le paramètre k, qui définit la taille des cliques. Le nombre de cliques dans un graphe de terrain étant généralement élevé, les méthodes par percolation proposent de rechercher d'abord la clique maximale, dont la taille détermine k. Cette approche est néanmoins coûteuse : la recherche d'une clique maximale dans un graphe est NP-difficile et celle de toutes les cliques d'un graphe est NP-complète. D'autre part, la contrainte imposée à la structure des communautés est forte : comme elles sont formées d'unions de cliques, leur densité est très proche de 1. Or il est généralement admis dans le champ de la détection de communautés que ces dernières se rapprochent de quasi-cliques dans lesquelles chaque sommet de l'ensemble n'est pas lié à tous les autres, mais à une grande majorité d'entre eux (Fortunato, 2009). La densité reste proche de 1 mais peut se révéler plus basse que celle de cliques percolées.

L'algorithme GCE (Lee et al., 2010) est à la croisée de la détection de cliques, de l'approche orientée sommet et de l'agrégation par optimisation : il détecte d'abord les cliques d'un graphe puis les utilise comme graines pour baser une expansion. Celle-ci repose sur l'optimisation d'une fonction objectif afin de transformer les cliques en communautés en étendant leurs voisinages.

Enfin, Evans (2010) propose de construire des projections du graphe en un meta-graphe appelé *clique-graph* dont chaque sommet est une clique du graphe source, puis d'appliquer un algorithme de partitionnement afin de produire la structure communautaire (en l'occurrence l'algorithme de Louvain).

Simulations multi-agents

Un autre type d'approches dédiées aux communautés recouvrantes utilise des simulations multi-agents pour donner former les communautés. Ces méthodes effectuent généralement une détection à base de processus locaux, mais utilisant la totalité de l'information du graphe. Elles sont donc plutôt centralisées, bien qu'elles puissent être implémentées de façon décentralisée relativement facilement.

La méthode de Chen et al. (2010), par exemple, repose sur un jeu appelé *community game*. Chaque sommet constitue un agent et choisit une ou plusieurs communautés parmi k possibles à l'aide d'une fonction d'utilité. La structure communautaire résulte d'un équilibre de Nash local entre les choix des agents. L'étude théorique (Chen et al., 2010) garantit une solution et des évaluations expérimentales sur des benchmarks (Xie et al., 2013) ont attesté que ces solutions sont généralement de bonne qualité. Néanmoins, cette méthode présente plusieurs désavantages majeurs dont un coût computationnel élevé, dû au nombre de communautés k que les agents doivent considérer, les auteurs établissant que k est polynomial de n (taille du graphe).

La méthode iLCD (Cazabet & Amblard, 2011) s'attache majoritairement à détecter des communautés sur des graphes *dynamiques* (voir chapitre 6), mais elle peut également être appliquée à des graphes statiques. Le processus de simulation considère chaque sommet comme un agent, qui peut procéder à 5 types d'actions : essayer de créer une nouvelle communauté, créer un nouveau lien avec un autre agent (arête du graphe), supprimer un lien avec un autre agent, demander à rejoindre une communauté et demander à expulser un agent d'une communauté. Des métriques comme la représentativité (*representativeness*) et l'isolement (*seclusion*), dont les définitions sont données dans l'article, sont utilisées afin de déterminer l'issue des actions des agents, caractérisant ainsi les communautés. Ainsi, les communautés ne sont pas des ensembles prédéfinis que les agents souhaitent rejoindre, mais des ensembles en constante évolution façonnés par les agents eux-mêmes.

iCLD a l'avantage d'être plutôt rapide sur un graphe statique, mais a pour inconvénient une qualité moyenne des résultats lors d'évaluations expérimentales sur benchmark (Xie et al., 2013).

4.2.3 Approches orientées-sommet

Les récentes approches orientées sommets ont largement été adaptées, voire conçues dès le départ, afin de traiter des communautés recouvrantes. Nous en évoquons ici quelques unes, tout d'abord représentatives des méthodes basées sur les marches aléatoires autour de graines, puis deux méthodes basées sur des analyses locales de voisinage.

Whang et al. (2013) étendent les travaux de Andersen et Lang (2006) et Gleich et Seshadhri (2012) sur la détection par marche aléatoire utilisant le PageRank Personnalisé (cf. section 2.5, p. 53). Les graines sont sélectionnées dans le graphe à partir d'ensembles

de sommets de faible conductance selon deux stratégies adaptées des méthodes citées. Les communautés sont détectées localement, un sommet placé dans plusieurs communautés locales est multi-appartenant. L'étude théorique proposée (Whang et al., 2013) fait état de bons résultats, notamment concernant le temps de calcul et la structure communautaire globale, mais ne traite pas de la pertinence des sommets multi-appartenants détectés. Moradi et al. (2014) calculent un score de similarité pour chaque sommet, basé sur le voisinage de ce sommet, et comparent les scores entre voisins afin d'identifier les graines. Les communautés sont ensuite détectées localement en utilisant une technique à base de marche aléatoire et de PageRank personnalisé similaire à la méthode précédente.

LICOD (Yakoubi & Kanawati, 2014) est un algorithme basé sur des leaders, proche de Top-Leaders (Rabbany et al., 2010) (voir également section 2.5.3, p. 59), dont la première phase consiste à sélectionner ceux-ci à l'aide d'une métrique, comme une centralité. Les leaders susceptibles d'appartenir à une même communauté (voisins par exemple) sont directement regroupés. Dans une seconde phase les sommets restants, les followers, choisissent un leader, puis calculent leur degré d'appartenance à sa communauté (formée du leader et des autres followers l'ayant choisi). Dans une dernière phase, l'algorithme compare ce degré avec celui de ses voisins, et peut choisir de changer de communauté s'l semble trop isolé par rapport à eux. La structure finale est celle obtenue après stabilisation, lorsque plus aucun sommet ne souhaite changer de communauté.

NECTAR (Cohen et al., 2016) évalue localement l'appartenance de chaque sommet à une ou plusieurs communautés, en optimisant une métrique choisie selon la densité de triangles dans le graphe : une adaptation de WCC (Prat-Pérez et al., 2012) pour les communautés recouvrantes ou une adaptation de la modularité. L'optimisation est réalisée en enlevant le sommet de sa communauté et en calculant le gain d'optimisation s'il était placé dans la communauté d'un de ses voisins, cette phase étant répétée jusqu'à un seuil prédéfini, usuellement n (nombre de sommets du graphe).

4.3 Algorithme LOCNeSs

Nous présentons dans cette section l'algorithme LOCNeSs, adaptation de VOLCAN à la détection de communautés recouvrantes. Le but de LOCNeSs est d'appliquer les procédés introduits avec VOLCAN à la détection de communautés recouvrantes en conservant leurs avantages parmi lesquels : une possibilité d'implémentation décentralisée et une limitation de propagation (voir sections 3.3, p. 74 et 3.4, p. 76).

Nous décrivons tout d'abord des principes permettant d'adapter VOLCAN aux communautés recouvrantes, puis les modifications que nous proposons de lui appliquer afin de les détecter. Nous discutons enfin certaines des propriétés de LOCNeSs.

4.3.1 Principes

LOCNeSs reprend les mêmes principes et étapes que VOLCAN : c'est une méthode orientée-sommet découpée en deux phases, la première opère une sélection de sommets leaders constituant le cœur des communautés et la seconde effectue constitue les communautés par fusion en respectant les rôles leader/follower.

L'unicité de l'affectation effectuée par VOLCAN, qui détermine une partition disjointe, est due à l'unicité du leader choisi (cf. section 3.2.3, p. 72). La différence essentielle de LOCNeSs vient de ce qu'il autorise la sélection de *plusieurs* leaders. En conséquence, l'étape de fusion est également modifiée. Plus précisément, elle est décomposée en 2 parties, consistant en une fusion puis une affectation à des communautés additionnelles : la multi-appartenance est vue comme un enrichissement a posteriori de la décomposition du graphe en sous-graphes.

Il faut noter que ces principes généraux sont indépendants du choix des candidats parmi lesquels les leaders choisis sont sélectionnés : la description est basée sur la méthode d'identification effectuée par VOLCAN, mais les candidats peuvent provenir de toute méthode orientée sommet. L'algorithme LOCNeSs peut être vu comme une méthode générique, pouvant être appliquée en post-traitement de nombreux algorithmes existants. Cette adaptation est néanmoins plus aisée pour les méthodes de type leader-follower.

4.3.2 Description de l'algorithme

L'algorithme LOCNeSs a donc la même structure que VOLCAN, en deux étapes. L'algorithme est décrit dans le pseudo-code 5, page 98.

Etape 1 : choix *des* leader*s*

Cette étape présente une différence essentielle d'avec VOLCAN : au lieu de sélectionner un unique leader a_v, chaque sommet sélectionne un ensemble de leaders A_v. Alors que VOLCAN choisit aléatoirement un leader parmi tous les sommets avec qui l'agreement est maximisé (voir équation 3.6, p. 72, LOCNeSs conserve tous ces sommets dans l'ensemble A_v.

Cette étape peut être réalisée à l'aide de n'importe quelle autre méthode de détection de type leader-follower.

Etape 2 : fusion et répartition

Cette étape est elle-même divisée en deux :

— 2.1 – affectation à une communauté principale et fusion : un leader principal $\hat{a}_v \in A_v$ unique est choisi par chaque sommet v, maximisant le degré, choisi aléatoirement si plusieurs sommets de A_v sont de degré maximal.

Algorithme 5 LOCNeSs - Vertex Program

Require: $G = (V, E)$ un graphe,
$\{A_v : v \in V\}$ ensemble des leaders pour chaque sommet v
Ensure: $C \subset \mathcal{P}(V)$ ensemble de communautés, partition non disjointe des sommets de G

```
Initialisation : C ← {{v} : v ∈ V}
Etape 2.1
for each v ∈ V do
    â_v ← arg max_{a∈A_v} d_a (unique, tiré aléatoirement si plusieurs possibilités)
    fusion(C(v), C(â_v)) (cf. description, étape 2.1)
end for

Etape 2.2
for each v ∈ V do
    if |A_v| > 1 then
        for all a_v ∈ A_v \ {â_v} do
            if v ∉ C(a_v) then
                C(a_v) ← C(a_v) ∪ {v}
            end if
        end for
    end if
end for
```

Ce compromis est effectué afin de conserver une bonne structure communautaire lors de l'étape de fusion. En effet, la fusion de communautés de followers avec celle(s) de(s) leader(s) devient non-triviale lorsqu'un follower dispose de plusieurs leaders. Nous avons constaté lors de travaux préparatoires que fusionner la communauté d'un follower avec celles de tous ses leaders est une forme de regroupement beaucoup trop fort : la structure communautaire n'est alors souvent plus constituée que d'un nombre très réduit de communautés, parfois une seule, sans lien avec la structure obtenue par une détection disjointe.

Une fois ces leaders principaux identifiés, une première phase de fusion identique à celle de VOLCAN est opérée, entre les communauté des sommets et de leur leader principal. A l'issue de cette phase, chaque sommet dispose d'une communauté principale.

— 2.2 – affectation des communautés additionnelles : une fois l'étape 2.1 terminée, chaque v identifié comme multi-appartenant, c'est-à-dire tel que $|A_v| > 1$, se voit ajouté à toutes les communautés $C(u), u \in A_v \setminus \{\hat{a}_v\}$, soit les communautés de ses leaders excepté celle de son leader principal dans laquelle il se trouve déjà. Cette sous-étape est celle qui apporte la multi-appartenance.

4.3.3 Propriétés

Globalement, LOCNeSs garde les mêmes propriétés que VOLCAN en termes de limitation de propagation et stabilité, comme montré dans les expériences.

Complexité

La complexité est très peu affectée : seule la nouvelle étape 2.2 ajoute des calculs, lors de l'addition de nouveaux sommets aux communautés existantes. Il y a autant d'ajouts que de sommets multi-appartenants, c'est donc une addition de $\mathcal{O}(n)$ opérations, qui est dominée par la complexité globale de l'algorithme (voir complexité de VOLCAN, section 3.4.2, p. 78).

Propagation

La quantité de propagation change peu pour LOCNeSs, en effet il n'y a pas d'étape supplémentaire génératrice de propagation : l'information générée peut être incluse dans les messages déjà échangés et recensés dans la description de la propagation pour VOLCAN (cf. section 3.4.4, p. 79), influençant uniquement la taille des messages générés.

Stabilité

La conception de LOCNeSs a été étudiée pour conserver la stabilité des communautés produites, comme nous l'avons détaillé dans la description de l'algorithme section 4.3.2, p. 97. Nous montrons que c'est bien le cas avec les résultats expérimentaux de la section 4.4 ci-dessous.

4.4 Etude expérimentale

Nous avons réalisé plusieurs expériences afin d'étudier, sur des graphes artificiels (benchmarks) ainsi qu'un graphe de terrain, le comportement et la validité des résultats de LOCNeSs, comparé à d'autres algorithmes, et en particulier sa capacité à identifier correctement et pertinemment des sommets multi-appartenants malgré son cadre de conception très contraint (décentralisation).

4.4.1 Protocole expérimental

Nous comparons les résultats de LOCNeSs à COPRA, GCE, iLCD, OSLOM et SLPA (cf. section 4.2, p. 92). Les valeurs des paramètres requis par ces algorithmes sont fixées d'après les recommandations de Xie et al. (2013).

Les trois premières expériences utilisent des graphes artificiels générés par le benchmark LFR (Lancichinetti et al., 2008), décrit en annexe B, p. 185, 10 instances sont créées pour

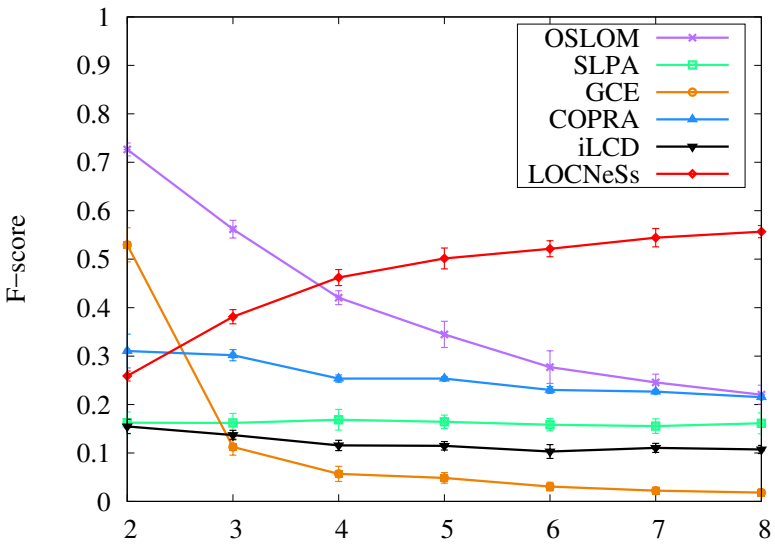


FIGURE 4.1 – Identification des sommets multi-appartenants : F-score en fonction de O_m

chaque jeu de paramètres. Les différents jeux de paramètres utilisés (nombre de sommets n, paramètres liés aux sommets recouvrants O_n et O_m, paramètre de mixage μ etc.) sont ceux proposés par Xie et al. (2013). Les paramètres soumis à variation sont détaillés dans les sous-sections relatives aux expériences. Deux intervalles de tailles de communautés sont utilisés : s (small) pour des communautés comprenant 10 à 50 sommets et b (big) pour des communautés comprenant 20 à 100 sommets.

Chaque algorithme est exécuté 10 fois sur chaque instance et les résultats donnés sont les moyennes et écart-types sur ces 100 lancements. Les valeurs des paramètres du benchmark sont celles utilisées par .

Nous utilisons également les critères classiques recommandés par Xie et al. (2013) pour comparer les résultats à la vérité terrain : F-score pour l'identification des sommets multi-appartenants, NMI (variante adaptée aux communautés recouvrantes) et Omega Index pour la comparaison des partitions de communautés. Les deux premiers sont compris entre $[0, 1]$ et l'Omega Index entre $]-1, 1]$. Tous sont à maximiser. Ces critères sont également détaillés en annexe A, p. 177.

4.4.2 Qualité de l'identification des sommets multi-appartenants

Cette expérience utilise le F-score pour montrer la capacité de LOCNeSs à identifier les sommets multi-appartenants, comparés aux autres méthodes considérées. Un sommet est considéré comme classé correctement si LOCNeSs lui affecte plusieurs communautés et qu'il en possède effectivement plusieurs dans la vérité terrain. Le nombre de communautés n'est pas pris en compte : dès qu'un sommet appartient à plus de 2 communautés, dans les résultats de LOCNeSs comme dans la vérité terrain, il est défini comme multi-appartenant.

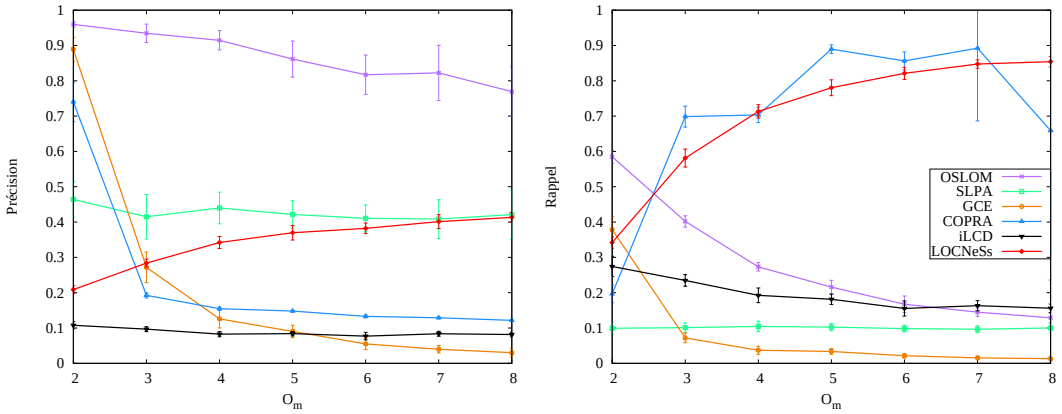


FIGURE 4.2 – Identification des sommets multi-appartenants : rappel et précision en fonction de O_m

Les paramètres de benchmark utilisés pour générer les graphes sont $n = 5000, \mu = 0.3, O_n = 10\%$, taille s. O_m varie.

La figure 4.1 montre la valeur du F-score en fonction du nombre de multi-appartenance dans la vérité terrain O_m, la figure 4.2 distingue la précision et le rappel constituant ce F-score. On constate que le F-score de LOCNeSs augmente lorsque O_m croît, contrairement à toutes les autres méthodes pour lesquelles il décroît ou reste stable. On peut l'expliquer par les valeurs de rappel atteintes par LOCNeSs : 34% pour $O_m = 2$ à 85% pour $O_m = 8$, alors que SLPA, par exemple, stagne aux alentours de 10% pour les mêmes valeurs de O_m. La précision est moins bonne : 20% pour $O_m = 2$ à 41% pour $O_m = 8$, alors que SLPA fluctue entre 40 et 50%, mais se classe troisième plus haute.

Ainsi, LOCNeSs atteint un bon compromis : il identifie plus de sommets multi-appartenants que les autres méthodes évaluées et a fortiori plus qu'il n'en existe réellement dans la vérité terrain, mais pas un trop grand nombre non plus. Cela lui permet de trouver significativement plus de sommets effectivement étiquetés comme multi-appartenants que ces autres méthodes. On note également que le F-score de LOCNeSs surpasse les autres pour $O_m \geq 4$ en restant croissant, signe que la méthode n'est pas perturbée par la multi-appartenance à un grand nombre de communautés différentes à la fois : en effet LOCNeSs atteint un F-score de plus de 55% lorsque $O_m = 8$ alors que toutes les autres méthodes testées atteignent un F-score inférieur à 25%.

4.4.3 Qualité des partitions

Le but de cette expérience est de mesurer la sensibilité de LOCNeSs aux paramètres n (fig. 4.3) et O_m (fig. 4.4). Les paramètres utilisés pour générer les graphes artificiels sont $\mu = 0.3, O_n = 10\%$, tailles de communautés s, ainsi que $O_m = 2$ (fig. 4.3) et $n = 5000$ (fig. 4.4).

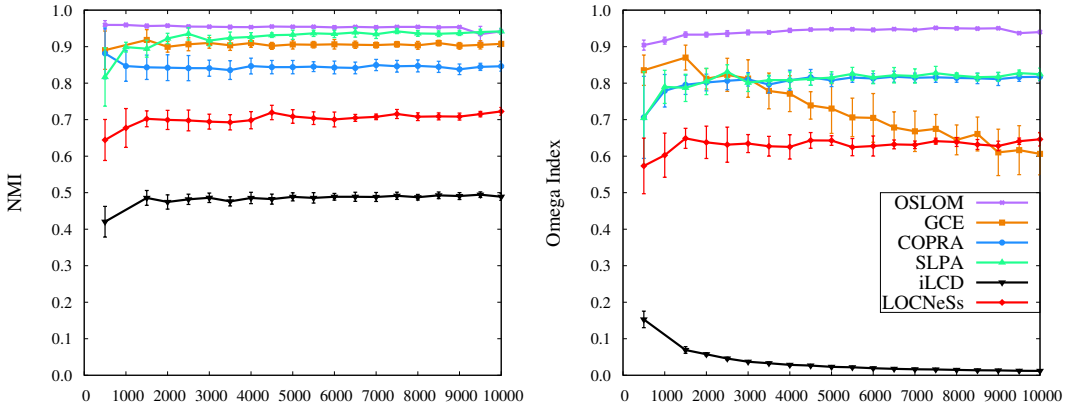


FIGURE 4.3 – Comparaison de la stabilité de détection en terme de NMI et Omega Index, en fonction du nombre de sommets du graphe, n

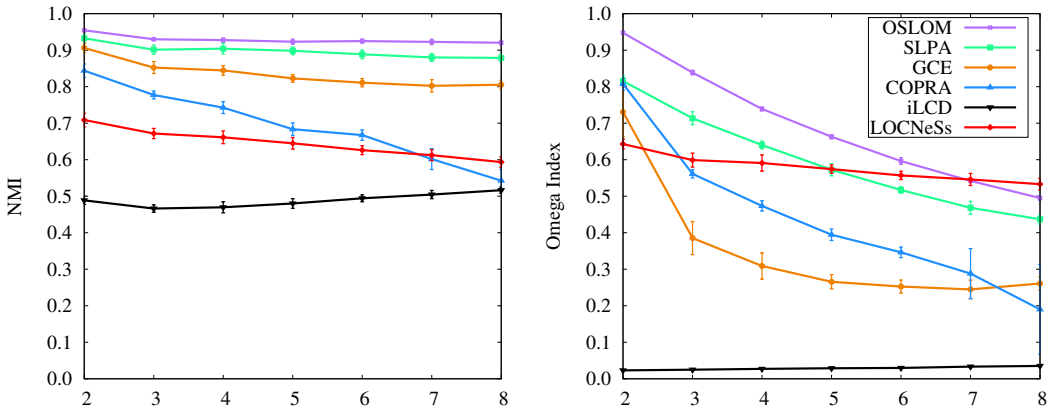


FIGURE 4.4 – Comparaison de la qualité des partitions en terme de NMI et Omega Index, en fonction du niveau de multi-appartenance, O_m

La figure 4.3 montre la moyenne et l'écart-type de la NMI et l'Omega Index entre les résultats de LOCNeSs et la vérité terrain, pour $O_m = 2$ et des valeurs de n variant de 1000 à 10 000 sommets. On observe que, bien qu'offrant une NMI et un Omega Index plus bas que les méthodes à optimisation et propagation, LOCNeSs reste relativement stable dans tous les cas. En effet on ne note pas de différence de plus de 0.1 sur les critères, et les écarts-types restent raisonnables. LOCNeSs reste également toujours meilleure que iLCD.

La figure 4.4 montre la moyenne et l'écart-type de NMI et Omega Index pour un graphe de $n = 5000$ sommets, en faisant varier la valeur de O_m entre 2 et 8. On constate que l'Omega Index de LOCNeSs baisse plus rapidement que pour les autres méthodes, signe que l'algorithme parvient bien à identifier les sommets multi-appartenants pertinents (cf. expérience précédente) sans trop dégrader la structure communautaire globale.

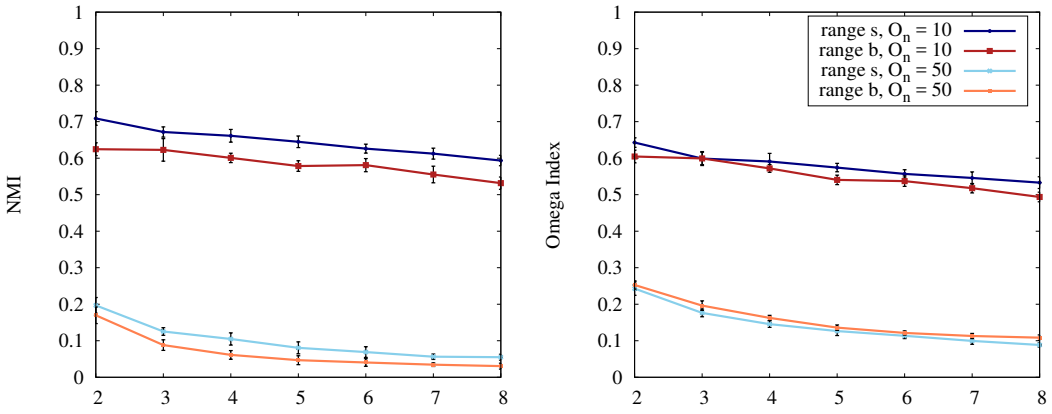


FIGURE 4.5 – Impact de O_n et des tailles s, b en fonction de O_m pour LOCNeSs seulement

4.4.4 Proportion de sommets recouvrants, tailles des communautés

Nous mesurons, avec cette expérience, l'impact de deux paramètres principaux : la proportion de sommets multi-appartenants O_n et les tailles de communautés s et b, en NMI en fonction de O_m. Les résultats (fig. 4.5) sont présentés uniquement pour LOCNeSs, des expériences similaires pouvant être trouvées dans Xie et al. (2013) pour les autres algorithmes.

Les paramètres utilisés pour générer les graphes sont $n = 5000, \mu = 0.3$. Deux valeurs de O_n sont testées : $O_n = 10\%, O_n = 50\%$, cette dernière valeur n'étant pas réaliste (cf. section 4.1.1, p. 88, critères structurels).

Comme l'on peut s'y attendre, les NMI pour $O_n = 50\%$ chutent fortement, les tailles s et b restant néanmoins corrélées. En effet, la moitié des sommets de chaque communauté appartient également à une autre, ce qui efface en grande partie les limites de ces communautés et complexifie donc grandement leur détection. Les petites communautés sont légèrement mieux identifiées que les grosses, un résultat déjà noté par Xie et al. (2013).

4.4.5 Dégénérescence par permutations

Nous présentons ici une expérience mesurant la dégradation de l'Omega Index et la NMI en fonction de permutations réalisées dans la répartition en communautés proposée par LOCNeSs.

Cette expérience vise également à mesurer la stabilité de la structure communautaire produite, elle consiste à permuter deux sommets, les échangeant possiblement de communauté. Plus la valeur des mesures décroît rapidement en fonction du nombre de permutations, moins la structure communautaire est stable. Une décroissance linéaire, ou plus lente, de 0 et n (taille du graphe) permutations indique une bonne stabilité des communautés produites. Les résultats sont montrés figure 4.6 : l'axe des abscisses donnent le nombre de

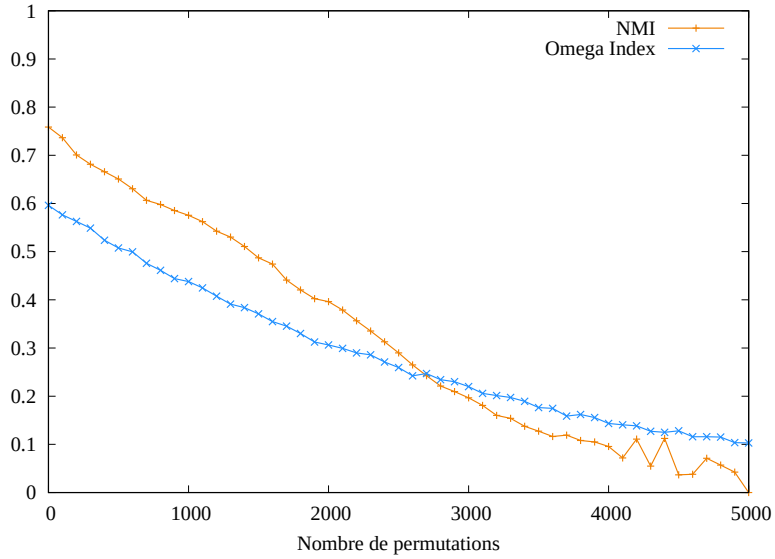


FIGURE 4.6 – Dégradation des performances par bruitage des résultats, en fonction du nombre de permutations

permutations effectuées sur le résultat produit par LOCNeSs et l'abscisse les valeurs des critères Omega Index et NMI.

Ces critères sont calculés en comparant la structure communautaire après permutations à la vérité terrain, sur un graphe de paramètres $n = 5000, \mu = 0.3$, taille des communautés s.

On remarque que l'Omega Index et la NMI décroissent de manière quasi-linéaire, sans montrer de chute brusque. On note que la NMI décroît plus vite, selon nous à cause de l'effet lié à l'entropie, décrit dans l'annexe A donnant plus d'importance aux paires de sommets mises dans les mêmes communautés indépendamment de la structure globale.

Ces résultats laissent à penser, à l'instar des expériences précédentes, que la structure des communautés trouvées par LOCNeSs est relativement stable, comparativement à la vérité terrain, elle résiste bien au bruit apporté par les permutations.

4.4.6 Réseau réel : High School Network

Cette expérience illustre visuellement le découpage produit par LOCNeSs sur le graphe d'un ensemble d'élèves d'un collège (junior high school), proposé par Xie et al. (2013). Ce graphe est issu d'une partie d'une collecte de données à but socio-sanitaire réalisée aux Etats-Unis, détaillée dans l'annexe B. Le graphe tiré de cette collecte associe un sommet à chaque lycéen et une arête à chacune de ses relations d'amitié avec d'autres lycéens telle que constatée par l'étude. Il comporte 69 sommets pour 220 arêtes, son degré moyen est de 6.4.

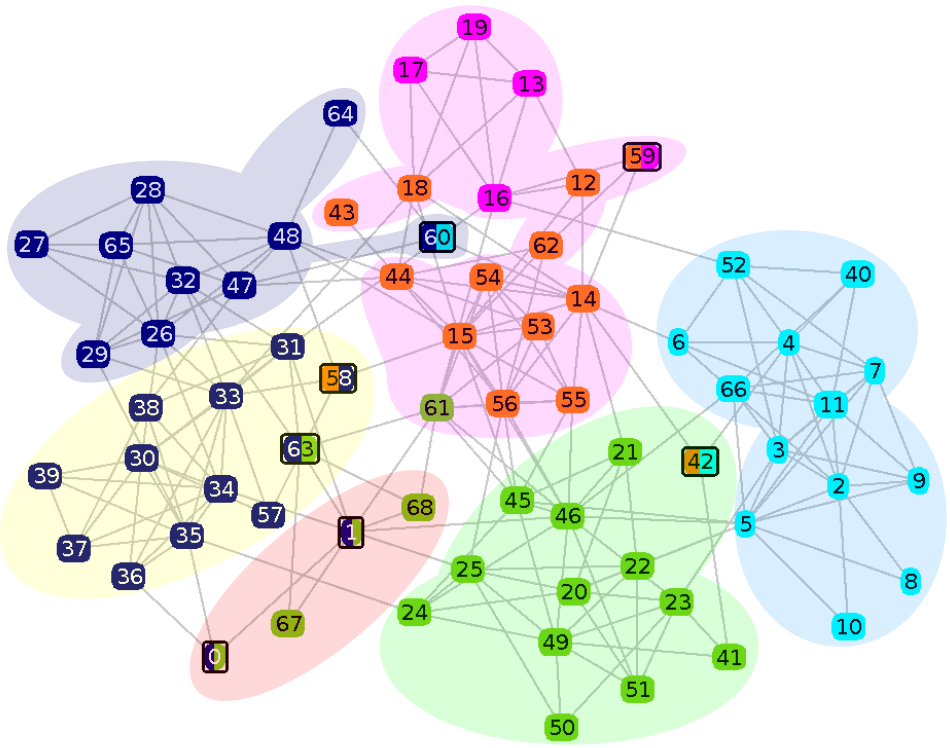


FIGURE 4.7 – Visualisation des résultats de la détection de communautés recouvrantes sur Highschool Network (Xie *et al.*, 2013). Couleur de sommet : communauté selon LOCNeSs, couleur de fond : vérité terrain. Les sommets multi-appartenants sont cerclés de noir et associés à deux couleurs.

Vérité terrain

La figure 4.7 montre six communautés « vérité terrain » (couleurs de fond) correspondent aux différentes classes (comprendre classe au sens d'un niveau scolaire, voir tableau 4.1) auxquelles les élèves appartiennent. On remarque que les amitiés ont tendance à être plus présentes au sein d'une même classe et relativement plus présentes entre classes proches. On remarque également un biais dans la taille de l'échantillon représenté : il y a seulement 4 élèves de 12^{th} *grade*, contre une douzaine pour les *grades* 8 et 11 par exemple.

La communauté du *grade* 9 est, selon les auteurs de l'étude, divisée en deux sous-groupes distincts (sommets #3, #13, #16, #17, #18, #19 d'une part, le reste d'autre part) correspondant à des élèves de couleurs de peau différentes.

Résultats de LOCNeSs

La figure 4.7 présente les résultats d'une exécution de LOCNeSs sur le High School Network, qui conduit à la détection de 5 communautés (couleurs des sommets) et 7 sommets multi-appartenants à 2 communautés simultanément (bi-colorés). On observe que LOCNeSs ne détecte pas la communauté rouge (sommets #0, #1, #67, #68), la plus petite du

Couleur de fond	**Classe**	**Equivalent français**
Cyan	7^{th}	$5^{\text{ème}}$ (12 ans)
Vert	8^{th}	$4^{\text{ème}}$ (13 ans)
Magenta	9^{th}	$3^{\text{ème}}$ (14 ans)
Bleu	10^{th}	2^{nde} (15 ans)
Jaune	11^{th}	$1^{\text{ère}}$ (16 ans)
Rouge	12^{th}	Terminale (17 ans)

TABLE 4.1 – Vérité terrain des communautés du réseau *High School Network* (Xie et al., 2013)

graphe, mais identifie les sommets #0 et #1 comme multi-appartenants, les plaçant dans les communautés jaune et verte. La différence de densité des sous-graphes n'est alors pas suffisante pour créer une communauté à part et #67, #68 sont placés dans la communauté verte. On peut également noter que le sommet #63, lié aux sommets #1, #67 et #68, est également vu comme multi-appartenant.

Le même phénomène, en plus important, conduit à ne pas découvrir la communauté jaune, qui est vue comme une continuité de la communauté bleue. La communauté magenta est scindée en deux, retrouvant la division en deux sous-groupes évoquée dans la description de l'étude (voir annexe B.2.2, p. 190). LOCNeSs réussit à capturer les liens plus distendus et répercute cette scission en deux communautés distinctes.

Parmi les sommets cités par Xie et al. (2013) comme justifiant une multi-appartenance, on trouve :

— des sommets en bordure, avec de fortes connexions dans deux communautés : #26, #32, #33, #45, #46, #61. LOCNeSs n'en a identifié parmi eux. On pourrait y voir le signe que cette caractéristique n'est pas bien capturée par l'algorithme, néanmoins les sommets #26 et #32 font partie de la communauté bleue alors que le sommet #33 fait partie de la communauté jaune, or LOCNeSs a regroupé ces deux communautés en une seule lors de la détection, en raison de la densité des liens entre les sommets en faisant partie.
  Le sommet #61 est classé uniquement dans la communauté verte, probablement car 5 de ses 8 voisins ont été classés dans cette communauté, ou en multi-appartenance, parfois de façon erronée.
— des sommets constituant des « interfaces » entre plusieurs communautés : #12, #18, #59. Le sommet #59 a bien été identifié comme multi-appartenant par LOCNeSs.
— des sommets « ponts » qui n'ont pas d'appartenance forte claire à une communauté, tel le sommet #42. Ce sommet est effectivement identifié comme multi-appartenant par LOCNeSs.

A titre de comparaison, les 14 algorithmes testés par Xie *et al.* sur ce graphe trouvent entre 2 et 20 communautés, et entre 1 et 34 sommets multi-appartenants.

4.5 Bilan

Nous avons exposé dans ce chapitre l'intérêt de détecter des communautés non-disjointes, ou communautés recouvrantes, comportant des sommets multi-appartenants. Nous en avons détaillé les différents types ainsi que les critères permettant de les identifier : sémantiques et structurels.

Nous avons présenté LOCNeSs, une extension de VOLCAN visant à détecter spécifiquement des sommets multi-appartenants, en particulier de type isolé.

Enfin, à travers plusieurs expériences, nous avons montré que LOCNeSs garde l'essentiel des propriétés de VOLCAN : en particulier les communautés produites sont stables et de bonne qualité par rapport à une vérité terrain en benchmarking. D'autre part, les sommets multi-appartenants détectés sont bien isolés et semblent pertinents d'une part sur des graphes artificiels, mais également sur un graphe tiré de données réelles.

Nous clôturons ainsi la partie concernant la détection de communautés sur graphe statique. La partie suivante présente une autre extension de nos travaux à la détection, dite dynamique, de communautés sur des graphes variant en fonction du temps.

Deuxième partie

Détection dynamique de communautés orientée sommet

Introduction

Les réseaux rencontrés dans le monde réel sont fréquemment dynamiques, c'est-à-dire qu'ils évoluent au cours du temps. Ainsi, dans les réseaux mobiles opportunistes présentés dans le chapitre 1.4, p. 30, les nœuds peuvent joindre ou quitter le réseau en fonction de leur proximité aux autres nœuds, déterminés par leur position géographique. Par conséquent, les liens du réseau évoluent également, ainsi que la structure communautaire induite.

De même dans l'exemple du laboratoire de recherche, la situation n'est pas figée et change au gré des arrivées et départs de chercheurs ainsi que des collaborations qui commencent ou s'arrêtent : l'apparition d'un nouveau sommet peut être assimilée au recrutement d'un nouveau chercheur et la disparition à un départ à la retraite ou un changement de laboratoire. L'apparition d'une arête entre deux chercheurs signale une nouvelle collaboration et sa disparition une collaboration qui cesse. Les communautés auxquelles ils appartiennent (département du laboratoire par exemple) évoluent donc en conséquence.

On appelle généralement *graphe variant avec le temps* (time-varying graph), *graphe évolutif* (evolving graph) ou *graphe dynamique* (dynamic graph), cette dernière dénomination étant retenue dans le manuscrit, les graphes modélisant de telles situations.

Passer du paradigme statique au paradigme dynamique soulève plusieurs difficultés. La fréquence et l'importance de l'évolution des communautés au cours du temps ajoutent des contraintes très fortes pour les méthodes de détection. En réponse à ce problème, nous proposons dans cette partie une extension de LOCNeSs baptisée DynLOCNeSs, qui traite les graphes dynamiques, en conservant les avantages conférés par l'approche locale et décentralisée.

Le chapitre 5 propose un état de l'art, portant dans un premier temps sur une présentation des graphes dynamiques, de leurs représentations ainsi que des communautés détectables dans ces graphes, puis dans un second temps sur les méthodes de détection de communautés existantes pour ce type de graphes.

Le chapitre 6 présente ensuite l'algorithme DynLOCNeSs, adaptation de LOCNeSs pour le traitement des graphes dynamiques, qui nécessite de modifier la mesure de proximité entre sommets ainsi que d'introduire un mécanisme de traitement et de propagation des évolutions, que nous réalisons à l'aide de la mesure proposée CWCN.

Enfin, le chapitre 7 présente l'application réelle à un réseau mobile opportuniste social : l'aide à l'orientation utilisant la structure communautaire. Le principe repose sur

une recommandation d'individus à visiter, basée sur les communautés. Il est incarné par algorithme SWAGG, modélisant une telle stratégie de rencontre.

Chapitre 5

Méthodes de détection pour le cas dynamique

Sommaire

Après avoir rappelé les motivations du traitement des réseaux dynamiques dans la section 5.1, ce chapitre présente les principales approches pour leur représentation, dans la section 5.2. Il décrit ensuite, dans la section 5.3, les méthodes de détection de communautés pour les réseaux représentés comme des collections d'instantanés, qui constituent la représentation la plus fréquemment mise en œuvre.

5.1 Motivations

Les systèmes complexes dont les interactions sont formalisées par des réseaux, eux-mêmes représentés mathématiquement comme des graphes, présentent souvent la particularité d'être dynamiques, c'est-à-dire de voir leurs ensembles de sommets ou d'arêtes évoluer. C'est le cas dans de très nombreux domaines, parmi lesquels on peut par exemple citer la biologie (Yule, 1925), la sociologie (Moreno & Jennings, 1938, Watts & Strogatz, 1998) ou la physique, en particulier la mécanique statistique (Barabási & Stanley, 1995, Schweitzer, 1997), entre autres (Marsden et al., 1993). A ce sujet, le livre de Dorogovtsev et Mendes (2002) apporte un éclairage intéressant sur la vision des réseaux complexes dynamiques par le prisme de la mécanique statistique.

Pour de tels réseaux dynamiques, la structure communautaire évolue alors aussi : les relations entre nœuds étant changeantes, les sous-groupes de nœuds fortement connectés, donc les communautés, changent également.

Plusieurs motivations président à la pertinence de la tâche de détection de ces évolutions de communautés. En effet, celle-ci fournit d'abord des informations riches, ainsi, en considérant l'exemple du laboratoire de recherche, le grossissement significatif puis la scission en deux d'une communauté peut indiquer qu'une équipe de recherche, portée sur une certaine thématique, a vu l'arrivée de nombreux nouveaux chercheurs et s'est divisée afin de constituer deux équipes aux thématiques plus précises. Cela peut être causé par le développement soudain et rapide d'un champ de recherche assez nouveau, spécifique et actif.

De plus, la détection de communautés dynamiques peut permettre de répondre à d'autres problèmes d'extraction d'informations, par exemple la détection d'événements (Leskovec et al., 2005, Greene et al., 2010) : l'évolution des communautés est interprétée comme une réaction du système complexe à un événement extérieur que l'on peut ainsi identifier, avant de chercher à le caractériser.

Ce chapitre présente les méthodes proposées pour la détection de communautés dynamiques, après avoir décrit les représentations formelles qui peuvent être considérées. Le chapitre 6 présente l'extension de LOCNeSs que nous avons proposée pour répondre à cette tâche.

5.2 Représentations de graphes dynamiques et modèles d'évolution

Cette section décrit quelques modélisations existantes pour les réseaux dynamiques : elle discute d'abord des évolutions qui peuvent être considérées puis détaille des modèles.

5.2.1 Evolution des réseaux

On peut choisir de considérer des graphes dont les sommets et arêtes, ou bien seules les arêtes, peuvent changer. Il faut noter que ce dernier cas permet en fait de traiter aussi le cas où les sommets et arêtes changent : il consiste à considérer qu'un sommet a disparu lorsqu'il ne possède plus d'arête, on dit aussi qu'il est mort. De façon symétrique, l'apparition d'un sommet consiste à ajouter une ou plusieurs arêtes à un sommet mort, le reliant à une composante connexe du graphe. Ce modèle considérant l'ensemble des sommets constant est largement utilisé (Granell et al., 2015).

Il faut noter que le choix de la représentation doit se faire par compatibilité avec la situation modélisée et de la méthode de traitement du graphe considérée, celle-ci pouvant être dédiée à des types de représentation spécifiques.

Tous les systèmes dynamiques n'évoluent pas de la même façon et, par exemple, un réseau social n'a pas le même type d'évolutions qu'un réseau de protéines. Différents modèles, génératifs ou descriptifs, ont été proposés, généralement indépendants de la représentation choisie pour le réseau (cf. section suivante). Ainsi, dans le cas des réseaux sociaux, les modèles génératifs incluent l'approche de Barabási et Albert (1999) basée l'attachement préférentiel, ainsi que celle de Watts et Strogatz (1998) reproduisant l'effet petit monde (cf. section 1.2.3.1, p. 15). Concernant les modèles descriptifs, un mécanisme souvent cité pour expliquer l'évolution d'un réseau social, également utilisé dans les définitions de communautés par similarité, est l'homophilie (cf. 1.2.3.6, p. 19).

5.2.2 Formalisme des graphes dynamiques

Après avoir discuté des éléments du graphe susceptible d'évoluer, cette section présente trois principaux formalismes pouvant être adoptés afin de représenter un graphe dynamique : la collection d'instantanés (que nous utilisons), le graphe à dimension temporelle et le flot de liens.

5.2.2.1 Collections d'instantanés

La représentation la plus utilisée est la *collection d'instantanés* (*snapshots*, Aynaud et al., 2013). Certains auteurs utilisent le terme *temporal graph* pour la désigner, à ne pas confondre avec les graphes à dimension temporelle évoqués ci-dessous. Très intuitive, elle

consiste en un ensemble d'instances $\mathcal{G}$ du même graphe G à différents instants $t_0...t_r$, soit :

$$\mathcal{G} = \{G_{t_0}, G_{t_1}, ..., G_{t_r}\}$$

Un graphe $G_t = (V_t, E_t)$ est donc un instantané de G au temps t. On peut noter que cette méthode ne permet de décrire que des intervalles de temps discrets, qui sont néanmoins très majoritairement utilisés par les algorithmes.

La discrétisation constitue son principal inconvénient : les changements ayant eu lieu entre deux instantanés ne peuvent être pris en compte. Des communautés peuvent fortuitement sembler différentes entre deux instants t_i et t_{i+1} si du bruit (des sommets « parasites ») apparaît, alors qu'elles sont très semblables entre t_i et t_{i+2}. C'est l'une des principales causes d'instabilité de la structure des communautés trouvées au cours du temps (Aynaud & Guillaume, 2010).

5.2.2.2 Graphes à dimension temporelle

Nous regroupons sous le terme *graphes à dimension temporelle* (généralement appelés *time-varying graphs* en anglais) les représentations visant à définir un graphe de type continu, comprenant dans sa nature la dimension évolutive. Les principales approches associent à chaque sommet et/ou chaque arête une étiquette ou une fonction (Mitra et al., 2012) décrivant son évolution en fonction du temps. On peut ainsi en tirer une fonction d'évolution globale du graphe.

Casteigts et al. (2011) ont également fourni un cadre formalisant la temporalité des graphes dynamiques, en lien avec l'utilisation dans les réseaux mobiles et le calcul parallèle. A travers la synthèse de différentes propriétés de ces réseaux, principalement basée sur la connectivité, ils introduisent des fonctions de présence et de latence sur les arêtes, permettant de modéliser des interactions temporelles.

5.2.2.3 Flots de liens

Un autre formalisme utilise les *flots de liens* (Viard et al., 2016, Gaumont, 2016). Ce flot est défini comme une collection de triplets (t, u, v) où $t \in \mathcal{T}$ (ensemble des indicateurs temporels considérés, comme des pas de temps) est un indicateur temporel et $u, v \in V$ sont des sommets. Un triplet indique la présence d'une arête (ie. l'existence d'une interaction) entre u et v au temps t. Cette modélisation peut être vue comme à une série temporelle, dans le sens où elle constitue une série de triplets, cependant elle est résolument tournée vers l'analyse des réseaux. Les principales propriétés des graphes ont par exemple été étendues aux flots de liens (Viard, 2016).

Ils représentent un état intermédiaire, permettant de leur appliquer des méthodes d'analyse conçues pour des séries temporelles comme pour des graphes.

5.2.3 Evolution des communautés : six scénarios

La détection sur graphe dynamique nécessite d'introduire de nouvelles notions, en particulier sur l'évolution des communautés. En effet, l'évolution des arêtes d'un graphe entraîne l'évolution des communautés, un phénomène exprimable sous forme de différents scénarios, dont on retient généralement six variantes principales (Palla et al., 2007) :

- *naissance* (resp. *mort*) : des sommets qui ne formaient pas (resp. formaient) une communauté en forment désormais (resp. n'en forment désormais plus) une,
- *accroissement* (resp. *rétrécissement*) : une communauté existante s'agrandit (resp. rétrécit), ie. comporte plus (resp. moins) de sommets et donc d'arêtes,
- *fusion* : plusieurs communautés existantes s'agrègent pour former une nouvelle communauté plus grande,
- *éclatement* : une communauté se scinde en plusieurs communautés plus petites.

Il est important de noter que ces notions sont indépendantes de, et peuvent s'adapter à, toute définition d'une communauté.

5.3 Détection de communautés sur collection d'instantanés

Bien qu'apparue récemment, la détection de communautés sur graphes dynamiques est un champ de recherche très riche et une abondante littérature existe à ce sujet (Aynaud et al., 2013). Cette section considère le cas où le graphe est représenté sous la forme d'une collection d'instantanés, choix que nous avons fait pour baser la conception de la méthode DynLOCNeSs, proposée dans le chapitre 6.

Nous détaillons dans cette section les méthodes globales à traitement temporel a posteriori ainsi qu'à traitement temporel intégré, c'est-à-dire ayant lieu pendant le processus de détection, puis les méthodes locales et décentralisées, qui nous intéressent spécifiquement dans le cadre de cette thèse, dont le traitement temporel peut également être intégré ou réalisé à posteriori.

5.3.1 Traitement temporel a posteriori

Static recomputation, matching, approche en deux temps

Sur une collection d'instantanés, une approche simple consiste à effectuer une détection statique globale sur chaque instantané du graphe puis à agréger le résultat (*static recomputation*, Aynaud et al., 2013).

Cette agrégation cherche généralement à établir des correspondances entre les communautés des différents pas de temps (*matching*). Cette opération peut être réalisée de façon naïve, comme la première approche proposée par Hopcroft et al. (2004) qui utilise une mesure basée sur la notion de *natural community* exprimée par le pourcentage d'éléments en commun dans une communauté, ou être plus évoluée (Tantipathananandh et al., 2007).

Le principal inconvénient de cette méthode est qu'elle effectue la liaison a posteriori, alors que deux états du graphe peuvent présenter des structures communautaires assez différentes. Ainsi, la liaison n'est pas toujours cohérente. De plus, l'utilisation de la *static recomputation* a tendance à produire des communautés instables (Aynaud & Guillaume, 2010).

Optimisation et modularité

Görke et al. (2010) proposent une implémentation dynamique de détection de communautés utilisant une optimisation incrémentale de la modularité. Mucha et al. (2010) proposent également une version de la modularité multi-niveaux, adaptée pour le traitement de graphes dynamiques. Lin et al. (2008) se basent sur une somme pondérée entre deux états successifs du graphe nommée alpha-cost, couplée à des processus stochastiques et à la modularité.

Ce type de méthodes souffre des mêmes problèmes que la *static recomputation*. En effet un changement mineur dans le graphe peut impacter lourdement la structure communautaire lors d'une détection globale : les résultats sont très instables, d'autant plus si la méthode statique utilisée est non-déterministe. C'est notamment la constatation dressée par Aynaud et Guillaume (2010), qui utilisent l'algorithme de Louvain appliqué à chaque graphe de la collection d'instantanés. Les auteurs ont ensuite proposé une version gommant les instabilités (Aynaud, 2011).

5.3.2 Traitement temporel intégré

Nous présentons ici deux types de méthodes ne faisant pas appel à un traitement temporel a posteriori mais ayant lieu de façon globale, au cours de la détection : la *temporal smoothness* puis la recherche de motifs de co-évolution.

Temporal smoothness

Cette représentation consiste à lier la détection des communautés à des événements ayant lieu dans le graphe. Il faut donc caractériser des événements, puis les retrouver dans l'évolution du graphe. Ainsi, Chakrabarti et Faloutsos (2006) et Chakrabarti et al. (2006) ont introduit, dans le champ du clustering évolutionnaire qui est la version temporelle du clustering de graphe évoqué dans la section 2.2, p. 41, la notion de *change point* et de *temporal smoothness*. Cette approche repose sur l'existence de *points de changement* constituant une rupture dans la continuité des états du graphe, un élément modificateur important affectant la structure et utile pour détecter des événements. Les auteurs « lissent » les communautés à travers le temps en calculant une correspondance de communauté à communauté entre deux pas de temps en tentant de garder une structure globalement co-

hérente, et ce jusqu'à ce que le lissage ne soit lui-même plus cohérent (impossible d'établir une correspondance) : un point de changement est alors atteint.

Motifs de co-évolution

Une autre approche intégrant le traitement temporel utilise la fouille de motifs de co-évolution ou *co-evolution pattern mining* (Plantevit et al., 2005) : l'évolution des sommets est considérée comme une séquence de descripteurs de l'état des sommets, qui comprennent des mesures topologiques. Les communautés sont identifiées à des séquences émergentes, c'est-à-dire ayant un fort support dans le graphe. La détection en elle-même utilise ensuite des méthodes de fouille de motifs séquentiels fréquents (Desmier, 2014,Orman et al., 2014).

5.3.3 Méthodes locales et décentralisées

Le problème de la détection de communautés dynamiques a églament été abordé dans le cadre des méthodes locales et décentralisées, recherchant les mêmes avantages que dans le cas statique (cf. section 2.3, p. 45). Nous distinguons ici les méthodes locales, orientées-sommet et les méthodes décentralisées de type multi-agents.

Méthodes orientées-sommet

Les méthodes orientées-sommet de détection de communautés dynamiques étendent pour la plupart des méthodes statiques. Gao et al. (2016) proposent une approche basée sur la détection de leaders (cf. méthode Top-Leaders, Rabbany et al., 2010) : elle forme des communautés initiales disposant chacune d'un leader à l'aide de Top-Leaders. A chaque pas de temps les communautés précédemment identifiées sont décomposées et l'appartenance de chaque sommet à sa communauté recalculée. Des opérations de fusion sont ensuite effectuées, les communautés étant ainsi reformées à quelques changements près, liés à leur évolution, à la manière d'un algorithme des k-moyennes (Jain et al., 1999). Les leaders sont mis à jour pour refléter la structure résultant de cette fusion. Cette approche est relativement coûteuse, le cycle d'éclatement-fusion des communautés étant répété à chaque pas de temps.

mux-LICOD (Hmimida & Kanawati, 2015) est une adaptation de LICOD (Yakoubi & Kanawati, 2014), détaillé dans la section 4.2.3, p. 95, permettant la détection dans des graphes multi-couches (réseaux multiplexes) et peut donc être appliquée à des collections d'instantanés. mux-LICOD utilise des mesures de centralité de degré et des calculs de plus courts chemins afin de réaliser les associations leaders et followers, ainsi qu'une phase d'agrégation répétée jusqu'à stabilisation, dont l'étude expérimentale réalisée par les auteurs ne permet pas d'évaluer si elle est rapide ou non.

Les algorithmes OLEM et OLTM, proposés par Pan et al. (2014), utilisent également une optimisation locale liée à un modèle génératif de l'évolution du graphe basé sur l'at-

tachement préférentiel (cf. section 1.2.3.6, p. 19). Ils traitent ensuite une à une les arêtes du graphe en affectant un sommet à la communauté avec laquelle il maximise le gain d'espérance de modularité.

Les approches citées précédemment possèdent un même inconvénient principal : les étapes de détection au cours de l'évolution du graphe sont lourdes (respectivement éclatement-reconstitution des communautés, calcul de centralités et plus courts chemins ou calcul de modularité). Elles perdent ainsi un des avantages majeurs des méthodes orientées-sommet qui nous intéresse : leur légèreté et flexibilité, qui permet de les utiliser dans le contexte d'un réseau mobile ad-hoc décentralisé (cf. section 1.4, p. 30).

La méthode de Zakrzewska et Bader (2015) calcule une structure initiale par expansion autour d'une graine, puis une fonction objectif ainsi qu'un ensemble de règles pré-définies afin de réaliser les mises à jour d'appartenance des sommets à leurs communautés. Les expériences réalisées par les auteurs montrent un intérêt en terme de gain de vitesse par rapport à des méthodes de type *static recomputation*, ce qui est attendu pour un algorithme utilisant des mises à jour dynamiques, mais la comparaison de la qualité des communautés produites n'est pas démontrée par les auteurs.

Méthodes multi-agents

La méthode iLCD (Cazabet & Amblard, 2011) est une méthode de détection par processus multi-agents déjà évoquée dans le chapitre 4.2.2, p. 93 : chaque sommet représente un agent pouvant effectuer cinq actions (création d'une communauté, intégration ou rejet d'une communauté, formation ou destruction d'un lien avec un autre agent), un processus de simulation multi-agents définit les communautés au cours du temps, au fur et à mesure des actions qu'ils effectuent.

La méthode de Zardi et al. (2014) pondère le graphe en utilisant le principe de l'homophilie (cf. section 1.2.3.6, p. 19) et utilise une simulation multi-agents dans laquelle chaque agent a une préférence pour certains de ses voisins, grâce à la pondération. L'agrégation des agents forme les communautés.

5.4 Bilan

Ce chapitre a présenté les principales approches permettant de traiter des réseaux dynamiques, en détaillant les modèles et représentations de graphes et communautés dynamiques ainsi que leur évolution, puis en décrivant différents algorithmes utilisant la représentation par collection d'instantanés de graphes, dont des méthodes locales et orientées-sommet. Dans le chapitre suivant, nous présentons l'algorithme DynLOCNeSs, adaptation de LOCNeSs pour traiter des graphes dynamiques, qui constitue une approche décentralisée pour cette représentation des graphes.

Chapitre 6

DynLOCNeSs : détection de communautés dynamiques

Sommaire

Ce chapitre présente une extension de LOCNeSs, nommée *DynLOCNeSs*, destinée à traiter des graphes dynamiques.

Nous présentons tout d'abord, dans al section 6.1, une vue générale de DynLOCNeSs, les objectifs qu'il poursuit et les attentes quant à son fonctionnement, ainsi que les principes généraux sur lesquels il repose. Nous décrivons ensuite DynLOCNeSs en détails, dans la section 6.2 et discutons de ses propriétés. La section 6.3 introduit une nouvelle mesure de préférence entre sommets, CWCN, adaptée pour le cas de DynLOCNeSs. Nous proposons enfin, dans la section 6.4, une étude expérimentale afin d'étudier les effets de différentes mesures de préférence entre sommets sur le modèle dynamique et de mesurer ses performances en terme de qualité des communautés.

Ces travaux ont été publié dans les actes du Workshop Complex Networks 2016 (Canu et al., 2016a).

6.1 Objectifs et principes

Le but global de DynLOCNeSs est d'utiliser les principes orientés-sommet, exploités par VOLCAN et LOCNeSs afin de permettre la détection de communautés dans des graphes dynamiques.

DynLOCNeSs repose sur l'algorithme LOCNeSs présenté chapitre 4 mais ses principes peuvent également être appliqués à VOLCAN ou toute autre méthode orientée-sommet.

Cette section décrit le type de graphe dynamique considéré puis les principes généraux de DynLOCNeSs.

6.1.1 Type de graphe dynamique considéré

Nous avons présenté dans la section 5.2.2, p. 115 différentes représentations possibles pour un graphe dynamique, dont les collections d'instantanés que nous avons choisies pour développer DynLOCNeSs. Bien que ce formalisme ne soit pas le plus naturel pour représenter une variation dans le temps, car il discrétise le temps qui est un phénomène continu, il constitue une approche courante (Aynaud, 2011), que l'on peut envisager de faire évoluer vers un autre formalisme par la suite. De plus, nous considérons des graphes dont seules les arêtes changent, qui ne constituent pas une limitation (cf. section 5.2.1, p. 115).

Nous définissons un *pas de temps* $t_i, i \in \mathbb{N}$ pour lequel un nouvel état du graphe G est retenu. Un nouveau pas de temps t_{i+1} survient dès qu'un changement d'arête (apparition ou disparition) est détecté. Un tel événement correspond, dans un réseau mobile du type considéré (cf. section 1.4), à l'établissement ou perte de connexion sans fil entre deux dispositifs, dû au rapprochement ou à l'éloignement des personnes les transportant. Il n'y a pas de nouveau pas de temps tant qu'aucune modification n'est apportée à G. Notons

qu'en conséquence un intervalle de temps, c'est-à-dire la durée entre deux pas de temps $|t_i - t_{i-1}|$, n'est pas nécessairement constant.

6.1.2 Principes de la méthode

DynLOCNeSs adapte l'algorithme LOCNeSs présenté dans le chapitre 4 aux graphes dynamiques, en analysant les changements qui surviennent dans le graphe, pour déterminer s'ils entraînent des changements de communautés. Pour cela, DynLOCNeSs établit d'abord une structure communautaire initiale, obtenue par exemple en appliquant LOCNeSs au graphe G_0, toute méthode de détection de communautés recouvrantes pouvant être mise en œuvre pour cette étape.

Il s'agit ensuite de déterminer si, lors des changements de structure, les leaders associés à chaque sommet peuvent et doivent rester les mêmes. Un changement parmi les leaders peut entraîner un changement de communauté, ce qui est réalisé par le biais d'un marquage des sommets du graphe. De plus, ces changements doivent être propagés au voisinage. La section suivante détaille ces étapes.

6.2 Description de l'algorithme

Après l'étape d'initialisation, qui identifie une structure communautaire dans G_0, à chaque pas de temps et pour chaque sommet, l'algorithme exploite un marquage de ses sommets, qui permet de propager les changements dans une étape de mise à jour. Ces deux composantes sont décrites ci-dessous, des propriétés de DynLOCNeSs sont ensuite examinées.

6.2.1 Marquage des sommets

Un sommet v est marqué pour signifier qu'il a changé de communauté. Un marquage effectué à t_i n'est destiné qu'aux voisins de ce sommet et n'est visible qu'à t_{i+1}. Ainsi, les voisins de v sont avertis que v a changé de communauté, a priori à cause de l'évolution d'une partie de son voisinage. Ces voisins sont donc incités à réévaluer leur propre appartenance à leur communauté : en effet un changement de communauté d'un sommet peut entraîner le changement de communauté de l'un de ses voisins.

Ainsi, si au temps t_i un sommet décèle que son voisinage a changé de marquage par rapport au temps t_{i-1}, il lance la procédure de mise à jour, décrite dans la section suivante, au temps t_i.

Ce marquage est un moyen d'accélérer la répercussion des changements locaux dans le graphe (changement de leader, changement d'appartenance à une communauté) sans recourir à une technique de propagation massive, de type inondation par exemple, que nous souhaitons éviter.

Algorithme 6 Mise à jour d'un sommet au temps t_i

Require:
$v \in V$, un sommet
$\Gamma_i(v)$, les voisins de v au temps t_i, avec leur marquage à t_{i-1}
Ensure:
$C_i(v)$, communauté de v mise à jour
1: **if** $\Gamma_i(v) \neq \Gamma_{i-1}(v)$ **then**
2: recalculer les leaders préférés de v : $A(v) \leftarrow \arg\max_{u \in \Gamma_i(v)} \sigma(v, u)$
3: **if** $A(v)$ change **ou** $\exists u \in A(v)$ marqué **then**
4: $C_i(v) \leftarrow$ communauté la plus fréquente dans $A(v)$
5: **if** $C_i(v) \neq C_{i-1}(v)$ **then**
6: marquer v au temps t_i
7: **end if**
8: **end if**
9: **end if**

6.2.2 Mise à jour d'un sommet

La procédure d'actualisation, décrite dans l'algorithme 6, est exécutée par un sommet v si un changement est survenu dans son voisinage, considéré comme une apparition/disparition ou un changement de marquage, de l'un de ses voisins. Ce changement, noté abusivement $\Gamma_i(v) \neq \Gamma_{i-1}(v)$ dans l'algorithme 6, constitue dans DynLOCNeSs la seule possibilité pouvant mener à un changement de communauté pour v.

Dans ce cas, le sommet v recalcule localement toutes les valeurs de préférence $\sigma(v, u)$ avec chacun de ses voisins $u \in \Gamma(v)$. En effet, le changement dans le voisinage peut causer la disparition d'un leader ou l'apparition d'un nouveau sommet pouvant être choisi comme leader. Si les nouvelles valeurs des préférences impliquent effectivement un changement dans $A(v)$, la communauté de v est également ré-évaluée. S'il en résulte que $C(v)$ doit changer, alors v est *marqué*, comme décrit dans la sous-section précédente.

6.2.3 Propriétés

En premier lieu il faut souligner que, à l'instar de VOLCAN et LOCNeSs, DynLOCNeSs effectue uniquement des calculs à l'échelle locale, conservant ainsi la flexibilité de l'approche orientée-sommet apportée par ces méthodes. L'étape initiale étant réalisée par LOCNeSs, ses propriétés sont identiques. Le recalcul des leaders des sommets garde les mêmes propriétés de complexité, stabilité et propagation que l'étape analogue de VOLCAN (cf. section 3.4, p. 76).

Les principaux changements sont liés à l'étape d'actualisation, qui n'était pas présente auparavant. Le mode de mise à jour des communautés, à l'aide des marquages, entraîne une propagation progressive des changements liés aux communautés concernant leurs scénarios d'évolution (voir section 5.2.3, p. 117). Si ce mode de fonctionnement permet de préserver la décentralisation et la limitation de la propagation, il ne permet néanmoins pas

de répercuter instantanément un changement local à l'intégralité des sommets. C'est pourquoi nous pouvons nous attendre à ce qu'une évolution des communautés, un scénario de scission (*split*) par exemple, ne se diffuse que progressivement dans le graphe : les sommets se répartissent en deux communautés en l'espace de plusieurs pas de temps et non d'un seul.

6.3 Mesure de préférence entre sommets : CWCN

A l'instar de VOLCAN et LOCNeSs, DynLOCNeSs repose sur une mesure de préférence afin d'identifier, du point de vue d'un sommet, les voisins avec lesquels il est le plus susceptible de former une communauté intéressante. Contrairement à VOLCAN et LOCNeSs qui utilisent l'agreement uniquement dans un contexte statique, la mesure mise en œuvre par DynLOCNeSs doit rendre compte de la *dynamique d'attraction et de cohésion* dans un ensemble de sommets formant une communauté et susceptible de la maintenir dans le temps. Par exemple, dans un réseau social, une telle mesure doit pouvoir quantifier une « force d'amitié » entre différentes personnes.

En ce sens, l'agreement est mal adapté car il ne capture pas les changements affectant des sommets de faible degré : les valeurs d'agreement ne changent que quand la liste S des voisins de plus hauts degrés change (apparition ou disparition d'un de ses sommets). Or, de nombreuses modifications affectant des sommets de plus faible degré entraînent des modifications substantielles dans les communautés.

Nous proposons donc la mesure **C**ommunity-based **W**eighted **C**ommon **N**eighbours (σ_{CWCN}) afin d'améliorer l'efficacité de la méthode DynLOCNeSs. Basée sur le nombre de voisins communs, elle prend de plus en compte un aspect lié à l'influence de l'un des deux sommets dans sa communauté, utilisant son degré pour pondérer le nombre de voisins communs. Ainsi, contrairement aux mesures présentées section 3.2.2, p. 69, elle est asymétrique :

$$\sigma_{CWCN}(u,v) = |\Gamma(u) \cap \Gamma(v)| \times d_v \tag{6.1}$$

Le sens de la relation est :
$\sigma_{CWCN}(u,v)$: u préfère v,
$\sigma_{CWCN}(v,u)$: v préfère u.
La pondération est donc apportée par le sommet « étant préféré ».

Le principe est, comme pour VOLCAN et LOCNeSs, basé sur le mécanisme d'attachement préférentiel et l'idée que les sommets de plus haut degré « attirent » plus que les sommets de plus faible degré. Néanmoins, CWCN est moins fort que la mesure d'attachement préférentiel σ_{PA} citée dans la section 3.2.2, p. 69, voir eq. 3.3 : pour deux sommets distincts on a

$$\sigma_{CWCN}(u,v) \leq \sigma_{PA}(u,v)$$

Démonstration Le nombre de voisins communs est nécessairement inférieur ou égal aux degrés des deux sommets, soit $|\Gamma(u) \cap \Gamma(v)| \leq \min(d_u, d_v) \leq d_u$. En multipliant les deux côtés de l'inégalité par d_v, on obtient le résultat souhaité. ∎

Les expériences menées dans le cadre de l'étude expérimentale ont montré que l'attachement préférentiel fusionnait rapidement les deux communautés en une et conservait cette communauté unique pour tous les pas de temps, que l'évolution suive un motif de croissance-décroissance ou de fusion-séparation.

6.4 Étude expérimentale

Cette section présente les résultats expérimentaux obtenus avec DynLOCNeSs dans deux cadres : la section 6.4.1 examine la qualité des communautés dynamiques qu'il identifie, en considérant différents motifs d'évolution et différentes mesures de préférences, la section 6.4.2 est consacrée à son temps d'exécution.

6.4.1 Qualité des communautés détectées

Nous présentons ces résultats en commençant par le protocole expérimental considéré, puis en discutant les résultats obtenus.

6.4.1.1 Protocole expérimental

Graphes considérés

Les expérimentations sont effectuées sur des graphes d'un modèle très utilisé dans le domaine (Clementi et al., 2015, Granell et al., 2015) : la *planted bissection*, c'est-à-dire considérant une seule communauté se divisant en deux selon un scénario établi. On considère deux modèles d'évolution des communautés : *grow-shrink* (accroissement suivi d'un rétrécissement) et *merge-split* (fusion suivie d'une scission). Le premier consiste à voir des sommets passer d'une communauté à l'autre, le second à fusionner, totalement et brusquement, deux communautés. Ils sont représentés respectivement sur les figures 6.1 et 6.4 dans le cas de deux communautés : l'axe des abscisses donne le temps, l'axe des ordonnées représente les identifiants des sommets, la couleur de chaque point du plan indique la communauté à laquelle appartient chaque sommet à chaque pas de temps. Pour merge-split, une large portion unicolore, de couleur blanche, signifie que tous les sommets sont regroupés dans une seule communauté. Pour générer les graphes, nous utilisons le benchmark de Granell et al. (2015), détaillé en annexe B, p. 185.

Pour chaque motif, 10 instances d'un graphe de 64 sommets pour 100 pas de temps sont générées, de densité intra-communautaire 0.5 et extra-communautaire 0.05. Ces paramètres sont ceux utilisés par les auteurs du benchmark. Le nombre de sommets de ces graphes peut

paraître faible, mais il permet une comparaison visuelle aisée des résultats de l'algorithme. Ces graphes comportent deux communautés, de 32 sommets chacun à t_0.

Critères d'évaluation

Nous utilisons la NMI et la NVI, détaillées en annexe A. Les méthodes de détection de communautés sur graphes dynamiques dépendant du modèle de dynamicité utilisé, ainsi que de divers paramètres et formats d'entrée et de sortie, il n'est pas évident de les comparer de manière non biaisée. Par exemple, les données d'entrée de iLCD (Cazabet & Amblard, 2011) sont événementielles : ajout ou suppression d'une arête. Ses données de sortie prennent la forme d'une séquence d'états des communautés, un changement d'état survient à chaque nouvel événement.

Cela signifie que, lorsqu'on utilise iLCD sur une séquence de graphes liée à un ensemble de pas de temps similaire à celle utilisée pour tester DynLOCNeSs, la structure communautaire produite peut varier plusieurs fois pour chaque pas de temps. Une agrégation de tous les états communautaires pour un pas de temps écraserait inévitablement de l'information et introduirait un biais.

De même, l'adaptation multi-pas stabilisée de l'algorithme de Louvain (Aynaud & Guillaume, 2010) utilise une séquence de graphes sur un ensemble de pas de temps, mais propose en sortie une communauté unique : il n'est pas possible de suivre l'évolution de la structure communautaire tout au long du processus de détection.

Configuration de DynLOCNeSs

Nous comparons différentes mesures de préférence entre sommets : le coefficient de Jaccard, l'attachement préférentiel, ainsi que la mesure proposée CWCN.

6.4.1.2 Résultats obtenus

Les résultats pour le motif *grow-shrink* sont présentés sous la forme de NVI et NMI moyennes obtenues sur les 10 instances de graphes sur la figure 6.2 (critères) et sous la forme d'une répartition colorée (*colormap*) des sommets sur la figure6.3 (visualisation) pour σ_{CWCN}, reprenant la représentation des figures 6.1 et 6.4. Les résultats des autres mesures ne sont pas présentés car ces dernières ne produisent qu'une seule communauté à chaque pas de temps, leur colormap est donc entièrement noire.

On peut observer que σ_{CWCN} détecte globalement bien le motif de bissection grow-shrink, excepté l'apparition d'une troisième communauté (orange). Cette communauté remplace en fait la communauté noire au début, et la blanche à la fin : DynLOCNeSs gère l'évolution croissance-décroissance comme un transfert entre deux communautés via une troisième, impactant les valeurs de NVI et NMI.

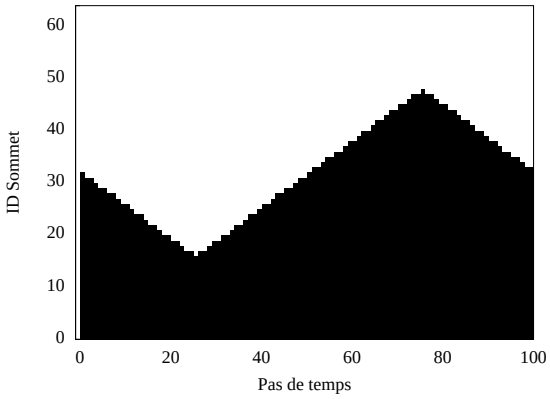


FIGURE 6.1 – Vérité terrain du motif grow-shrink (pas de temps en abscisse, identifiant sommet en ordonnée).

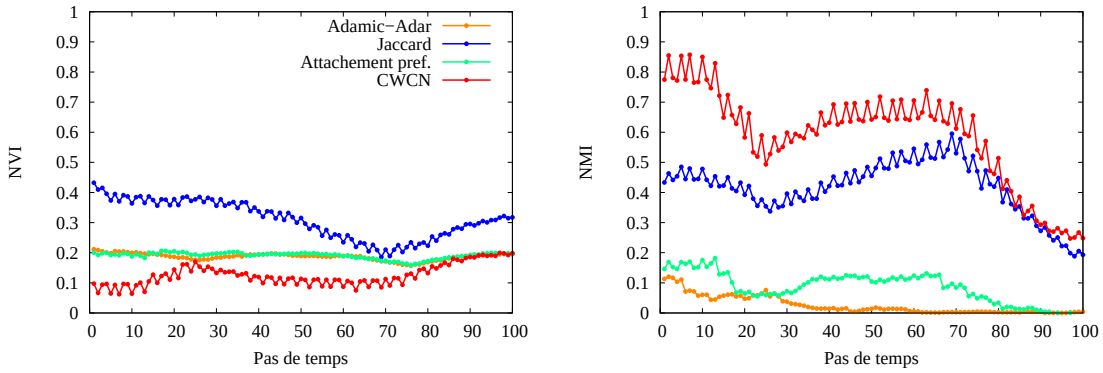


FIGURE 6.2 – Comparaison des mesures de préférences pour grow-shrink (100 pas de temps) : à gauche NVI (à minimiser), à droite NMI (à maximiser).

Cependant, le triangle caractéristique du motif grow-shrink (cf. vérité terrain figure 6.1) est clairement visible, montrant que ce motif d'évolution est correctement identifié, ce qui est moins évident avec σ_{Jac}, qui détecte 14 communautés. Bien que le triangle soit également décelable, les résultats sont très bruités et de nombreuses affectations sont fausses.

Les résultats pour le motif *merge-split* sont présentés figures 6.5 (critères) et 6.6 (visualisation). Cette fois encore, les mesures produisant une unique communauté, à chaque pas de temps et dont les colormaps sont entièrement noires, ne sont pas représentées sur la figure 6.6 .

Le motif merge-split est globalement détecté, moins bien que grow-shrink toutefois. On remarque que σ_{CWCN} trouve bien deux communautés, là où σ_{Jac} en trouve dix, mais la fusion brutale n'est pas correctement identifiée, bien que σ_{CWCN} génère moins de bruit que σ_{Jac}.

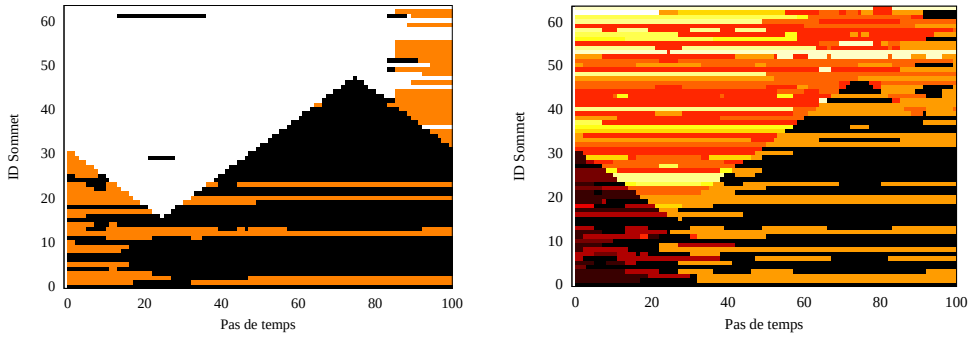


FIGURE 6.3 – Visualisation comparative des résultats pour le motif grow-shrink en utilisant σ_{CWCN} (gauche) et σ_{Jac} (droite).

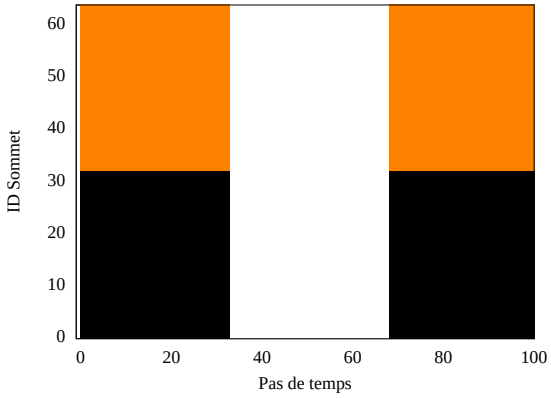


FIGURE 6.4 – Vérité terrain du motif merge-split (pas de temps en abscisse, identifiant sommet en ordonnée)

Excepté la phase de fusion (merge), on remarque que les NMI et NVI des résultats produits par σ_{CWCN} sont meilleures (resp. plus hautes et plus basses) que pour les autres méthodes. La NVI nulle, donc parfaite, de σ_{AA} et σ_{PA} durant la phase de fusion est due à un effet de bord du choix du modèle de planted bissection : la mauvaise qualité de détection de ces deux similarités fait qu'elles ne produisent qu'une seule communauté à chaque pas de temps. Incidemment, cela est conforme à la vérité terrain lors de la fusion, mais leur est préjudiciable lorsqu'il y a deux communautés.

6.4.2 Temps d'exécution

Cette section examine les performances de LOCNeSs en termes de temps d'exécution sur six graphes de 64, 128, 256, 512, 1024 et 2048 sommets, générés avec les mêmes paramètres de densité que les expériences précédentes : 0.05 de densité d'arêtes intra-communautaire et 0.5 de densité d'arêtes inter-communautaire, sur 10 pas de temps. Ainsi, le graphe de 1024 sommets comprend approximativement 375,000 arêtes sur l'ensemble des 10 pas de temps.

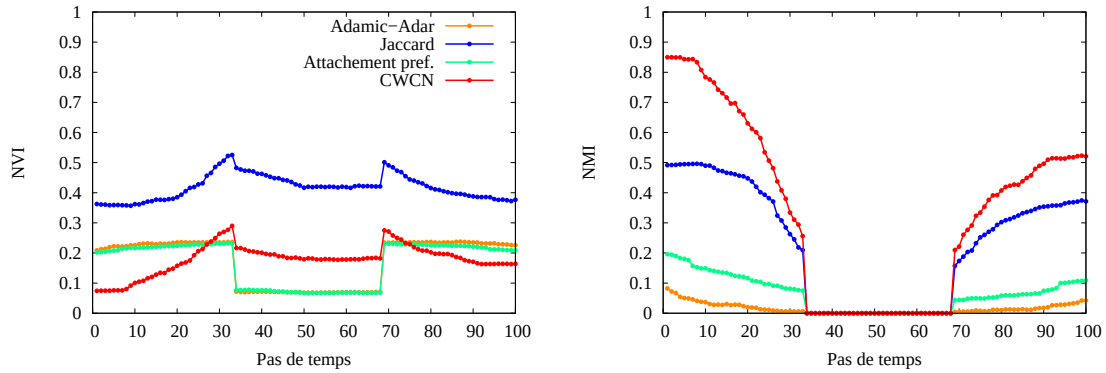


FIGURE 6.5 – Comparaison des mesures de préférences pour merge-split (100 pas de temps) : à gauche NVI (à minimiser), à droite NMI (à maximiser)

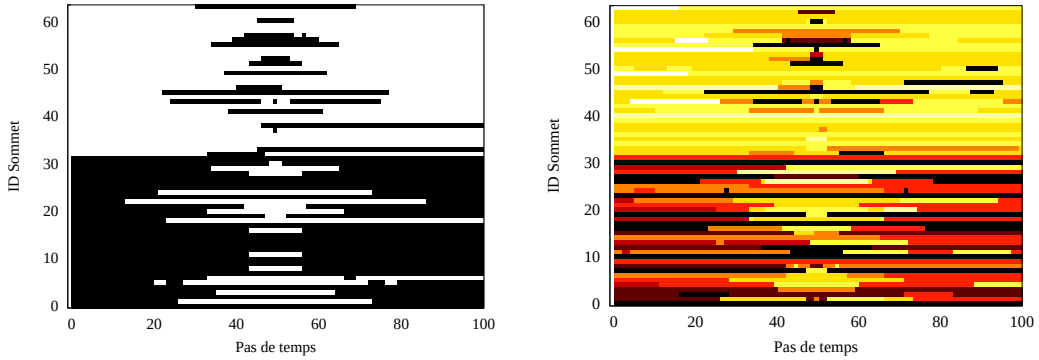


FIGURE 6.6 – Visualisation comparative des résultats pour le motif merge-split en utilisant σ_{CWCN} (à gauche) et σ_{Jac} (à droite)

Nous mesurons le temps moyen d'exécution, sur 5 instances, de DynLOCNeSs avec σ_{CWCN} et iLCD pour traiter chaque graphe. La plateforme utilisée est un PC Linux Debian, processeur Intel Core i7-2600K CPU @ 3.40GHz Workstation avec 16GB de RAM.

Les résultats sont présentés sur la figure 6.7. On peut observer que le temps de traitement de iLCD est bien plus élevé que celui de DynLOCNeSs, de plus sa courbe est parabolique là où celle de DynLOCNeSs est se rapproche d'une courbe linéaire.

DynLOCNeSs dispose donc d'un avantage significatif pour analyser un grand nombre d'arêtes : des grands graphes sur un petit nombre de pas de temps ou de petits graphes analysés sur un grand nombre de pas de temps.

6.5 Bilan

Nous avons présenté dans ce chapitre DynLOCNeSs, un algorithme de détection de communautés pour graphe dynamique. Cet algorithme propose d'utiliser les principes de

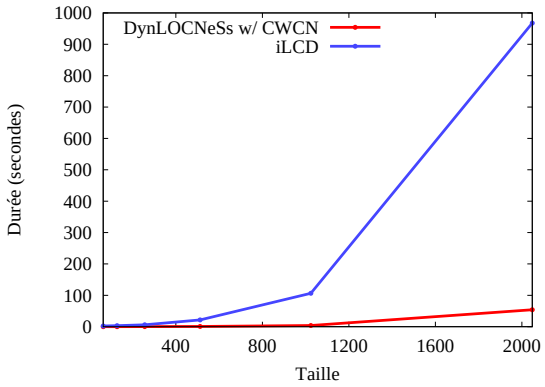


FIGURE 6.7 – Vitesse d'exécution en fonction de la taille du graphe

VOLCAN et LOCNeSs de détection orientée-sommet et de les combiner avec un cycle de mise à jour des leaders et répercussion des changements induits. Il utilise de plus une nouvelle mesure de préférence, adaptée aux particularités des graphes dynamiques comportant une structure communautaire.

Les expériences menées montrent que DynLOCNeSs offre des résultats prometteurs sur des graphes synthétiques et que CWCN est plus adaptée pour son utilisation que les autres mesures existantes considérées. DynLOCNeSs a également l'avantage de s'exécuter rapidement, par rapport à des méthodes existantes comme iLCD (Cazabet & Amblard, 2011).

Chapitre 7

Application : orientation de personnes lors d'un événement

Sommaire

Ce chapitre présente une application de nos travaux, développée au cours du projet Homo Textilus financé par l'Agence Nationale de la Recherche, sur le vêtement interactif. Il vise de façon générale à prospecter, concevoir, réaliser des prototypes de vêtements interactifs intégrant des composants électroniques permettant entre autres la communication sans fil (vêtements dits *intelligents*). Le volet exploré par nos travaux porte sur leur exploitation et vise à les utiliser pour favoriser les interactions sociales. Plus précisément, nous nous sommes intéressé aux possibilités d'amélioration de l'orientation d'une personne au cours d'un événement pendant lequel tous les participants portent un vêtement intelligent, c'est-à-dire dans notre cas communicant et intégrant un écran. Il s'agit de favoriser les rencontres permettant de faire connaissance avec un participant cible, en proposant une stratégie de recommandation exploitant les communautés du graphe social représenté par le biais des vêtements intelligents.

La première section présente le contexte du projet, les vêtements intelligents ainsi que leur exploitation envisagée pour favoriser les interactions sociales. La section 7.2 formalise la tâche considérée : utiliser des communautés détectées en temps réel afin d'assister des personnes dans leurs interactions au cours de l'événement. La section 7.3 présente l'algorithme SWAGG proposé pour y répondre et la section 7.4 en propose une évaluation expérimentale.

Ces travaux ont fait l'objet de la publication de Canu et al. (2014).

7.1 Contexte

Cette section présente le cadre dans lequel les travaux ont été effectués, la problématique du projet ANR Homo Textilus. La première sous-section introduit les vêtements dits intelligents et la seconde expose des exploitations envisagées de tels vêtements qui ont été proposées dans la littérature pour enrichir certaines interactions sociales.

7.1.1 Vêtements intelligents et affichage dynamique

Alors que les accessoires connectés tels que les montres, bracelets et autres *wearables* ont récemment émergé sur le marché grand public et semblent promis à un certain succès commercial, les vêtements intégrant de l'électronique constituent leur suite annoncée (Suh et al., 2010).

A usage technique, de mode ou prêt-à-porter, ces nouveaux habits embarquant capteurs, transmetteurs ou afficheurs sont actuellement en phase active de développement. L'essentiel du marché de ces vêtements, dits intelligents, semble se destiner à un usage de loisir grand public (Mann, 1996). Les composants et technologies intégrés prévoient, dans la plupart des cas, de pouvoir communiquer directement entre eux ou via un smartphone ou un autre wearable déjà connecté. Ils auront ainsi la possibilité de former un réseau du type mobile social opportuniste (OMSN) décrit en début de manuscrit (section 1.4, p. 30).

Un prototype développé par le laboratoire GEMTEX de l'ENSAIT à Roubaix, partenaire du projet ANR Homo Textilus, est un T-shirt embarquant un afficheur à encre électrochrome [19] directement cousu sur le textile, permettant d'afficher des formes géométriques représentant des symboles qui ont la possibilité de changer dynamiquement grâce aux propriétés de l'encre.

Le projet s'est orienté vers l'utilisation de ce type de vêtement à des fins d'interactions sociales, en particulier de rencontres interpersonnelles : le T-shirt augmenté devient un moyen d'identification et de reconnaissance dans une situation où plusieurs dizaines de personnes sont rassemblées dans un même lieu.

7.1.2 Exploitation pour interactions sociales

Des expériences ont été conduites dans des contextes similaires, en particulier au MIT (Laibowitz et al., 2006, Paradiso et al., 2010), avec pour but d'étudier les dynamiques sociales humaines lors d'interactions de groupe, conjointement à l'utilisation de dispositifs visuels électroniques.

Ainsi, lors d'un salon du type salon de l'étudiant, les chercheurs du MIT (Paradiso et al., 2010) ont équipé 100 adolescents de badges intégrant une matrice de Diodes Electro-Luminescentes, permettant de former des motifs relativement simples et de les changer . Ces badges étaient communicants et permettaient par exemple l'échange individuel (d'un porteur vers un autre) de cartes de visite virtuelles ou le vote en temps réel sur certaines propositions émises lors de présentations données au cours du salon. Ils avaient également disposé des « kiosques », permettant aux participants d'accéder à des fonctionnalités au niveau du réseau global des porteurs de badge, afin de compléter les échanges individuels.

L'étude menée sur cette expérience montre, entre multiples résultats, que ces dispositifs de communication permettent de retrouver des personnes, selon un principe ludique : un porteur de badge cherchant un autre porteur (*cible*) se place devant un kiosque et indique la cible qu'il cherche à localiser dont le kiosque lui communique l'identifiant de badge. Désormais, lorsqu'il se trouve face à un autre participant muni d'un badge il peut, en appuyant sur un bouton du badge, faire apparaître un motif sur le badge de son interlocuteur représentant le temps écoulé depuis la dernière rencontre entre cet interlocuteur et la cible. Le porteur peut également, en maintenant le bouton, diffuser sa requête, auquel cas une indication temporelle s'affiche également sur les badges de tous les participants à proximité. Ainsi, il peut trouver sa cible en interprétant les motifs selon le principe ludique « chaud/froid » (il se rapproche ou il s'éloigne, respectivement). Les chercheurs ont tracé les interactions sur la durée du salon et ont ensuite analysé les dynamiques d'interactions sociales à l'œuvre.

19. Encre possédant 2 états colorés et passant de l'un à l'autre par phénomène d'oxydo-réduction à l'application d'un courant électrique.

Notre but est de proposer une méthode d'orientation similaire, en utilisant l'information sur la structure communautaire comme indication, et non le temps écoulé. L'intérêt est de favoriser l'orientation via des personnes ayant un lien plus fort entre elles car faisant partie de la même communauté, alors que l'horodatage du dernier contact paraît constituer une information moins riche.

7.2 Formalisation de la tâche considérée

Le travail mené dans le cadre du projet ANR Homo Textilus vise donc à proposer et évaluer par simulation la recherche d'une personne dans un environnement inconnu, à l'aide de la structure communautaire du graphe social lié à cet environnement. Cette section détaille tout d'abord, dans la section 7.2.1, l'environnement considéré, puis exprime la tâche d'identification d'interlocuteur comme la recherche d'un chemin dans un réseau social. La section 7.2.2 formalise cette tâche dans le cadre des graphes dynamiques. La section 7.2.3 compare les stratégies naturelles qui peuvent être mises en œuvre par un être humain selon des postulats explicites sur son comportement et les stratégies qui peuvent être envisagées informatiquement.

7.2.1 Environnement : job dating et identification d'interlocuteur visé

Un événement de type job dating est un salon, similaire au salon de l'étudiant décrit dans les travaux de Paradiso et al. (2010), mais où se trouvent uniquement des professionnels et des recruteurs. Ces derniers sont à la recherche de professionnels correspondant à leurs attentes en termes d'emploi, de domaine ou d'expérience.

Contrairement aux salons classiques, organisés sur le principe de stands fixes repérables au moyen d'un plan, les professionnels participant à un événement de type job dating ne sont généralement repérables qu'à un badge mentionnant leur nom et leur entreprise.

Dans ce contexte, nous souhaitons proposer des solutions améliorant l'orientation d'un recruteur, lui permettant d'optimiser ses rencontres avec les professionnels présents, afin qu'il identifie au plus vite un interlocuteur.

On suppose que ces participants portent un dispositif correspondant aux prototypes décrits dans la section 7.1, p. 134 : un objet électronique permettant d'afficher des motifs modifiables et visibles lors d'un face à face avec une personne. Ce motif visible constitue donc une information dont chaque personne dispose sur chaque autre sans interaction directe (conversation).

7.2.2 Formalisation : graphe dynamique et recherche de chemin par recommandation locale

Afin de représenter le problème dans un cadre formel, nous établissons un graphe $G = (V, E)$ des participants au salon job-dating, simple et non-orienté dont les sommets

représentent les participants au salon et les arêtes des relations professionnelles du type « travaille avec », « est partenaire commercial de » ou plus généralement « connaît ».

On considère que le graphe est dynamique car les interactions entre participants amènent à construire de nouvelles relations dans le réseau et donc de nouvelles arêtes : un participant p_1 en relation avec un autre participant p_2 peut lui recommander un troisième, p_3, conduisant à établir une arête entre les sommets représentant p_1 et p_3. Les arêtes du graphe évoluent également en fonction de l'arrivée ou du départ des participants : on considère le cadre présenté dans la section 5.2.1, p. 115, où les sommets n'évoluent pas, mais leurs relations les intègrent ou non dans le graphe.

Dans ce formalisme, la tâche d'identification d'interlocuteur visé peut alors être exprimée comme une recherche de chemin entre un sommet source, correspondant au recruteur considéré et un sommet cible, correspondant au professionnel visé. En effet, comme indiqué dans la prochaine section, on considère qu'un participant ne peut s'adresser à un autre que s'il existe une relation entre eux.

Nous utilisons le mot *chemin* pour décrire la tâche par abus de langage, le graphe étant non orienté, car il correspond mieux à l'idée que nous souhaitons exposer (une personne « parcourt » le graphe pour en trouver une autre), mais il s'agit bien d'une *chaîne* dans le langage de la théorie des graphes. Un recruteur s'adresse donc à des participants avec lesquels il est en relation pour établir de nouvelles relations et, ainsi, atteindre de proche en proche son interlocuteur visé.

La tâche que nous considérons consiste alors à fournir une stratégie de recommandation locale, permettant à leur participant de renseigner au mieux la personne qui l'interroge, pour lui permettre de rencontrer au plus vite l'interlocuteur qu'elle vise.

Notre principale proposition est de considérer la structure communautaire détectée sur le graphe comme une source d'information supplémentaire permettant de faciliter l'orientation des participants. Le calcul et la mise à jour en temps réel de cette structure sont possibles grâce à un algorithme de détection de communautés sur graphe dynamique, tel que DynLOCNeSs. De plus, DynLOCNeSs permet à chaque sommet de connaître sa communauté en temps réel et rend ainsi possible et réellement efficace l'utilisation des communautés à la volée.

7.2.3 Stratégie informatique vs stratégie humaine

Trouver un chemin entre les sommets d'un graphe est l'une des tâches les plus courantes en théorie des graphes et de nombreux travaux ont été réalisés autour de cette problématique (voir par exemple Kleinberg et Tardos (2005) pour une vue globale des approches existantes). Ils diffèrent par exemple par le critère optimisé, qui peut être la longueur du chemin ou le temps de recherche.

On pourrait a priori envisager d'appliquer un tel algorithme entre les sommets représentant le recruteur et le professionnel visé. Toutefois, une telle approche suggère qu'il existe

une entité centralisée possédant une vue globale de tout le graphe, ce qui ne correspond pas au cas d'application considéré : une méthode locale et décentralisée est nécessaire pour modéliser une phase d'exploration progressive, de proche en proche, mise en œuvre par le professionnel cherchant un emploi ou le recruteur. Un algorithme tel que l'algorithme de Dijkstra (1959) n'est ainsi par envisageable pour modéliser une stratégie d'exploration humaine.

Recherche de chemin par un humain

Un être humain a des caractéristiques et un comportement propres, par exemple il ne dispose pas d'une mémoire illimitée dans le temps ou en volume, le coût unitaire d'une action qu'il mène n'est pas le même que celui d'une machine simulant informatiquement le même type d'action.

En particulier, un être humain ne garde généralement pas en mémoire un historique des personnes les plus intéressantes trouvées jusqu'à un certain point, et n'effectue pas de *retour arrière (backtracking)*. Aussi, un professionnel ou un recruteur égaré ne pourrait pas revenir en arrière pour demander une autre recommandation à un participant dont la recommandation antérieure l'aurait fourvoyé.

De plus, sa prise de décision dans ce type de situation est plutôt simple, intuitive et met en œuvre un très petit nombre de paramètres.

Postulats sur les comportements

On ne peut donc pas supposer que la stratégie d'une personne souhaitant trouver un chemin dans le graphe des participants, conformément au but exposé plus haut, serait similaire à celle d'un algorithme. C'est pourquoi nous avons posé plusieurs contraintes au développement de notre solution, sous forme de postulats sur les comportements humains :

Postulat 1. Chaque personne connaît un nombre défini non nul d'autres personnes présentes lors de l'événement : elle possède au moins une et généralement plusieurs connaissances dès le début de l'événement.

Postulat 2. Chaque personne est présente pour en rencontrer exactement une autre lors de l'événement, qu'elle ne connaît pas préalablement.

Postulat 3. Chaque personne appartient à une ou plusieurs communautés, c'est-à-dire un groupe de personne partageant un trait commun. L'ensemble des participants au salon modélisé par le graphe décrit dans la section précédente possède une structure communautaire.

Postulat 4. Chaque personne peut présenter deux personnes de sa connaissance l'une à l'autre. Ainsi, une relation entre ces deux personnes est établie et chacune devient une connaissance de l'autre.

Postulat 5. Chaque personne n'interagit qu'avec une et seule seule personne à la fois. Chaque interaction implique donc au maximum deux personnes simultanément.

Remarque

La modélisation de la stratégie de recherche d'une personne doit être réaliste. En effet, un être humain ne peut pas chercher une personne parmi d'autres qu'elle ne connaît pas de la même façon qu'un algorithme explore un espace d'états.

Il est important de noter que cette contrainte est très stricte, et a très fortement influencé les solutions proposées.

Discussion sur d'autres effets comportementaux

Si les postulats présentés nous permettent d'effectuer des simulations informatiques de situations sans devoir prendre en compte de notions trop complexes et que nous jugeons d'un intérêt moindre dans un premier temps, par exemple la modélisation des rencontres simultanées autant sémantiquement qu'en terme d'implémentation, de programmation, de nombreux autres effets liés à une situation réelle seraient à prendre en compte.

Ainsi, nous ne considérons pas l'éventuelle attente que peut subir une personne qui souhaite en rencontrer une autre. Par exemple, si la personne à qui un recruteur souhaite parler est déjà en conversation avec une autre, le postulat 5 lui interdisant d'entrer dans la conversation, il doit attendre que cette conversation soit finie avant de parler à la personne qui l'intéresse. L'attente ainsi générée, tant au niveau temporel qu'à celui de l'ordonnancement (gérer l'ordre de la file des personnes attendant de parler à une seule et même autre déjà en conversation) n'est pas prise en compte dans les modèles que nous proposons.

Convergence entre stratégies informatique et humaine

Il existe plusieurs manières d'explorer un graphe afin de pouvoir trouver un chemin entre deux sommets (largeur d'abord, profondeur d'abord...), l'une des plus simples mais néanmoins fréquemment utilisée est la *marche aléatoire* (cf. section 2.5.2), notamment lorsque l'algorithme n'a connaissance que d'un sous-ensemble local du graphe à un instant donné, par exemple dans le cas d'Internet (bien trop vaste) ou des réseaux sociaux (on ne peut connaître que ses amis et pas les amis de ses amis) (Brin & Page, 1998, Avin et al., 2008).

Les travaux présentés dans ce chapitre sont basés sur les marches aléatoires, car elles représentent une méthode d'exploration naturelle, souvent utilisée en milieu inconnu, que ce soit par des humains dans un contexte réel ou en simulation dans les systèmes multi-agents par exemple. En particulier, la marche aléatoire est compatible avec les postulats présentés précédemment.

7.3 Recommandation communautaire : algorithme SWAGG

Nous avons conçu l'algorithme nommé SWAGG (**S**tray **W**alk **A**lgorithm for **G**uided **G**raph exploration) afin de répondre à la problématique formalisée dans la section 7.2, p. 136, tenant compte des postulats évoqués section 7.2.3 :

« Etant donné un humain portant un dispositif électronique communicant permettant d'afficher des informations visuelles changeantes, tel que le T-shirt augmenté du projet ANR Homo Textilus, comment utiliser ce dispositif afin d'améliorer l'expérience utilisateur, c'est-à-dire les rencontres effectuées ? »

Il est très important de garder à l'esprit que le but est de se rapprocher le plus possible de l'utilisation que ferait un humain, et que l'algorithme est conçu en conséquence.

Nous exposons tout d'abord l'idée générale à la base de l'algorithme que nous proposons, puis nous présentons les notations utilisées avant de décrire l'algorithme plus en détails et de conclure par différentes expériences réalisées afin de montrer son intérêt et sa validité.

7.3.1 Principes de l'algorithme SWAGG

Le principe de l'algorithme que nous proposons consiste à biaiser une marche aléatoire à l'aide d'une information ajoutée, nommée marquage, et qui correspond à un symbole qui serait affiché sur le t-shirt augmenté.

Nous ne cherchons pas à trouver un optimum global pour un critère d'efficacité particulier (taille du chemin, temps de calcul...) mais à concevoir un algorithme identifiant un chemin entre deux sommets distincts sur le graphe particulier donné en entrée en minimisant le nombre de sommets visités et en tenant compte des contraintes et postulats évoqués dans la section 7.2.

Ainsi l'algorithme SWAGG repose sur deux étapes :

1. le marquage du graphe en utilisant la structure communautaire afin de créer de l'information pour la marche aléatoire,
2. l'exploration du graphe depuis le sommet source par marche aléatoire « orientée » à l'aide de l'information créée à l'étape précédente.

7.3.2 Notations

En sus des notations présentées en début de manuscrit, nous utiliserons également les notations suivantes :

Nous considérons toujours un graphe $G = (V, E)$ non orienté. Un *marcheur* M visite l'un après l'autre des sommets adjacents de G, le sommet suivant à visiter étant aléatoirement tiré parmi les voisins du sommet précédent selon la distribution de probabilités décrite ci-dessous.

Algorithme 7 Algorithme de marquage μ

Require: Graphe $G = (V, E)$, ensemble de communautés C
Ensure: Ensemble des marquages C_μ
for all $v \in V$ **do**
  if v a des voisins dans plus d'une communauté **then**
    $\mu(v) \leftarrow$ communauté la plus fréquente parmi les voisins de v sauf $C(v)$
  else
    $\mu(v) \leftarrow C(v)$
  end if
end for
for all $v \in V$ **do**
  if $\mu(v) = \mu$ le plus fréquent parmi les voisins de v **then**
    $\mu(v) \leftarrow \mu$ le moins fréquent parmi les voisins de v
  end if
end for

Le sommet source v_d et la cible v_a, ainsi que sa communauté $C(v_a)$ étant des informations connues pour M, le problème est de déterminer le chemin S entre v_d et v_a en explorant une partie V_S des sommets V de G, soit $v_d \in S, v_a \in S, S \subset V_S \subset V$.

Les deux ensembles ordonnés S et V_S correspondent à des sous-ensembles de V contenant respectivement la séquence des sommets par lesquels M est passé, et la séquence des sommets que M a scrutés pour décider par où passer, on a donc

$$\forall s \in S, \exists s' \in S, s' \neq s / (s, s') \in E$$

Nous souhaitons que V_S contienne le moins de sommets possibles, minimisant l'égarement pour M dans son parcours de v_d à v_a.

Nous présentons successivement les deux étapes de SWAGG : le marquage du graphe, qui exploite l'information communautaire dans cette section et le parcours heuristique, une marche aléatoire modifiée pour forcer l'orientation vers certains sommets, dans la section suivante.

7.3.3 Marquage du graphe

La fonction de marquage μ, dont le pseudo-code est donné dans l'algorithme 7, est une classification non-déterministe des sommets de G ajoutant une information partielle et choisie à chacun de ces sommets déterminée à l'aide de leurs voisinages, qui permet au marcheur M de s'orienter en maximisant ses chances de se rapprocher de v_a et minimisant celles de faire un détour. Nous proposons que cette information soit la communauté la plus « représentative » du voisinage de v, comprise comme la communauté à laquelle appartient le plus grand nombre de voisins de v. En reprenant les notations de la section précédente, $\forall v \in V, \mu(v) \in [\![1, |C|]\!]$. Nous appelons C_μ l'ensemble des $\mu(v)$ calculés sur G.

Algorithme 8 Algorithme de parcours heuristique h

Require: Graphe $G = (V, E)$, C, C_μ, $v_d \in V$, $v_a \in V$
Ensure: Chemin S de sommets, $S \subset V$
  sommet_courant $\leftarrow v_d$
  $S \leftarrow \{\}$
  while sommet_courant $\neq v_a$ **do**
    voisins $\leftarrow \Gamma$(sommet_courant) - sommet_courant.predecesseur
    if $\exists v \in$ voisins$/C(u) = C(v_a)$ && $\mu(v) = C(v_a)$ **then**
      sommet_courant $\leftarrow v$
    else if $\exists v' \in$ voisins$/C(v') = C(v_a)$ **then**
      sommet_courant $\leftarrow v'$
    else if $\exists v'' \in$ voisins$/\mu(v'') = C(v_a)$ **then**
      sommet_courant $\leftarrow v''$
    else
      sommet_courant $\leftarrow$ voisins.aleatoire
    end if
    S.ajouter(sommet_courant)
  end while

L'algorithme proposé affecte dans une première passe, à chaque sommet v, un marquage $\mu(v)$ correspondant à la communauté la plus fréquente parmi celles de ses voisins. Dans une seconde passe, l'algorithme affecte à chaque sommet v le marquage le moins fréquent parmi les marquages de ses voisins.

En effet, la connexité intra-communautaire étant par définition plus forte que la connexité inter-communautaire, en effectuant seulement la première passe on risque d'établir un découpage C_μ proche de C car beaucoup de sommets voisins au sein d'une même communauté partagent en grande partie le même voisinage et héritent donc d'un marquage similaire à l'issue de la première passe.

La deuxième passe évite ce phénomène en valorisant les voisinages *atypiques*, c'est-à-dire les connexions entre deux communautés possédant peu d'arêtes entre leurs sommets respectifs : la majorité des chemins entre ces deux communautés passe par ces sommets au voisinage atypique.

Un sommet v à voisinage atypique se signale comme *interface* avec une communauté C' qui possède peu de liens avec sa propre communauté $C(v)$ et est donc plus difficile à atteindre pour une marche aléatoire. Il est donc susceptible de raccourcir le parcours de M si celui-ci le choisit pour aller vers une telle communauté C'.

Cette passe est non-déterministe car dépendante de l'ordre dans lequel les sommets sont traités pour la réaffectation de marquage.

7.3.4 Parcours heuristique

Le parcours heuristique h dont le pseudo-code est donné dans l'algorithme 8 consiste à exploiter l'information calculée par la fonction de marquage μ afin d'orienter une marche

aléatoire, qui devient donc une marche non uniforme. La marche commence au sommet v_d et se poursuit en choisissant un sommet successeur v_{i+1}, que M doit visiter ensuite, parmi les voisins du sommet v_i sur lequel il se trouve.

Une marche aléatoire classique tire uniformément v_{i+1} parmi $V_{i+1} = \Gamma(v_i)$, dont on retire généralement v_{i-1}, prédécesseur de v_i, afin d'éviter de « tourner en rond » trop vite. Nous proposons de définir à la place trois règles pour sélectionner v_{i+1} de façon plus restrictive. Partant de $V_{i+1} = \emptyset$, chaque règle est appliquée successivement jusqu'à ce que $V_{i+1} \neq \emptyset$. Nous les donnons ici dans l'ordre décroissant de leurs priorités d'exécution (la plus prioritaire en premier). V_{i+1} est l'ensemble des voisins v de v_i distincts de v_{i-1} tels que :

1. v présente une communauté $C(v)$ et un marquage $\mu(v)$ correspondant à la communauté du sommet cible, c'est-à-dire $C(v) = C(v_a)$ et $\mu(v) = C(v_a)$,
2. v présente la communauté recherchée mais pas le marquage correspondant, $C(v) = C(v_a)$ et $\mu(v) \neq C(v_a)$,
3. v présente seulement un marquage correspondant à $C(v_a)$, soit $\mu(v) = C(v_a)$, ce qui indique que ce voisin « connaît » un sommet dans $C(v_a)$ et donc que M va pouvoir ensuite se rendre rapidement dans $C(v_a)$.

Si aucun sommet ne satisfait la première règle ($V_{i+1} = \emptyset$ après son exécution) alors on sélectionne les sommets en utilisant la deuxième etc. Lorsqu'à la fin de l'exécution d'une règle on a au moins 1 sommet dans V_{i+1}, on choisit alors v_{i+1} dans V_{i+1} selon un tirage aléatoire uniforme.

Si V_{i+1} est toujours vide après application de la règle 3 alors on pose $V_{i+1} = \Gamma(v_i) \setminus \{v_{i-1}\}$, c'est-à-dire que v_{i+1} est simplement choisi aléatoirement de façon uniforme parmi tous les voisins distincts, à l'instar d'une marche aléatoire simple classique. On note ainsi qu'une exécution du parcours heuristique h sur un graphe sur lequel il n'existe pas de marquage μ revient à une marche aléatoire simple classique.

L'idée est que, pour se rapprocher du sommet cible v_a dont il connaît la communauté, la meilleure solution pour M est de tenter prioritairement de se rapprocher de la communauté $C(v_a)$. Le marquage n'est présent que pour lui donner une information dans le cas où il commencerait à « se perdre », c'est-à-dire à s'éloigner de sa cible.

7.4 Etude expérimentale

Nous avons mené diverses expériences visant à déterminer l'efficacité de SWAGG par rapport aux critères du cas d'étude considéré. Cette section présente ces expériences et leurs résultats.

L'algorithme implémenté est celui décrit ci-dessus auquel nous avons ajouté une modification interdisant de « rencontrer » les sommets qui n'ont qu'un seul voisin (sauf si ce seul voisin est la cible). Cela évite de se retrouver « coincé » inutilement sur un sommet

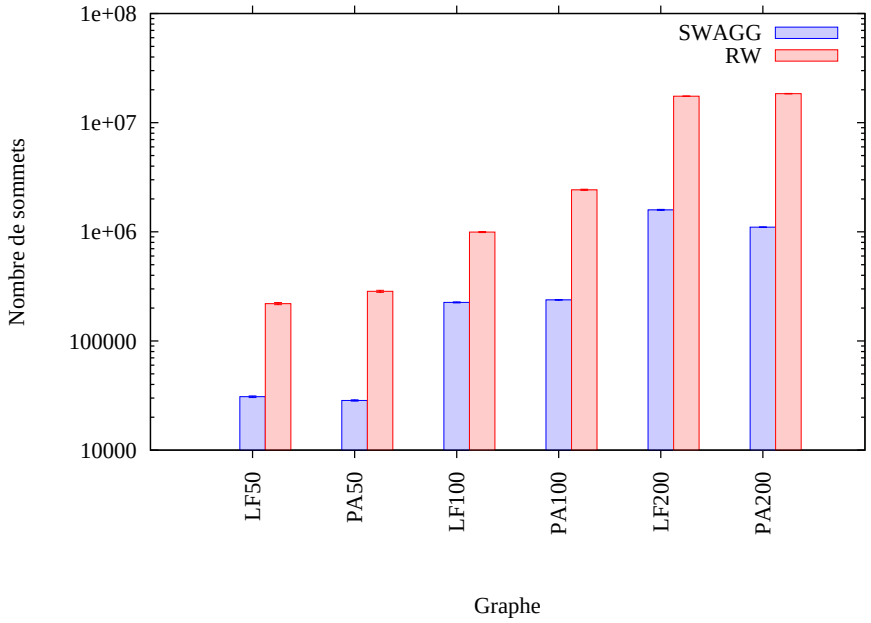


FIGURE 7.1 – Nombre total de sommets visités par SWAGG et RW, graphes LF et PA, échelle logarithmique en ordonnée

dont on sait que la seule issue (le seul successeur possible) est le sommet d'où l'on vient. Le comportement humain diffèrerait certainement ici : on ne peut pas savoir à l'avance qu'une personne n'a qu'un seul ami et l'on n'aurait pas nécessairement de raison de l'éviter. Notre but était ici uniquement d'améliorer la rapidité de notre algorithme lors des tests, sans influer trop sur les résultats de l'expérience.

Les expériences sont menées sur huit graphes différents (voir en annexe B une description des générateurs et instances de graphes utilisés) :

— trois graphes générés par LFR, notés G_{LFx} avec $x \in \{50, 100, 200\}$
— trois graphes d'attachement préférentiel générés par GraphStream : G_{PAx} pour $x \in \{50, 100, 200\}$
— deux graphes G_{test} et G_{cc}.

Nombre de sommets visités par rapport à une marche aléatoire

Le premier critère considéré est le nombre de sommets visités, assimilable au nombre de rencontres que ferait le marcheur M. Avec les notations de la section 7.3.2, p. 140, ce critère s'écrit $|V_S|$.

Nous comparons l'algorithme SWAGG et une marche aléatoire simple pour différents critères détaillés ci-dessous. Pour chaque graphe, nous appliquons les algorithmes pour chaque couple source-cible de sommets non voisins $(v_d, v_a) \in V^2$ tel que $(v_d, v_a) \notin E$. La figure 7.1 présente les résultats, en échelle logarithmique en ordonnée, constitués par la moyenne et l'écart-type calculés sur tous les couples.

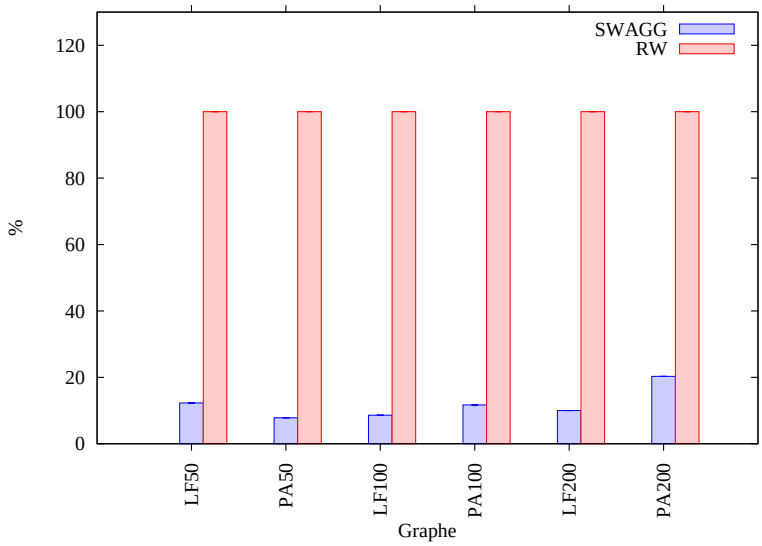


FIGURE 7.2 – Pourcentage de choix aléatoires effectué par SWAGG et RW par rapport au nombre de choix total

On observe que, pour le graphe considéré, SWAGG permet de bien mieux s'orienter par rapport à une marche aléatoire classique, avec par exemple 86% de sommets visités en moins sur G_{LF50}, et jusqu'à 94% en moins sur G_{PA200}.

On note également que les écarts-types indiquent des variabilités négligeables.

Nombre de sommets choisis aléatoirement

Nous évaluons ici la proportion de choix aléatoires effectués par l'algorithme SWAGG, c'est-à-dire le rapport du nombre de sommets sélectionnés par la règle par défaut, en cas d'échec des trois premières (cf. section 7.3.4, p. 142), sur le nombre total de sommets sélectionnés. Cette proportion est évidemment toujours de 100% pour la marche aléatoire simple classique.

Les résultats obtenus sont présentés figure 7.2. On remarque que SWAGG utilise effectivement l'information apportée par la structure communautaire dans l'essentiel des cas et qu'un choix aléatoire n'est réalisé que dans 20% des cas au plus.

Influence de la taille et du nombre des communautés

Nous avons répété la première expérience mesurant le nombre de sommets visités sur le graphe G_{cc}, une chaîne de 10 3-cliques (voir annexe B).

Les résultats obtenus, comparés à G_{LF50} et G_{PA50}, sont présentés figure 7.3 (en échelle logarithmique).

On remarque que SWAGG est moins efficace sur le graphe G_{cc}, en terme de ratio de nombre de sommets visités comparé à G_{LF50} et G_{PA50}, alors qu'il comprend moins

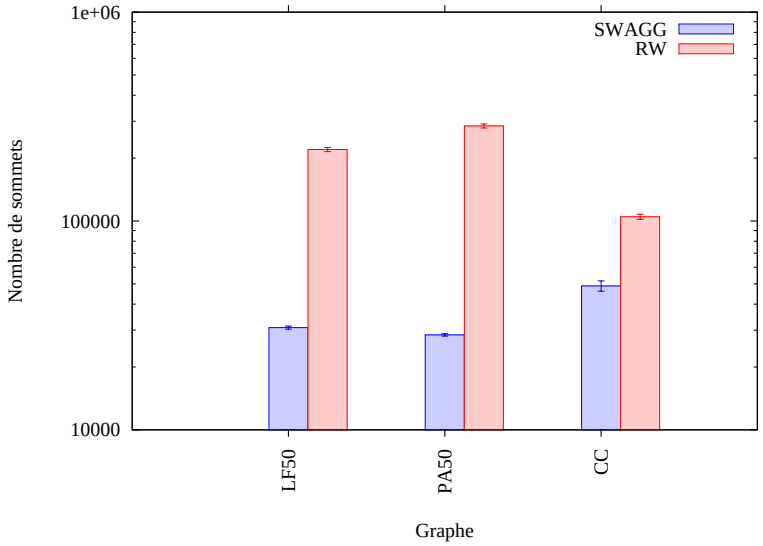


FIGURE 7.3 – Nombre total de sommets visités par SWAGG et RW, petits graphes, échelle logarithmique en ordonnée

de sommets. L'écart-type du nombre de sommets visités par SWAGG est plus important également.

Nous expliquons cela par le fait que le graphe G_{cc} est beaucoup moins « évident » à parcourir pour SWAGG, sa disposition cyclique fait qu'il possède un diamètre important et la taille des chemins source-cible est grande en moyenne par rapport au nombre de sommets dans le graphe : si on tire deux sommets au hasard dans G_{LF50}, la taille du chemin entre ces deux sommets a une plus grande probabilité d'être faible que si l'on tire deux sommets dans G_{cc}.

Ainsi, le parcours heuristique a plus de chance de se perdre dans G_{cc} et de faire augmenter le nombre de sommets visités. On remarque néanmoins que SWAGG conserve de bien meilleures performances sur cette métrique que la marche aléatoire classique : environ 40 000 sommets visités pour SWAGG contre environ 100 000 pour cette dernière.

Longueurs de chemins par rapport à l'optimal

Nous considérons un autre cas de référence pour évaluer SWAGG en comparant la taille du chemin qu'il évalue à celle d'un chemin optimal, tel que formé par l'algorithme de Dijkstra (1959). Nous rappelons que, comme indiqué section 7.2.3, p. 137, cet algorithme n'est pas un compétiteur car il ne respecte pas les contraintes de réalisme fixées, il est utilisé ici uniquement comme critère d'évaluation.

Pour chaque couple de sommets non voisins, on calcule le rapport

$$\frac{|S_{Dijkstra}|}{|S_{SWAGG}|}$$

Le rapport est égal à 1 si le chemin trouvé par SWAGG est le plus court, car l'algorithme de Dijkstra garantit de trouver le chemin le plus court, il décroît lorsque la taille du chemin trouvé par SWAGG augmente.

Ces rapports sont ensuite additionnés, et l'on cherche donc à obtenir une somme de valeur maximale, la valeur maximum étant atteinte si tous les ratios valent 1 soit le nombre de paires de sommets non-voisins, qui vaut 14 208 pour G_{test}.

On évalue les moyennes et écarts-types de ces sommes, en répétant 50 fois cette expérience sur le graphe G_{test}. Nous avons obtenu un score moyen de **1329**, avec un écart-type de **19**.

Sur l'ensemble du graphe G_{test}, on peut donc déduire que les chemins trouvés sont environ 10 fois plus longs, en moyenne, que les chemins trouvés par Dijkstra. Ce nombre peut paraître élevé mais souffre de l'effet d'agrégation de la moyenne : si l'on regarde dans le détail il y a un certain nombre de ratios valant 0.5 (chemin de SWAGG deux fois plus long donc), quand certains valent 0.01 environ, les cas où SWAGG s'égare et fait appel à un tirage complètement aléatoire du successeur par exemple, impactant de ce fait lourdement la moyenne. Cependant, l'écart-type faible laisse penser que ces cas sont peu fréquents.

De plus, la plupart des chemins étant relativement courts du fait du diamètre du graphe et de l'effet petit-monde, SWAGG peut trouver des chemins dont la différence de longueur à l'optimale en valeur absolue est de 1 ou 2, diminuant pourtant de beaucoup la somme. Par exemple, un chemin de longueur 5 trouvé par SWAGG pour une longueur optimale de 3, produit un ratio de $3/5 = 0.6$. L'impact sur la somme est donc fort mais le chemin retourné n'est pas nécessairement non pertinent : la taille moyenne des chemins induit un léger biais de perception sur l'efficacité réelle de SWAGG. Il est alors nécessaire de déterminer, par rapport au contexte, si l'on considère qu'une personne en rencontrant 5 autres a perdu trop de temps par rapport à une personne en rencontrant 3 pour arriver à la même cible.

7.5 Bilan

A partir du cas d'étude proposé par le projet ANR Homo Textilus (section 7.1), un événement semblable à un salon de type job dating dans lequel chaque participant porte un dispositif mobile interactif et communicant, nous avons identifié une problématique particulière : trouver une personne cible préalablement inconnue mais dont on connaît la communauté.

Après avoir formalisé la tâche considérée (section 7.2, nous avons proposé un scénario permettant d'exploiter le dispositif communicant dans le contexte du cas d'étude en utilisant une stratégie par recommandation basée sur de l'information provenant de la structure communautaire, ajoutée sur le graphe à l'aide d'une fonction de marquage, afin d'orienter chaque participant vers sa cible. Nous avons élaboré un algorithme, SWAGG, et mené dif-

férentes expériences permettant de valider l'apport du scénario proposé par rapport à un scénario classique dans lequel les participants n'ont pas de stratégie d'orientation, assimilé ici à une marche aléatoire. Nous avons constaté que SWAGG permet de visiter en moyenne un nombre de sommets très inférieur à la marche aléatoire, tout en gardant une longueur de chemin type acceptable.

Conclusion et perspectives

Nous concluons cette thèse en synthétisant les différentes contributions apportées ainsi que les perspectives qu'elles ouvrent.

Synthèse des contributions

Dans le cadre général et très actif de la détection de communautés dans les graphes, nous nous sommes intéressé à trois variantes de cette tâche, dans le cas de référence des communautés disjointes dans des graphes statiques et dans deux extensions concernant respectivement les communautés disjointes et les graphes dynamiques. Nous avons considéré ces problématiques dans le contexte plus spécifique où ces graphes formalisent des réseaux décentralisés d'objets mobiles connectés, qui imposent des contraintes fortes et nécessitent de relever deux défis principaux : les méthodes proposées doivent être décentralisables et limiter la propagation de données dans le réseau.

Nous avons, pour chacune des trois variantes de la tâche de détection de communautés, proposé des états de l'art des méthodes existantes de la littérature, en nous intéressant en particulier au domaine récent en pleine expansion des méthodes dites orientées-sommet qui mettent en œuvre le paradigme TLAV particulièrement approprié aux contraintes spécifiques proposées.

Nous avons proposé un principe global de fonctionnement original obéissant à l'approche appelée leader-follower dans lequel certains sommets endossent un rôle particulier, de leader, à qui sont associés les autres sommets, dits followers, qui permettent de former les communautés après agrégation. L'une des spécificités essentielles du principe que nous proposons est d'autoriser chaque sommet à être à la fois leader *et* follower. Nous avons proposé dans ce but une mesure originale de préférence entre sommets, selon les principes d'homophilie et d'attachement préférentiel constatés dans les réseaux sociaux, l'agreement.

Nous avons ensuite décliné ces principes généraux pour les trois variantes de la tâche de détection que nous avons considérées, en proposant pour chacun un algorithme ainsi qu'une implémentation décentralisée satisfaisant les contraintes des réseaux spécifiques auxquels nous nous sommes intéressé : VOLCAN permet d'identifier les communautés disjointes dans un graphe statique, LOCNeSs l'étend pour permettre la détection de communautés recouvrantes, dans lesquelles un sommet peut appartenir à plusieurs communautés, néces-

sitant en particulier une nouvelle définition des sommets leaders candidats, c'est-à-dire des sommets préférés, ainsi que de l'étape d'agrégation. DynLOCNeSs généralise encore l'approche proposée afin de permettre le traitement de graphes dynamiques, dont les sommets et les arêtes évoluent au cours du temps.

Nous avons étudié expérimentalement chacun de ces algorithmes sur les benchmarks classiques de données artificielles et de petits graphes réels, en les comparant aux méthodes de l'état de l'art et en établissant empiriquement la pertinence de nos propositions.

Enfin nous nous sommes intéressé à une application dans le cadre du projet de recherche ANR Homo Textilus sur les vêtements intelligents qui conduisent à la formation d'un réseau mobile ad-hoc opportuniste, spontané et décentralisé. Nous avons formalisé la tâche de l'identification d'interlocuteur comme une recherche de chemin par recommandation locale contrainte par la prise en compte de stratégies humaines et avons proposé un algorithme original, SWAGG, qui exploite la structure communautaire du graphe pour biaiser efficacement une marche aléatoire : les expérimentations montrent que SWAGG permet de réduire l'exploration du graphe et constitue un premier pas vers un parcours de graphe stratégique utilisant des informations sur les communautés pour s'orienter : il pourrait être appliqué dans un contexte de routage de trafic réseau, de recherche d'information ou de modélisation.

Perspectives

Les perspectives ouvertes par nos travaux s'articulent principalement autour de deux axes, qui concernent respectivement l'approfondissement de la caractérisation, expérimentale et théorique, des principes de fonctionnement et algorithmes que nous avons proposés d'une part et, d'autre part, leur généralisation à d'autres formes de détection de communautés dans les graphes, à la fois concernant le type de communautés et le type de graphes.

Approfondissement de la caractérisation

Comme nous l'avons souligné à plusieurs reprises, les méthodes de détection de communautés dans les graphes peuvent être organisées en deux grandes familles du fait de la définition imprécise et non consensuelle de cette tâche : la première repose sur une définition a priori, le plus souvent topologique, de la notion de communauté et regroupe des méthodes qui optimisent l'extraction de ces structures dans les graphes. La seconde se base sur une approche plus procédurale et définit des méthodes dont les structures extraites dont l'objet d'une caractérisation a posteriori. C'est parmi ces dernières que figurent les approches orientées-sommet et les algorithmes que nous avons proposés.

Caractérisation expérimentale topologique

Nous envisageons dans un premier temps de développer cette caractérisation des méthodes que nous avons développées dans cette thèse, en particulier pour étudier plus préci-

sément la forme (taille, répartition des degrés, valeurs de conductance...) et la composition (densité, pureté...) des communautés identifiées par VOLCAN et LOCNeSs. Dans le cas dynamique, pour DynLOCNeSs, cette problématique vise à évaluer le comportement de la méthode sur un plus grand nombre de types d'évolution de communautés, dans un contexte plus riche que la *planted bissection*.

L'une des problématiques associées est alors la définition de graphes de test appropriés, nous envisageons par exemple d'utiliser le benchmark de Largeron et al. (2015) ainsi que, pour le cas dynamique, des graphes réels tel que celui tiré d'une rando-roller capturée dans le cadre du projet RollerNet (Tournoux et al., 2011).

Caractérisation computationnelle et implémentation parallèle

A côté de la caractérisation topologique des résultats fournis par les algorithmes, il est également crucial, dans le domaine du traitement des graphes, d'évaluer les temps de calcul et capacité de passage à l'échelle. Nous avons constaté que notre implémentation primitive de LOCNeSs traite un graphe ayant de l'ordre d'un million de sommets généré par le benchmark LFR (Lancichinetti et al., 2008) de degré moyen 4 en moins de 20 minutes.

Les algorithmes que nous avons proposés étant totalement décentralisables, nous avons veillé à proposer une algorithmique adaptée, nous souhaitons en réaliser une implémentation dans les frameworks Giraph ou Pregel (Han et al., 2014a). Elle permettra d'étudier, en particulier, la question du passage à l'échelle et le traitement de graphes de très grandes tailles et d'appliquer les algorithmes que nous avons proposés à des réseaux réels.

Evaluation écologique

Nous souhaitons également caractériser nos approches dans un autre contexte expérimental, en situation « écologique », c'est-à-dire en recréant le contexte d'exploitation envisagé : il s'agit plus précisément d'exploiter des communautés détectées en temps réel par DynLOCNeSs par exemple pour router le trafic dans un réseau de téléphones portables.

Une telle mise en œuvre permet de s'affranchir de la question de la définition des critères de qualité des communautés identifiées, en envisageant une évaluation globale et applicative. Une telle mise en situation soulève soulève de nombreux défis, parmi lesquels le développement d'une implémentation sous forme d'application mobile et l'organisation d'une expérimentation réelle.

Caractérisation théorique

En parallèle de la démarche expérimentale décrite dans les paragraphes précédents, il est important de s'intéresser à une approche théorique de caractérisation des résultats de l'algorithme, à la fois en terme de complexité calculatoire (temps et espace) et topologique. Nous souhaitons également la mettre en œuvre pour les principes de fonctionnement et algorithmes proposés. En particulier, nous souhaitons établir des résultats sur les liens entre la mesure de préférence proposée (agreement) et la conductance des ensembles de sommets identifiés comme communautés, conformément aux résultats observés expérimentalement.

Autres communautés, autres graphes

Les travaux que nous avons réalisés ouvrent également des perspectives concernant la mise en œuvre des principes que nous avons proposés pour d'autres formes de détection de communautés, plus précisément pour l'identification de communautés enrichies et pour le traitement de graphes plus riches.

Communautés avec multi-appartenances floues

LOCNeSs permet d'identifier des communautés recouvrantes, c'est-à-dire dont les intersections sont non vides. Les multi-appartenances qu'il définit sont binaires, au sens où un sommet appartient ou n'appartient pas à chacun des communautés de l'ensemble des communautés. Il nous paraît intéressant d'étendre cette définition à des multi-appartenances floues, où un sommet peut appartenir avec un degré compris en 0 et 1 à chaque communauté. Cette problématique a été investiguée par Gregory (2011) mais hors du contexte de la théorie des sous-ensembles flous (Zadeh, 1965). La mise en œuvre de techniques de clustering évidentiel (Antoine et al., 2014) pour la détection de communautés (Zhou et al., 2015) constitue également une piste prometteuse dans ce cadre.

Graphes attribués

Les graphes attribués sont des graphes dans lesquels chaque sommet, ou chaque arête, porte une information supplémentaire, par exemple sous forme catégorielle, numérique ou textuelle (Zhou et al., 2009b, Bothorel et al., 2015). Ce paradigme très riche est particulièrement adapté aux réseaux sociaux, où la proximité des utilisateurs se définit par des liens de natures diverses (intérêts/goûts, relationnels, géographiques...) mais pose un certain nombre de défis, en particulier concernant les compromis à effectuer entre le regroupement de sommets sur critères structurels et sur critères sémantiques. Cette complexité pose une difficulté toute particulière pour l'adaptation des approches que nous avons proposées, locales et décentralisées, car la comparaison de données nécessite leur échange entre sommets, générant du trafic réseau que l'on souhaite pourtant limiter.

Liens faibles

Enfin, si les méthodes que nous avons proposées sont basées sur la formation du graphe considéré par homophilie et notamment par attachement préférentiel, valorisant donc particulièrement les sommets de fort degré, nous souhaitons étudier plus précisément les situations ne correspondant pas à ce modèle. Par exemple, comme nous l'avons exposé dans l'état de l'art, l'homophilie n'est pas universellement présente, sa prévalence étant particulièrement mise en doute dans les réseaux sociaux (Shalizi & Thomas, 2011,Han et al., 2014b) : l'étude des réseaux sociaux a également montré la force des « liens faibles » (Granovetter, 1973) qu'ils comportent, c'est-à-dire des relations entre utilisateurs de zones distantes du graphe social, partageant peu de relations ou d'intérêts. Ces liens sont donc difficiles à détecter ou à anticiper dans le cas des graphes dynamiques (Centola & Macy, 2007), pourtant ils sont d'une grande importance car ils constituent des « ponts » entre communautés

(De Meo et al., 2014), dans notre cas des arêtes inter-communautaires. Nous pensons que la prise en compte de ces phénomènes permettrait d'enrichir les méthodes proposées, spécifiquement en vue d'analyser de grands réseaux sociaux.

Bibliographie

Adamic, L. A. & Adar, E. (2003). Friends and neighbors on the Web. *Social Networks*, *25*, 211–230.

Albert, R. & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*, 47–97.

Aldecoa, R. & Marín, I. (2011). Deciphering Network Community Structure by Surprise. *PLoS ONE*, *6*, e24195.

Almeida, H., Guedes, D., Meira, W. & Zaki, M. J. (2011). Is There a Best Quality Metric for Graph Clusters ? *Proc. Eur. Conf. Machine Learning & Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD'11), LNCS, vol. 6911* (pp. 44–59). Springer, Berlin, Heidelberg.

Aloi, G., Felice, M. D., Loscrì, V., Pace, P. & Ruggeri, G. (2014). Spontaneous smartphone networks as a user-centric solution for the future internet. *IEEE Commun. Magazine*, *52*, 26–33.

Alon, U. (2003). Biological networks : the tinkerer as an engineer. *Science*, *301*, 1866–1867.

Amaral, L. a. N., Scala, A., Barthélémy, M. & Stanley, H. E. (2000). Classes of small-world networks. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *97*, 11149–11152.

Amelio, A. & Pizzuti, C. (2015). Is normalized mutual information a fair measure for comparing community detection methods ? *Proc. 2015 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'15)* (pp. 1584–1585).

Andersen, R., Chung, F. & Lang, K. (2006). Local Graph Partitioning using PageRank Vectors. *Proc. 47th IEEE Symp. on Foundations of Computer Science (FOCS'06)* (pp. 475–486).

Andersen, R. & Lang, K. J. (2006). Communities from Seed Sets. *Proc. 15th Int. Conf. on World Wide Web (WWW'06)* (pp. 223–232). ACM.

Antoine, V. & Labroche, N. (2015). Classification évidentielle avec contraintes d'étiquettes. *Proc. 2015 French. Conf. Extraction Gestion de Connaissances (EGC'15) 2015.*

Antoine, V., Labroche, N. & Vu, V. V. (2014). Evidential seed-based semi-supervised clustering. *Proc. Joint 7th Int. Conf. on Soft Computing and Intelligent Systems (SCIS)'14 and 15th Int. Symp. on Advanced Intelligent Systems (ISIS'14)* (pp. 706–711).

Asratian, A. S., Denley, T. M. J. & Häggkvist, R. (1998). *Bipartite Graphs and Their Applications*. Cambridge University Press.

Auber, D., Chiricota, Y., Delest, M., Melançon, G., Domenger, J.-P. & Mary, P. (2007). Visualisation de graphes avec Tulip : exploration interactive de grandes masses de données en appui à la fouille de données et à l'extraction de connaissances. *Proc. 2007 French Conf. Extraction et Gestion de Connaissances (EGC'07)* (pp. 147–156). Cépaduès.

Avin, C., Koucký, M. & Lotker, Z. (2008). How to Explore a Fast-Changing World (Cover Time of a Simple Random Walk on Evolving Graphs). In *Automata, Languages and Programming*, LNCS, vol. 5125, 121–132. Springer Berlin Heidelberg.

Aynaud, T. (2011). *Détection de communautés dans les réseaux dynamiques*. Thèse de doctorat, Université Pierre et Marie Curie, Paris, France.

Aynaud, T., Fleury, E., Guillaume, J.-L. & Wang, Q. (2013). Communities in Evolving Networks : Definitions, Detection, and Analysis Techniques. In *Dynamics on and of Complex Networks, Volume 2*, Modeling and Simulation in Science, Engineering and Technology, 159–200. Springer New York.

Aynaud, T. & Guillaume, J.-L. (2010). Static community detection algorithms for evolving networks. *Proc. 8th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)* (pp. 513–519).

Backstrom, L., Boldi, P., Rosa, M., Ugander, J. & Vigna, S. (2012). Four Degrees of Separation. *Proc. 4th ACM Web Science Conf.* (pp. 33–42). ACM.

Bagrow, J. P. & Bollt, E. M. (2005). Local method for detecting communities. *Phys. Rev. E*, *72*, 046108.

Barabási, A.-L. (2016). *Network science*. Cambridge University Press.

Barabási, A.-L. & Albert, R. (1999). Emergence of Scaling in Random Networks. *Science*, *286*, 509–512.

Barabási, A.-L., Jeong, H., Néda, Z., Ravasz, E., Schubert, A. & Vicsek, T. (2002). Evolution of the social network of scientific collaborations. *Physica A : Statistical mechanics and its applications*, *311*, 590–614.

Barabási, A.-L. & Stanley, H. E. (1995). *Fractal concepts in surface growth*. Cambridge University Press.

Basu, S., Bilenko, M. & Mooney, R. J. (2004). A probabilistic framework for semi-supervised clustering. *Proc. 10th Int. Conf. ACM SIGKDD on Knowledge Discovery and Data mining (KDD'04)* (pp. 59–68). ACM.

Bauman, E. & Bauman, K. (2017). Discovering the Graph Structure in the Clustering Results. *arXiv :1705.06753 [cs, stat]*.

Bedi, P. & Sharma, C. (2016). Community detection in social networks. *WIREs Data Mining Knowledge Discovery*, *6*, 115–135.

Ben N'Cir, C.-E., Cleuziou, G. & Essoussi, N. (2015). Overview of Overlapping Partitional Clustering Methods. In *Partitional Clustering Algorithms*, 245–275. Springer, Cham.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers.

Bland, A. (2014). FireChat – the messaging app that's powering the Hong Kong protests. *The Guardian.*

Blondel, V. D., Guillaume, J.-L., Hendrickx, J. M., de Kerchove, C. & Lambiotte, R. (2008a). Local leaders in random networks. *Phys. Rev. E*, *77*, 036114.

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

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M. & Hwang, D. U. (2006). Complex networks : Structure and dynamics. *Physics Reports*, *424*, 175–308.

Bollobás, B. (1988). The Isoperimetric Number of Random Regular Graphs. *Eur. J. of Combinatorics*, *9*, 241–244.

Bollobás, B. (1998). *Modern Graph Theory.* Springer Science & Business Media.

Bollobás, B. (2001). *Random graphs.* Cambridge Studies in Advanced Mathematics, vol 73. 2nd edition. Cambridge University Press.

Bornholdt, S. & Schuster, H. G. (2006). *Handbook of graphs and networks : from the genome to the internet.* John Wiley & Sons.

Bothorel, C., David Cruz, J., Magnani, M. & Micenková, B. (2015). Clustering attributed graphs : Models, measures and methods. *Network Science*, *3*, 408–444.

Brach, P., Cygan, M., Lącki, J. & Sankowski, P. (2016). Algorithmic Complexity of Power Law Networks. *Proc. 27th ACM-SIAM Symposium on Discrete Algorithms* (pp. 1306–1325). SIAM.

Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z. & Wagner, D. (2007). On finding graph clusterings with maximum modularity. *Proc. 33th Int. Workshop on Graph-Theoretic Concepts in Computer Science* (pp. 121–132). Springer.

Brin, S. & Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. Brisbane, Australia. *Proc. 7th Int. Conf. on World Wide Web (WWW'98)* (pp. 107–117). Elsevier Science Publishers. B. V. Amsterdam

Bron, C. & Kerbosch, J. (1973). Algorithm 457 : finding all cliques of an undirected graph. *Commun. ACM*, *16*, 575–577.

Buluç, A., Meyerhenke, H., Safro, I., Sanders, P. & Schulz, C. (2016). Recent Advances in Graph Partitioning. In *Algorithm Engineering*, LNCS, vol. 9220, 117–158. Springer International Publishing.

Canu, M., Detyniecki, M. & Lesot, M.-J. (2014). Utilisation de la structure communautaire pour guider une marche aléatoire. *Proc. 5th French Conf. Modèles et Analyse des Réseaux : Approches Mathématiques et Informatiques (MARAMI'14).*

Canu, M., Detyniecki, M., Lesot, M.-J. & Revault d'Allonnes, A. (2015). Fast community structure local uncovering by independent vertex-centred process. *Proc. 2015 IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM'15)* (pp. 823–830).

Canu, M., Lesot, M.-J. & Revault d'Allonnes, A. (2016a). Vertex-centred Method to Detect Communities in Evolving Networks *Complex Networks and their Applications V* (pp. 275–286). Springer International Publishing.

Canu, M., Lesot, M.-J. & Revault d'Allonnes, A. (2016b). Détection de communautés recouvrantes orientée sommet. *Proc. 7th French Conf. Modèles et Analyse des Réseaux : Approches Mathématiques et Informatiques (MARAMI'16).*

Canu, M., Lesot, M.-J. & Revault d'Allonnes, A. (2016c). Overlapping Community Detection by Local Decentralised Vertex-Centred Process. *Proc. IEEE 16th Int. Conf. on Data Mining Workshops (ICDMW'16)* (pp. 77–84).

Cao, Y. & Sun, Z. (2013). Routing in Delay/Disruption Tolerant Networks : A Taxonomy, Survey and Challenges. *IEEE Commun. Surveys Tutorials*, *15*, 654–677.

Carrier, K. (2017). National Longitudinal Study of Adolescent to Adult Health. Bureau of Labor Statistics, USA.

Casteigts, A., Flocchini, P., Quattrociocchi, W. & Santoro, N. (2011). Time-Varying Graphs and Dynamic Networks. In *Ad-hoc, Mobile, and Wireless Networks*, LNCS, vol. 6811, 346–359. Springer Berlin Heidelberg.

Cazabet, R. & Amblard, F. (2011). Simulate to Detect : A Multi-agent System for Community Detection. *Proc. 2011 IEEE/WIC/ACM Web Intelligence and Intelligent Agent Technology (WI-IAT'11)* (pp. 402–408).

Centola, D. & Macy, M. (2007). Complex Contagions and the Weakness of Long Ties. *Amer. J. Sociology*, *113*, 702–734.

Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R. & Scott, J. (2005). *Pocket Switched Networks : Real-world mobility and its consequences for opportunistic forwarding* (Technical Report UCAM-CL-TR-617). University of Cambridge, Department of Computer Science and Technology.

Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R. & Scott, J. (2007). Impact of Human Mobility on Opportunistic Forwarding Algorithms. *IEEE Trans. on Mobile Computing*, *6*, 606–620.

Chakrabarti, D. & Faloutsos, C. (2006). Graph Mining : Laws, Generators, and Algorithms. *ACM Comput. Surv.*, *38*.

Chakrabarti, D., Kumar, R. & Tomkins, A. (2006). Evolutionary Clustering. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'06)* (pp. 554–560). ACM.

Chan, S.-Y., Hui, P. & Xu, K. (2009). Community Detection of Time-Varying Mobile Social Networks. In *Complex Sciences*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 4, 1154–1159. Springer Berlin Heidelberg.

Cheeger, J. (1969). A lower bound for the smallest eigenvalue of the Laplacian.

Chen, J., Zaïane, O. & Goebel, R. (2009). Local community identification in social networks. *Proc. 2009 Int. Conf. Advances in Social Network Analysis and Mining (ASONAM'09)* (pp. 237–242). IEEE.

Chen, W., Liu, Z., Sun, X. & Wang, Y. (2010). A game-theoretic framework to identify overlapping communities in social networks. *Data Mining Knowledge Discovery*, *21(2)*, 224–240. Springer.

Chowdhury, D., Santen, L. & Schadschneider, A. (2000). Statistical physics of vehicular traffic and some related systems. *Physics Reports*, *329*, 199–329.

Clauset, A. (2005). Finding local community structure in networks. *Phys. Rev. E*, *72*, 026132.

Clauset, A., Shalizi, C. R. & Newman, M. E. J. (2009). Power-law distributions in empirical data. *SIAM Rev.*, *51*, 661–703.

Clementi, A., Di Ianni, M., Gambosi, G., Natale, E. & Silvestri, R. (2015). Distributed community detection in dynamic graphs. *Theoretical Computer Science, 584*, 19-41. Elsevier B.V. Amsterdam.

Cleuziou, G. (2008). An extended version of the k-means method for overlapping clustering. *Proc. 19th Int. Conf. Pattern Recognition (ICPR'08)* (pp. 1–4). IEEE.

Cohen, S., Kimelfeld, B. & Koutrika, G. (2012). A Survey on Proximity Measures for Social Networks. In *Search Computing*, LNCS, vol. 7538, 191–206. Springer Berlin Heidelberg.

Cohen, Y., Hendler, D. & Rubin, A. (2016). Node-centric detection of overlapping communities in social networks. *Proc. 2016 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'16)* (pp. 1384–1385). IEEE.

Colizza, V., Flammini, A., Serrano, M. A. & Vespignani, A. (2006). Detecting rich-club ordering in complex networks. *Nat. Phys.*, *2*, 110–115.

Conti, M. & Giordano, S. (2014). Mobile ad hoc networking : milestones, challenges, and new research directions. *IEEE Commun. Magazine*, *52*, 85–96.

Cordasco, G. & Gargano, L. (2012). Label propagation algorithm : a semi-synchronous approach. *Int. J. Social Network Mining*, *1*, 3–26.

Costa, L. d. F., Jr, O. N. O., Travieso, G., Rodrigues, F. A., Boas, P. R. V., Antiqueira, L., Viana, M. P. & Rocha, L. E. C. (2011). Analyzing and modeling real-world phenomena with complex networks : a survey of applications. *Advances in Physics*, *60*, 329–412.

Costa, P., Mascolo, C., Musolesi, M. & Picco, G. (2008). Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, *26*, 748–760.

Cournier, A. & Habib, M. (1994). A new linear algorithm for modular decomposition. *Trees in Algebra and Programming—CAAP'94, 19th Int. Colloquium*, 68–84.

Daly, E. M. & Haahr, M. (2007). Social Network Analysis for Routing in Disconnected Delay-tolerant MANETs. *Proc. 8th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing* (pp. 32–40). ACM.

Danisch, M. (2015). *Mesures de proximité appliquées à la détection de communautés dans les grands graphes de terrain.* Thèse de Doctorat. Université Pierre et Marie Curie.

Danisch, M., Guillaume, J.-L. & Grand, B. L. (2013a). Une approche à base de proximité pour la détection de communautés egocentrées *Proc. 4th French Conf. Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'13)* (pp. 1–4).

Danisch, M., Guillaume, J.-L. & Le Grand, B. (2013b). Towards multi-ego-centred communities : a node similarity approach. *Int. J. of Web Based Communities*, *9*, 299–322.

Danon, L., Díaz-Guilera, A., Duch, J. & Arenas, A. (2005). Comparing community structure identification. *J. Stat. Mech.*, *2005*, P09008.

De Meo, P., Ferrara, E., Fiumara, G. & Provetti, A. (2014). On Facebook, Most Ties Are Weak. *Commun. ACM*, *57*, 78–84.

Derényi, I., Palla, G. & Vicsek, T. (2005). Clique Percolation in Random Networks. *Phys. Rev. Lett.*, *94*, 160202.

Desmier, E. (2014). *Co-evolution pattern mining in dynamic attributed graphs.* Thèse de doctorat. INSA Lyon.

Dhillon, I. S., Guan, Y. & Kulis, B. (2007). Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *29*.

Di Pietro, R. & Domingo-Ferrer, J. (2013). Security in Wireless Ad Hoc Networks. In *Mobile Ad Hoc Networking*, 106–153. John Wiley & Sons, Inc.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numer. Math.*, *1*, 269–271.

Dor, D. & Tarsi, M. (1997). Graph Decomposition is NP-Complete : A Complete Proof of Holyer's Conjecture. *SIAM J. Comput.*, *26*, 1166–1187.

Doreian, P., Batagelj, V. & Ferligoj, A. (2005). *Generalized blockmodeling.* No. 25 in Structural Analysis in the Social Sciences. Cambridge University Press.

Dorogovtsev, S. N., Goltsev, A. V. & Mendes, J. F. F. (2006). K-core organization of complex networks. *Physical Rev. Lett.*, *96*, 040601.

Dorogovtsev, S. N. & Mendes, J. F. F. (2002). Evolution of networks. *Advances in Physics*, *51*, 1079–1187.

Dutot, A., Guin, F., Olivier, D. & Pigné, Y. (2007). Graphstream : A tool for bridging the gap between complex systems and dynamic graphs. *Proc. 2007 Workshop on Emergent Properties in Natural and Artificial Complex Systems (EPNACS'07), Eur. Conf. Complex Systems (ECCS'07)*.

Dörfler, F. & Bullo, F. (2014). Synchronization in complex networks of phase oscillators : A survey. *Automatica*, *50*, 1539–1564.

Erdös, P. & Rényi, A. (1959). On random graphs, I. *Publicationes Mathematicae (Debrecen)*, *6*, 290–297.

Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, *8*, 128–140.

Evans, T. S. (2010). Clique graphs and overlapping communities. *J. Stat. Mech.*, *2010*, P12037.

Everitt, B. S. (1984). *An Introduction to Latent Variable Models*. Dordrecht : Springer Netherlands.

Faloutsos, M., Faloutsos, P. & Faloutsos, C. (1999). On Power-law Relationships of the Internet Topology. *Proc. Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication* (pp. 251–262). ACM.

Ford, L. R. (1956). *Network flow theory*. Rand Corp.

Fortunato, S. (2009). Community detection in graphs. *Phys. Rep.*, 75–174.

Fortunato, S. & Barthélemy, M. (2007). Resolution limit in community detection. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *104*, 36–41.

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

Foss, S., Korshunov, D. & Zachary, S. (2011). *An Introduction to Heavy-Tailed and Subexponential Distributions*. Springer Science & Business Media.

Freeman, L. (1977). A Set of Measures of Centrality Based on Betweenness. *Sociometry*, *40*, 35–41.

Freeman, L. C. (1982). Centered graphs and the structure of ego networks. *Mathematical Social Sciences*, *3*, 291–304.

Gallai, T. (1967). Transitiv orientierbare Graphen. *Acta Mathematica Hungarica*, *18*, 25–66.

Gao, W., Luo, W. & Bu, C. (2016). Evolutionary community discovery in dynamic networks based on leader nodes. *Proc. 2016 Int. Conf. on Big Data and Smart Computing (BigComp'16)* (pp. 53–60).

Gaumont, N. (2016). *Groupes et Communautés dans les flots de liens : des données aux algorithmes*. Thèse de doctorat. Université Pierre et Marie Curie.

Giatsidis, C., Thilikos, D. M. & Vazirgiannis, M. (2011). Evaluating cooperation in communities with the k-core structure. *Proc. 2011 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'11)* (pp. 87–93). IEEE.

Girvan, M. & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *99*, 7821–7826.

Gleich, D. F. & Seshadhri, C. (2012). Vertex Neighborhoods, Low Conductance Cuts, and Good Seeds for Local Community Methods. *Proc. of the 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'12)* (pp. 597–605). ACM.

Gondran, M. & Minoux, M. (1995). *Graphes et algorithmes*. Eyrolles.

Granell, C., Darst, R. K., Arenas, A., Fortunato, S. & Gómez, S. (2015). Benchmark model to assess community structure in evolving networks. *Phys. Rev. E*, *92*, 012805.

Granovetter, M. S. (1973). The Strength of Weak Ties. *Amer. J. of Sociology*, *78*, 1360–1380.

Greene, D., Doyle, D. & Cunningham, P. (2010). Tracking the Evolution of Communities in Dynamic Social Networks. *Proc. 2010 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'10)* (pp. 176–183).

Gregory, S. (2010). Finding overlapping communities in networks by label propagation. *New Journal of Physics*, *12*, 1–26.

Gregory, S. (2011). Fuzzy overlapping communities in networks. *Journal of Statistical Mechanics : Theory and Experiment*, *2011*, P02017.

Guillaume, J.-L. (2012). *Déterminisme et non-déterminisme au service de la détection de communautés dynamiques*. Habilitation à Diriger les Recherches. Université Pierre et Marie Curie.

Görke, R., Maillard, P., Staudt, C. & Wagner, D. (2010). Modularity-Driven Clustering of Dynamic Graphs. *Experimental Algorithms*, 436–448. Springer, Berlin, Heidelberg.

Habib, M. & Paul, C. (2010). A survey of the algorithmic aspects of modular decomposition. *Computer Science Review*, *4*, 41–59.

Han, M., Daudjee, K., Ammar, K., Özsu, M. T., Wang, X. & Jin, T. (2014a). An Experimental Comparison of Pregel-like Graph Processing Systems. *Proc. VLDB Endow.*, *7*, 1047–1058.

Han, X., Wang, L., Park, S., Cuevas, A. & Crespi, N. (2014b). Alike people, alike interests ? A large-scale study on interest similarity in social networks. *Proc. 2014 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'14)* (pp. 491–496).

Handcock, M. S., Raftery, A. E. & Tantrum, J. M. (2007). Model-based clustering for social networks. *Journal of the Royal Statistical Society : Series A (Statistics in Society)*, *170*, 301–354.

Hastings, M. B. (2006). Community detection as an inference problem. *Phys. Rev. E*, *74*, 035102.

He-Li, S., Jian-Bin, H., Yong-Qiang, T., Qin-Bao, S. & Huai-Liang, L. (2015). Detecting overlapping communities in networks via dominant label propagation. *Chinese Phys. B*, *24*, 018703.

Hern, A. (2014). *Firechat updates as 40,000 Iraqis download 'mesh' chat app in censored Baghdad*. The Guardian, UK.

Hmimida, M. & Kanawati, R. (2015). Community Detection in Multiplex Networks : a Seed-Centric Approach. *Networks & Heterogeneous Media*, *10*.

Hoff, P. D., Raftery, A. E. & Handcock, M. S. (2002). Latent Space Approaches to Social Network Analysis. *J. Amer. Stat. Assoc.*, *97*, 1090–1098.

Holland, P. W., Laskey, K. B. & Leinhardt, S. (1983). Stochastic blockmodels : First steps. *Social Networks*, *5*, 109–137.

Hoory, S., Linial, N. & Wigderson, A. (2006). Expander graphs and their applications. *Bull. Amer. Math. Soc.*, *43*, 439–561.

Hopcroft, J., Khan, O., Kulis, B. & Selman, B. (2004). Tracking evolving communities in large linked networks. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *101 Suppl 1*, 5249–5253.

Hric, D., Darst, R. K. & Fortunato, S. (2014). Community detection in networks : Structural communities versus ground truth. *Phys. Rev. E*, *90*, 062805.

Hubert, L. & Arabie, P. (1985). Comparing partitions. *J. Classif.*, *2*, 193–218.

Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J. & Diot, C. (2005). Pocket Switched Networks and Human Mobility in Conference Environments. *Proc. 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking* (pp. 244–251). ACM.

Hui, P., Crowcroft, J. & Yoneki, E. (2011). BUBBLE Rap : Social-Based Forwarding in Delay-Tolerant Networks. *IEEE Trans. on Mobile Computing*, *10*, 1576–1589.

Hui, P., Yoneki, E., Chan, S. Y. & Crowcroft, J. (2007). Distributed Community Detection in Delay Tolerant Networks. *Proc. 2007 ACM/IEEE Int. Workshop on Mobility in the Evolving Internet Architecture* (p. 7). ACM.

Höppner, F., Klawonn, F., Kruse, R. & Runkler, T. (1999). *Fuzzy Cluster Analysis : Methods for Classification, Data Analysis and Image Recognition*. 1st edition. John Wiley & Sons, Inc.

Jaccard, P. (1901). Etude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, *37*, 547–579.

Jain, A. K., Murty, M. N. & Flynn, P. J. (1999). Data clustering : a review. *ACM Comput. Surv.*, *31*, 264–323.

James, L. O., Stanton, R. G. & Cowan, D. D. (1972). Graph decomposition for undirected graphs. *Proc. 3rd Southeastern Conf. on Combinatorics, Graph Theory, and Computing* (pp. 281–290).

Jardine, N. & Sibson, R. (1968). The construction of hierarchic and non-hierarchic classifications. *The computer journal*, *11*, 177–184.

Jeh, G. & Widom, J. (2003). Scaling Personalized Web Search. *Proc. 12th Int. Conf. on World Wide Web (WWW'03)* (pp. 271–279). ACM.

Jerrum, M. & Sinclair, A. (1989). Approximating the Permanent. *SIAM J. Comput.*, *18*, 1149–1178.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, *32*, 241–254.

Jünger, M. & Mutzel, P. (2012). *Graph drawing software*. Springer Science & Business Media.

Kanawati, R. (2014a). Seed-centric approaches for community detection in complex networks. In *Social Computing and Social Media*, 197–208. Springer.

Kanawati, R. (2014b). YASCA : an ensemble-based approach for community detection in complex networks. In *Computing and Combinatorics*, 657–666. Springer.

Karamshuk, D., Boldrini, C., Conti, M. & Passarella, A. (2011). Human mobility models for opportunistic networks. *IEEE Commun. Magazine*, *49*, 157–165.

Karrer, B. & Newman, M. E. J. (2011). Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, *83*, 016107.

Karypis, G., Han, E.-H. & Kumar, V. (1999). Chameleon : Hierarchical clustering using dynamic modeling. *Computer*, *32*, 68–75.

Karypis, G. & Kumar, V. (1998). A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM J. Sci. Comput.*, *20*, 359–392.

Kernighan, B. W. & Lin, S. (1970). An Efficient Heuristic Procedure for Partitioning Graphs. *Bell System Technical Journal*, *49*, 291–307.

Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y. & Porter, M. A. (2014). Multilayer networks. *J. Complex Netw.*, *2*, 203–271.

Kleinberg, J. (2002). Small-world phenomena and the dynamics of information. *Advances in Neural Information Processing Systems (NIPS'02)*, *14*, 431–438.

Kleinberg, J. & Tardos, E. (2005). *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc.

Kleinberg, J. M. (2003). An impossibility theorem for clustering. *Advances in Neural Information Processing Systems (NIPS'03)* (pp. 463–470).

Kloster, K. & Gleich, D. F. (2014). Heat Kernel Based Community Detection. *Proc. 20th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining* (pp. 1386–1395). ACM.

Knoke, D. (1994). *Political networks : the structural perspective*, vol. 4. Cambridge University Press.

Knoke, D. (2014). *Economic networks*. John Wiley & Sons.

Kossinets, G. & Watts, D. J. (2009). Origins of Homophily in an Evolving Social Network. *Amer. J. of Sociology*, *115*, 405–450.

Krebs, V. (2002). Uncloaking Terrorist Networks. *First Monday*, *7*.

Labatut, V. (2014). Étude de l'omniprésence des propriétés petit-monde et sans-échelle. *Proc. 5th French Conf. Modèles et Analyse des Réseaux : Approches Mathématiques et Informatiques (MARAMI'14)*.

Labatut, V. (2015). Generalised measures for the evaluation of community detection methods. *Int. J. Social Network Mining*, *2*, 44–63.

Laibowitz, M., Gips, J., Aylward, R., Pentland, A. & Paradiso, J. (2006). A sensor network for social dynamics. *Proc. 2006 Information Processing in Sensor Networks (ISPN'06)* (pp. 483–491).

Lancichinetti, A. & Fortunato, S. (2011). Limits of modularity maximization in community detection. *Phys. Rev. E*, *84*, 066122.

Lancichinetti, A., Fortunato, S. & Kertesz, J. (2009). Detecting the overlapping and hierarchical community structure of complex networks. *New J. of Physics*, *11*, 033015.

Lancichinetti, A., Fortunato, S. & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, *78*, 046110.

Lancichinetti, A., Radicchi, F., Ramasco, J. J. & Fortunato, S. (2011). Finding Statistically Significant Communities in Networks. *PLOS ONE*, *6*, e18961.

Lang, K. & Rao, S. (2004). A Flow-Based Method for Improving the Expansion or Conductance of Graph Cuts. In *Integer Programming and Combinatorial Optimization*, LNCS, vol. 3064, 325–337. Springer Berlin Heidelberg.

Largeron, C., Mougel, P.-N., Rabbany, R. & Zaïane, O. R. (2015). Generating Attributed Networks with Communities. *PLOS ONE*, *10*, e0122777.

Lee, C., Reid, F., McDaid, A. & Hurley, N. (2010). Detecting highly overlapping community structure by greedy clique expansion. *Proc. of the 4th SNA-KDD workshop on social network mining and analysis, 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'09)*.

Lee, K., Hui, P. & Chong, S. (2013). Mobility Models in Opportunistic Networks. In *Mobile Ad Hoc Networking*, 360–418. John Wiley & Sons, Inc.

Lee, K.-H., Lee, Y.-J., Choi, H., Chung, Y. D. & Moon, B. (2012). Parallel Data Processing with MapReduce : A Survey. *SIGMOD Rec.*, *40*, 11–20.

Leicht, E. A., Holme, P. & Newman, M. E. J. (2006). Vertex similarity in networks. *Phys. Rev. E*, *73*, 026120.

Lepp, H. (2015). An Investigation of Decentralized Networks Based Upon Wireless Mobile Technologies. *Intersect : The Stanford Journal of Science, Technology and Society*, *8*.

Leskovec, J. & Faloutsos, C. (2006). Sampling from Large Graphs. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'06)* (pp. 631–636). ACM.

Leskovec, J., Kleinberg, J. & Faloutsos, C. (2005). Graphs over Time : Densification Laws, Shrinking Diameters and Possible Explanations. *Proc. 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'05)* (pp. 177–187). ACM.

Leskovec, J., Lang, K. J., Dasgupta, A. & Mahoney, M. W. (2008). Statistical Properties of Community Structure in Large Social and Information Networks. *Proc. 17th Int. Conf. on World Wide Web (WWW'08)* (pp. 695–704). ACM.

Leskovec, J., Lang, K. J., Dasgupta, A. & Mahoney, M. W. (2009). Community Structure in Large Networks : Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Internet Mathematics*, *6*, 29–123.

Leung, I. X. Y., Hui, P., Liò, P. & Crowcroft, J. (2009). Towards real-time community detection in large networks. *Phys. Rev. E*, *79*, 066107.

Liben-Nowell, D. & Kleinberg, J. (2007). The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci.*, *58*, 1019–1031.

Lin, Y.-R., Chi, Y., Zhu, S., Sundaram, H. & Tseng, B. L. (2008). Facetnet : A Framework for Analyzing Communities and Their Evolutions in Dynamic Networks. *Proc. 17th Int. Conf. on World Wide Web (WWW'08)* (pp. 685–694). ACM.

Lovász, L. (1996). Random Walks on Graphs : A Survey. In *Combinatorics, Paul Erdős is Eighty*, vol. 2, 353–398. Budapest : János Bolyai Mathematical Society.

Lovász, L. & Simonovits, M. (1993). Random walks in a convex body and an improved volume algorithm. *Random Struct. Alg.*, *4*, 359–412.

Luo, F., Wang, J. Z. & Promislow, E. (2008). Exploring local community structures in large networks. *Web Intelligence and Agent Systems : An International Journal*, *6*, 387–400.

Lü, L. & Zhou, T. (2011). Link prediction in complex networks : A survey. *Physica A : Statistical Mechanics and its Applications*, *390*, 1150–1170.

Ma, L., Huang, H., He, Q., Chiew, K. & Liu, Z. (2014). Toward seed-insensitive solutions to local community detection. *J. Intell. Inf. Syst.*, *43*, 183–203.

Macker, J. P. & Corson, M. S. (1998). Mobile Ad Hoc Networking and the IETF. *SIGMOBILE Mob. Comput. Commun. Rev.*, *2*, 9–14.

Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N. & Czajkowski, G. (2010). Pregel : A System for Large-scale Graph Processing. *Proc. 2010 ACM SIGMOD Int. Conf. on Management of Data (ICMD'10)* (pp. 135–146). ACM.

Mann, S. (1996). Smart Clothing : The Shift to Wearable Computing. *Commun. ACM*, *39*, 23–24.

Marsden, S. A. J., Wiggins, L. S. S., Glass, L., Kohn, R. & Sastry, S. (1993). *Interdisciplinary Applied Mathematics*. Springer.

Martelot, E. L. & Hankin, C. (2014). Fast Multi-Scale Detection of Relevant Communities in Large-Scale Networks. *Computing, 96(11)*, 1011-1027. Springer-Verlag.

Masson, M.-H. & Denoeux, T. (2008). ECM : An evidential version of the fuzzy c-means algorithm. *Pattern Recognition*, *41*, 1384–1397.

Matsuda, H., Ishihara, T. & Hashimoto, A. (1999). Classifying molecular sequences using a linkage graph with their pairwise similarities. *Theoretical Computer Science*, *210*, 305–325.

McAuley, J. J., da Fontoura Costa, L. & Caetano, T. S. (2007). Rich-club phenomenon across complex network hierarchies. *Appl. Phys. Lett.*, *91*, 084103.

McConnell, R. M. & De Montgolfier, F. (2005). Linear-time modular decomposition of directed graphs. *Discrete Applied Mathematics*, *145*, 198–209.

McCune, R. R., Weninger, T. & Madey, G. (2015). Thinking Like a Vertex : A Survey of Vertex-Centric Frameworks for Large-Scale Distributed Graph Processing. *ACM Comput. Surv.*, *48*, 25 :1–25 :39.

McDaid, A. F., Greene, D. & Hurley, N. (2011). Normalized Mutual Information to evaluate overlapping community finding algorithms. *arXiv :1110.2515 [physics]*.

McPherson, M., Smith-Lovin, L. & Cook, J. M. (2001). Birds of a feather : Homophily in social networks. *Annual review of sociology*, *27*, 415–444.

Melo, D. d. F. P., Fadigas, I. d. S. & Pereira, H. B. d. B. (2016). Community detection in visibility networks : an approach to categorize percussive influence on audio musical signals. *Complex Networks & Their Applications V* (pp. 321–334). Springer, Cham.

Merton, R. K. (1968). The Matthew effect in science. *Science*, *159*, 56–63.

Mihail, M., Papadimitriou, C. & Saberi, A. (2003). On certain connectivity properties of the Internet topology. *Proc. 44th IEEE Symp. on Foundations of Computer Science (FOCS'03)* (pp. 28–35).

Milgram, S. (1967). The Small World Problem. *Psychology Today*, *1*, 61–67.

Mitra, B., Tabourier, L. & Roth, C. (2012). Intrinsically dynamic network communities. *Computer Networks*, *56*, 1041–1053.

Mohar, B. (1989). Isoperimetric numbers of graphs. *J. Combinatorial Theory, Series B*, *47*, 274–291.

Montgolfier, F. d. (2003). *Décomposition modulaire des graphes : théorie, extensions et algorithmes*. Thèse de doctorat. Montpellier 2, Université des Sciences et Techniques du Languedoc.

Montresor, A., Pellegrini, F. D. & Miorandi, D. (2013). Distributed k-Core Decomposition. *IEEE Trans. on Parallel and Distributed Systems*, *24*, 288–300.

Moradi, F., Olovsson, T. & Tsigas, P. (2014). A local seed selection algorithm for overlapping community detection. *Proc. 2014 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'14)* (pp. 1–8).

Moreno, J. L. & Jennings, H. H. (1938). Statistics of Social Configurations. *Sociometry*, *1*, 342–374.

Mucha, P. J., Richardson, T., Macon, K., Porter, M. A. & Onnela, J.-P. (2010). Community Structure in Time-Dependent, Multiscale, and Multiplex Networks. *Science*, *328*, 876–878.

Newman, M. (2003). The Structure and Function of Complex Networks. *SIAM Rev.*, *45*, 167–256.

Newman, M. (2010). *Networks : an introduction*. Oxford University Press.

Newman, M. E. J. (2004a). Coauthorship networks and patterns of scientific collaboration. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *101*, 5200–5205.

Newman, M. E. J. (2004b). Detecting community structure in networks. *Eur. Phys. J. B*, *38*, 321–330.

Newman, M. E. J. (2004c). Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, *69*, 066133.

Newman, M. E. J. (2012). Communities, modules and large-scale structure in networks. *Nat. Phys.*, *8*, 25–31.

Newman, M. E. J. (2013). Spectral methods for network community detection and graph partitioning. *Phys. Rev. E*, *88*.

Newman, M. E. J. & Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, *69*, 026113.

Newman, M. E. J. & Leicht, E. A. (2007). Mixture models and exploratory analysis in networks. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *104*, 9564–9569.

Ngonmang, B., Tchuente, M. & Viennet, E. (2012). Local community identification in social networks. *Parallel Processing Letters*, *22*, 1240004.

Orecchia, L. & Zhu, Z. A. (2014). Flow-based Algorithms for Local Graph Clustering. *Proc. 25th ACM-SIAM Symp. on Discrete Algorithms* (pp. 1267–1286). SIAM.

Orlinski, M. & Filer, N. (2013). The rise and fall of spatio-temporal clusters in mobile ad hoc networks. *Ad Hoc Networks*, *11*, 1641–1654.

Orman, G., Labatut, V., Plantevit, M. & Boulicaut, J.-F. (2014). A method for characterizing communities in dynamic attributed complex networks. *Proc. 2014 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'14)* (pp. 481–484).

Orman, G. K., Labatut, V. & Cherifi, H. (2013). Towards Realistic Artificial Benchmark for Community Detection Algorithms Evaluation. *Int. J. Web Based Communities*, *9*, 349–370.

Ouellet, M. L. (2016). *Terrorist Networks and the Collective Criminal Career : The Relationship between Group Structure and Trajectories.* Thesis, Arts & Social Sciences : School of Criminology. Simon Fraser University.

Ozawa, K. (1985). A stratificational overlapping cluster scheme. *Pattern Recognition*, *18*, 279 – 286.

Palla, G., Barabási, A.-L. & Vicsek, T. (2007). Quantifying social group evolution. *Nature*, *446*, 664–667.

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

Pan, G., Zhang, W., Wu, Z. & Li, S. (2014). Online Community Detection for Large Complex Networks. *PLoS ONE*, *9*, e102799.

Papadopoulos, S., Kompatsiaris, Y., Vakali, A. & Spyridonos, P. (2012). Community detection in Social Media. *Data Mining Knowledge Discovery*, *24*, 515–554.

Paradiso, J. A., Gips, J., Laibowitz, M., Sadi, S., Merrill, D., Aylward, R., Maes, P. & Pentland, A. (2010). Identifying and Facilitating Social Interaction with a Wearable Wireless Sensor Network. *Personal Ubiquitous Comput.*, *14*, 137–152.

Plantevit, M., Choong, Y. W., Laurent, A., Laurent, D. & Teisseire, M. (2005). M2sp : Mining Sequential Patterns Among Several Dimensions. *Proc. Eur. Conf. Principles and Practice of Knowledge Discovery in Databases (PKDD'05)* (pp. 205–216). Springer, Berlin, Heidelberg.

Pons, P. & Latapy, M. (2005). Computing Communities in Large Networks Using Random Walks. In *Computer and Information Sciences - ISCIS 2005*, LNCS, 3733, 284–293. Springer Berlin Heidelberg.

Prat-Pérez, A., Dominguez-Sal, D., Brunat, J. M. & Larriba-Pey, J.-L. (2012). Shaping communities out of triangles. *Proc. 21st ACM Int. Conf. on Information and Knowledge Management (CIKM'12)* (pp. 1677–1681). ACM.

Price, D. d. S. (1976). A general theory of bibliometric and other cumulative advantage processes. *J. Amer. Soc. Information science*, *27*, 292–306.

Rabbany, R., Chen, J. & Zaıane, O. R. (2010). Top leaders community detection approach in information networks. *Proc. of the 4th SNA-KDD workshop on social network mining and analysis, 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'09)*.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. & Parisi, D. (2004). Defining and identifying communities in networks. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *101*, 2658–2663.

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

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *J. Am. Stat. Assoc.*, *66*, 846–850.

Reichardt, J. & Bornholdt, S. (2006). Statistical Mechanics of Community Detection. *Phys. Rev. E*, *74*.

Reid, F., McDaid, A. & Hurley, N. (2013). Partitioning Breaks Communities. In *Mining Social Networks and Security Informatics*, LNSN, vol. 3, 79–105. Springer Netherlands.

Riedy, E. J., Meyerhenke, H., Ediger, D. & Bader, D. A. (2012). Parallel Community Detection for Massive Graphs. In *Parallel Processing and Applied Mathematics*, LNCS, vol. 7203, 286–296. Springer Berlin Heidelberg.

Riedy, J., Bader, D. A., Jiang, K., Pande, P. & Sharma, R. (2011). *Detecting communities from given seeds in social networks*. Technical Report. Georgia Institute of Technology.

Rigney, D. (2010). *The Matthew Effect : How Advantage Begets Further Advantage*. Columbia University Press.

Rosvall, M. & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. U.S.A. (PNAS)*, *105*, 1118–1123.

Salton, G. & McGill, M. J. (1984). *Introduction to modern information retrieval*. McGraw-Hill Inc., (Computer Science Series).

Saltz, M., Prat-Pérez, A. & Dominguez-Sal, D. (2015). Distributed Community Detection with the WCC Metric. *Proc. 24th Int. Conf. on World Wide Web (WWW'15)* (pp. 1095–1100). ACM.

Santos, J. M. & Embrechts, M. (2009). On the use of the adjusted rand index as a metric for evaluating supervised classification. *Proc. Int. Conf. on Artificial Neural Networks* (pp. 175–184). Springer.

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, *1*, 27–64.

Schaub, M. T., Delvenne, J.-C., Rosvall, M. & Lambiotte, R. (2017). The many facets of community detection in complex networks. *Applied Network Science*, *2*, 4.

Schweitzer, F. (1997). *Self-organization of complex structures : From individual to collective dynamics*. CRC Press.

Scott, J. (2012). *Social Network Analysis*. SAGE.

Seshadhri, C., Kolda, T. G. & Pinar, A. (2012). Community structure and scale-free collections of Erdős-Rényi graphs. *Phys. Rev. E*, *85*, 056109.

Shah, D. & Zaman, T. (2010). Community detection in networks : The leader-follower algorithm. *Proc. 2010 NIPS Workshop on Networks Across Discilines in Theory and Applications*.

Shalizi, C. R. & Thomas, A. C. (2011). Homophily and contagion are generically confounded in observational social network studies. *Sociological methods & research*, *40*, 211–239.

Shannon, C. E. (1948). A mathematical theory of communication, Part I, Part II. *Bell Syst. Tech. J.*, *27*, 623–656.

Shen, H.-W., Cheng, X.-Q. & Guo, J.-F. (2009). Quantifying and identifying the overlapping community structure in networks. *J. Stat. Mech. : Theory and Experiment*, *2009*, P07042.

Shepard, R. N. & Arabie, P. (1979). Additive clustering : Representation of similarities as combinations of discrete overlapping properties. *Psychological Review*, *86*, 87–123.

Shi, J. & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, *22*, 888–905.

Spielman, D. A. & Teng, S.-H. (2004). Nearly-linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems. *Proc. 36th ACM Symp. on Theory of Computing* (pp. 81–90). ACM.

Strehl, A. & Ghosh, J. (2003). Cluster Ensembles — a Knowledge Reuse Framework for Combining Multiple Partitions. *J. Mach. Learn. Res. (JMLR)*, *3*, 583–617.

Sueur, C., Jacobs, A., Amblard, F., Petit, O. & King, A. J. (2011). How can social network analysis improve the study of primate behavior ? *Amer. J. Primatology*, *73*, 703–719.

Suh, M., Carroll, K. E. & Cassill, N. L. (2010). Critical review on smart clothing product development. *J. Textile and Apparel, Technology and Management*, *6*.

Tang, L. & Liu, H. (2009). Relational Learning via Latent Social Dimensions. *Proc. 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'09)* (pp. 817–826). ACM.

Tantipathananandh, C., Berger-Wolf, T. & Kempe, D. (2007). A framework for community identification in dynamic social networks. *Proc. 13th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'07)* (pp. 717–726). ACM.

Tedder, M., Corneil, D., Habib, M. & Paul, C. (2008). Simpler linear-time modular decomposition via recursive factorizing permutations. *Automata, languages and programming*, 634–645.

Tian, Y., Balmin, A., Corsten, S. A., Tatikonda, S. & McPherson, J. (2013). From "Think Like a Vertex" to "Think Like a Graph". *Proc. VLDB Endow.*, *7*, 193–204.

Tournoux, P. U., Leguay, J., Benbadis, F., Whitbeck, J., Conan, V. & Amorim, M. D. d. (2011). Density-Aware Routing in Highly Dynamic DTNs : The RollerNet Case. *IEEE Trans. on Mobile Computing*, *10*, 1755–1768.

Tseng, Y.-C., Ni, S.-Y., Chen, Y.-S. & Sheu, J.-P. (2002). The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Wireless Networks*, *8*, 153–167.

van Laarhoven, T. & Marchiori, E. (2016). Local Network Community Detection with Continuous Optimization of Conductance and Weighted Kernel K-Means. *J. Mach. Lear. Res. (JMLR)*, *17*, 1–28.

Viard, J., Latapy, M. & Magnien, C. (2016). Computing maximal cliques in link streams. *Theoretical Computer Science, 609*. 245–252.

Viard, T. (2016). *Link streams for the modelling of interactions over time and application to the analysis of IP traffic*. Thèse de doctorat. Université Pierre et Marie Curie.

Vicsek, T. (2002). Complexity : The bigger picture. *Nature*, *418*, 131–131.

Wasserman, S. & Faust, K. (1994). *Social Network Analysis : Methods and Applications*. Cambridge : Cambridge University Press. 1 edition édition.

Watts, D. J. & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*, 440–442.

Whang, J. J., Gleich, D. F. & Dhillon, I. S. (2013). Overlapping community detection using seed set expansion. *Proc. 22nd ACM Int. Conf. on Information & Knowledge Management (CIKM'13)* (pp. 2099–2108). ACM.

Xie, J., Kelley, S. & Szymanski, B. K. (2013). Overlapping Community Detection in Networks : The State-of-the-art and Comparative Study. *ACM Comput. Surv.*, *45*, 43.

Xie, J., Szymanski, B. K. & Liu, X. (2011). SLPA : Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process. *Proc. 2011 Workshop on Data Mining Technologies for Computational Collective Intelligence (DMCCI), 11th IEEE Int. Conf. on Data Mining (ICDM'11)*.

Yakoubi, Z. & Kanawati, R. (2014). LICOD : A Leader-driven algorithm for community detection in complex networks. *Vietnam J. Comput. Sci.*, *1*, 241–256.

Yang, J. & Leskovec, J. (2015). Defining and evaluating network communities based on ground-truth. *Knowl. Inf. Syst.*, *42*, 181–213.

Youness, G. & Saporta, G. (2004). Une méthodologie pour la comparaison de partitions. *Revue de statistique appliquée*, *52*, 97–120.

Yule, G. U. (1925). A mathematical theory of evolution, based on the conclusions of Dr. JC Willis, FRS. *Philosophical Trans. Royal Society of London. Series B, containing papers of a biological character*, *213*, 21–87.

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

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, *8*, 338–353.

Zakrzewska, A. & Bader, D. A. (2015). A Dynamic Algorithm for Local Community Detection in Graphs. *Proc. 2015 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM'15)* (pp. 559–564). ACM.

Zanghi, H., Volant, S. & Ambroise, C. (2010). Clustering based on random graph model embedding vertex features. *Pattern Recognition Letters*, *31*, 830–836.

Zardi, H., Romdhane, L. B. & Guessoum, Z. (2014). A Multi-agent Homophily-Based Approach for Community Detection in Social Networks. *Proc. 26th IEEE Int. Conf. on Tools with Artificial Intelligence (ICTAI'14)* (pp. 501–505).

Zhang, Y., Wang, J., Wang, Y. & Zhou, L. (2009). Parallel Community Detection on Large Networks with Propinquity Dynamics. *Proc. 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining (KDD'09)* (pp. 997–1006). ACM.

Zhao, Y. (2017). A survey on theoretical advances of community detection in networks. *WIREs Computational Statistics*, *9(5)*.

Zhou, K., Martin, A., Pan, Q. & Liu, Z.-g. (2015). Median evidential c-means algorithm and its application to community detection. *Knowledge-Based Systems*, *74*, 69–88.

Zhou, T., Lü, L. & Zhang, Y.-C. (2009a). Predicting missing links via local information. *Eur. Phys. J. B*, *71*, 623–630.

Zhou, Y., Cheng, H. & Yu, J. X. (2009b). Graph Clustering Based on Structural/Attribute Similarities. *Proc. 35th Int. Conf. on Very Large DataBases (VLDB'09)*, Proc. VLDB Endow., vol. *2*, 718–729.

Zhu, Z. A., Lattanzi, S. & Mirrokni, V. (2013). A Local Algorithm for Finding Well-Connected Clusters. *Proc. 30th Int. Conf. on Machine Learning (ICML'13)* (pp. 396–404). JMLR Workshop and Conference Proceedings.

Šubelj, L. & Bajec, M. (2011). Robust network community detection using balanced propagation. *Eur. Phys. J. B-Condensed Matter and Complex Systems (EPJB)*, *81*, 353–362.

Annexes

Annexe A

Critères d'évaluation expérimentaux

Sommaire

Cette annexe présente différents critères utilisés afin d'évaluer la qualité des communautés détectées pour les expériences proposées dans ce manuscrit. Ils sont utilisés afin de comparer deux partitions, généralement l'une produite par l'algorithme évalué et l'autre servant de base de comparaison, produite par un algorithme de référence ou constituée par annotation d'expert (vérité terrain).

Ces critères sont également présentés dans différents états de l'art sur la détection de communautés (Fortunato, 2009, Xie et al., 2013) et sur le partitionnement de données en général (Jain et al., 1999), ainsi que dans différents articles cités. Bien que nous en citions et utilisions un certain nombre, de nombreuses controverses subsistent quant à la propension de certains critères à représenter la qualité ou la précision d'un partitionnement. Différents travaux s'attachent à évaluer la pertinence des critères eux-mêmes (Almeida et al., 2011, Amelio & Pizzuti, 2015, Labatut, 2015).

Nous commençons par rappeler des notations utiles pour exprimer les différents critères, en particulier concernant la matrice de confusion. Nous poursuivons en évoquant l'indice de Jaccard section A.1, puis l'indice Rand et sa version ajustée section A.2. Nous présentons ensuite les mesures basées sur l'entropie d'information NMI, section A.3.1 et NVI A.3.2 et concluons cette annexe avec, section A.4, une mesure basée sur la précision et le rappel : le score F_1.

Avant-propos : éléments en commun et matrice de confusion

Un certain nombre d'indices et mesures présentés dans la suite de l'annexe utilisent le nombre d'éléments en commun dans les clusters des partitions considérées (dans les expériences, les clusters comparés sont des communautés).

Soient deux partitions X et Y disjointes des sommets d'un graphe $G = (V, E)$ et $n = |V|$, telles que $X = \{X_1, X_2, ..., X_r\}$ et $Y = \{Y_1, Y_2, ..., Y_s\}$.

La matrice de confusion M de ces deux partitions est donc de taille $r \times s$ et chaque élément M_{ij} est le nombre d'éléments en commun entre les clusters X_i et Y_j :

$$i \in [\![1, r]\!], j \in [\![1, s]\!] M_{ij} = |X_i \cap Y_j|$$

A.1 Indice de Jaccard

Une des mesures de comparaison de partition les plus anciennes est l'indice de Jaccard (1901), introduit pour comparer la distribution d'espèces de fleurs de montagne.

De façon simple, l'indice de Jaccard, noté J, calculé entre deux ensembles A et B, est le rapport entre le cardinal de l'intersection de ces ensembles et celui de leur union :

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

La formulation de cet indice pour des partitions en sous-ensembles X, Y est plus complexe, nous l'exprimons donc à l'aide de la matrice de confusion M :

$$J(X, Y) = \frac{\sum_i \sum_j C_2^{M_{ij}}}{\sum_i C_2^{|X_i|} + \sum_j C_2^{|Y_j|} - \sum_i \sum_j C_2^{M_{ij}}}$$

A.2 Indice de Rand (RI) et version ajustée (ARI)

L'indice de Rand (1971) (*Rand Index*, RI) est une mesure utilisée pour comparer deux partitions d'un même ensemble, en évaluant leur degré de similarité.

Le principe de cet indice est d'établir un rapport entre le nombre de couples d'éléments présents dans les mêmes ensembles, ou dans différents ensembles, dans les deux partitions considérées.

On exprime généralement le Rand Index RI sous la forme intuitive suivante, avec $i, i_1, i_2 \in [\![1, r]\!], j, j_1, j_2 \in [\![1, s]\!], i_1 \neq i_2, j_1 \neq j_2$:

$a = |\{(u, v) \in V^2 | u, v \in X_i \text{et} u, v \in Y_j\}|$, le nombre de couples de sommets présents dans le même sous-ensemble dans X *et* dans Y.

$b = |\{(u, v) \in V^2 | u \in X_{i_1}, v \in X_{i_2} \text{et} u \in Y_{j_1}, v \in Y_{j_2}\}|$, le nombre de couples de sommets présents dans des sous-ensembles différents dans X *et* dans Y.

$c = |\{(u,v) \in V^2 | u,v \in X_i \text{et} u \in Y_{j_1}, v \in Y_{j_2}\}|$, le nombre de couples de sommets présents dans le même sous-ensemble dans X mais dans des sous-ensembles différents dans Y.

$d = |\{(u,v) \in V^2 | u \in X_{i_1}, v \in X_{i_2} \text{et} u,v \in Y_j\}|$, le nombre de couple de sommets présents dans des sous-ensembles différents dans X mais dans le même sous-ensemble dans Y.

L'indice de Rand RI est alors exprimé de la façon suivante :

$$\text{RI} = \frac{a+b}{a+b+c+d}$$

On remarque que la somme $a+b+c+d$ correspond au nombre de couples de sommets, soit C_2^n.

L'indice de Rand varie entre 0 et 1 : 0 indiquant que les sous-ensembles des deux partitions n'ont aucun couple en commun et 1 qu'ils sont identiques. Lors de l'évaluation des résultats d'une tâche de détection de communautés, si l'on compare à une partition de référence on souhaite que le maximum de couples de sommets soient placés dans les mêmes ensembles, et donc, on souhaite maximiser l'indice.

Problème avec l'indice de Rand

L'un des problèmes majeurs de l'indice de Rand est qu'il peut être facilement biaisé (Hubert & Arabie, 1985, Santos & Embrechts, 2009), en effet l'espérance de cet indice entre deux partitions aléatoires n'est pas nulle, et augmente « artificiellement » avec le nombre d'ensembles dans les partitions considérées. Deux cas extrêmes sont à considérer : d'abord si la partition considérée comporte très peu d'ensembles par rapport à la taille des données, qui regroupent donc de nombreux sommets et augmentent beaucoup la valeur de a et également si elle comporte de très nombreux ensembles de petite taille, qui diminuent drastiquement le nombre de couples de sommets possibles dans ces ensembles et fait donc augmenter la valeur de d, alors que la répartition proposée peut être tout à fait pertinente.

C'est pour cela que différentes variantes ont été proposées (Youness & Saporta, 2004), dont l'indice de Rand Ajusté que nous détaillons ci-dessous.

Indice de Rand ajusté

L'indice de Rand ajusté (Adjusted Rand Index, ARI) est une version de l'indice de Rand décrit ci-dessus intégrant la probabilité que deux couples de sommets soient classées « au hasard » dans un même ensemble (Hubert & Arabie, 1985). Cette probabilité est à définir dans la formule de l'ARI, on la choisit généralement uniforme. Soit p le nombre

total de couples, on a alors :

$$\text{ARI} = \frac{C_2^p(a+d) - [(a+b)(a+c)+(c+d)(b+d)]}{(C_2^p)^2 - [(a+b)(a+c)+(c+d)(b+d)]}$$

L'ARI peut également être exprimé en fonction des M_{ij} décrits en avant-propos au début de l'annexe.

Soit $x_i = \sum_{j=1}^{s} M_{ij}$ et $y_j = \sum_{i=1}^{r} M_{ij}$.

La formule de l'ARI est alors exprimée comme suit :

$$\text{ARI} = \frac{\sum_{ij}\binom{M_{ij}}{2} - \left[\sum_i \binom{x_i}{2}\sum_j \binom{y_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{x_i}{2} + \sum_j \binom{y_j}{2}\right] - \left[\sum_i \binom{x_i}{2}\sum_j \binom{y_j}{2}\right]/\binom{n}{2}}$$

La valeur de l'ARI peut être négative, si le nombre de couples placés dans des ensembles identiques est moins élevé que la valeur attendue. Si les couples sont placés de manière identique dans les deux partitions, la valeur est de 1, et si ils sont répartis de manière aléatoire, la valeur est proche de 0.

Cela répond en partie au biais des partitions aléatoires dont souffre le RI originel, cependant d'autres biais subsistent, notamment l'augmentation « artificielle » dans le cas où les partitions sont composées de nombreux ensembles de petite taille.

A.3 Mesures basées sur l'entropie d'information

Nous présentons dans cette section deux mesures basées sur l'entropie d'information. De même que RI et ARI, ces mesures permettent de comparer la similarité de deux partitions d'un jeu de données en sous-ensembles. Comme nous l'avons vu avec l'indice de Rand, l'assertion de qualité par présence de couples de sommets dans les mêmes ensembles est contestable : la mesure peut-être biaisée, particulièrement lorsque l'on s'en sert comme critère d'optimisation, par les nombre de sous-ensembles, et donc leur taille, contenant les couples (voir *Problèmes avec l'indice de Rand* ci-dessus)

Les mesures présentées proposent, pour pallier cela, de mesurer l'entropie de l'information apportée par une paire de deux sommets, au lieu de simplement compter les couples présentes. On calcule donc l'entropie des couples de sommets présentes dans les clusters, au sens de l'entropie d'information de Shannon (1948).

Nous présentons tout d'abord l'information mutuelle normalisée ou NMI, puis une mesure proche se basant sur un calcul de la variation d'information plutôt que de l'information mutuelle : la variation d'information normalisée, NVI.

Note : lors des calculs on utilise par convention $0 \log 0 = 0$.

A.3.1 Normalised Mutual Information (NMI)

L'information mutuelle normalisée, NMI de son signe anglophone (Strehl & Ghosh, 2003), est une variante normalisée de l'information mutuelle (MI). L'information mutuelle mesure l'entropie apportée par la présence de couples de sommets dans les mêmes clusters, à la manière de l'indice de Rand. Cependant, elle ne définit pas distance car elle ne respecte pas nécessairement l'inégalité triangulaire (Amelio & Pizzuti, 2015).

Soit n le nombre total d'éléments, autrement dit le nombre de sommets dans le graphe. On exprime la NMI à l'aide de la matrice de confusion M :

$$\mathrm{NMI}(X,Y) = \frac{-2\sum_i \sum_j M_{ij} \log \frac{nM_{ij}}{|X_i|\cdot|Y_j|}}{\sum_i |X_i| \log \frac{|X_i|}{n} + \sum_j |Y_j| \log \frac{|Y_j|}{n}}$$

De même que l'indice de Rand, la NMI varie entre 0 et 1, 0 signifiant que les partitions ne partagent aucune information et 1 qu'elles partagent toute l'information, c'est-à-dire qu'elles sont identiques.

L'effet de l'entropie d'information par rapport au comptage des couples est que la présence des mêmes couples de sommets est très valorisée par rapport à la structure globale des clusters. Autrement dit, la présence des mêmes couples dans un même cluster contribue très fortement à la valeur de la NMI, y compris si le nombre de clusters dans les deux partitions est différent. Il est ainsi possible d'obtenir une valeur de NMI très haute en constituant une partition comprenant de nombreux petits clusters, qui constitueraient des sous-ensembles des clusters de la partition de référence.

Cet effet est beaucoup moins important avec l'ARI, c'est pourquoi nous présentons conjointement les deux mesures dans nos expériences : la valeur d'ARI rendant mieux compte de l'adéquation avec la structure communautaire dans son ensemble, alors que la valeur de NMI rend compte de la co-présence des sommets dans les mêmes ensembles.

Variante pour communautés recouvrantes

Nous utilisons dans le chapitre 4 une variante de la NMI prenant en compte la possibilité de multi-appartenance d'un sommet. Nous renvoyons à l'annexe B de l'article de Lancichinetti et al. (2009) pour une description détaillée incluant les équations. D'autres variantes de la NMI pour les communautés recouvrantes ont été proposées, notamment par McDaid et al. (2011).

A.3.2 Normalised Variation of Information (NVI)

La variation d'information normalisée est quant à elle une variante normalisée de la variation d'information, c'est-à-dire de la distance entre l'entropie des deux ensembles com-

parées, calculée en additionnant ces entropies et en retranchant leur information mutuelle. C'est une métrique.

Elle varie entre 0 et 1 mais le sens est inverse de celui de la NMI : 0 signifie qu'il n'y a pas de variation entre les deux ensembles comparés, donc qu'ils sont identiques, alors que 1 signifie que la variation est totale, ils n'ont donc aucun élément en commun.

Soit n le nombre total d'éléments, autrement dit le nombre de sommets dans le graphe. On exprime la NVI à l'aide de la matrice de confusion M :

$$\mathrm{NVI}(X,Y) = \frac{-1}{\log n} \sum_i \sum_j \frac{M_{ij}}{n} \log \frac{{M_{ij}}^2}{|X_i| * |Y_j|}$$

A.4 Mesures basées sur la précision

Nous présentons la précision et du rappel, ainsi qu'une mesure basée sur l'agrégation des deux : le score F_1.

En classification binaire, on distingue les éléments correctement classés (*true positives*) et correctement rejetés (*true negatives*) des éléments incorrectement classés (*false positives*) et incorrectement rejetés (*false negatives*).

Pour une partition en communautés $X = \{X_1, ..., X_r\}$ comparée à une partition de référence $Y = \{Y_1, ..., Y_s\}$, on scrute des couples de sommets, comme pour les indices présentés précédemment. Les définitions présentées sont les définitions adaptées du contexte de la classification (Jain et al., 1999), dont le vocabulaire est différent de celles concernant la recherche d'information (Salton & McGill, 1984).

On reprend pour cela le comptage des couples a, b, c, d utilisé pour le Rand Index (cf. section A.2, p. 178).

La précision P est la statistique indiquant la proportion des couples de sommets classées ensemble avec exactitude. Elle est calculée en effectuant le rapport du nombre de true positives (a) sur la somme de tous les éléments « positives » dans la partition évaluée, vrais comme faux ($a + c$) :

$$P(X,Y) = \frac{a}{a+c}$$

Le rappel R est la statistique indiquant la proportion des couples de sommets classés ensemble parmi tous les couples qui le sont également dans la référence. Elle est calculée en effectuant le rapport du nombre de true positives a sur la somme des true positives et false negatives ($a + d$, i.e. les couples « positives » dans la partition de référence) :

$$R(X,Y) = \frac{a}{a+d}$$

La précision et le rappel sont deux mesures indispensables à présenter conjointement. En effet, il est possible par exemple d'obtenir un rappel de 100% en produisant une seule communauté regroupant tous les sommets. On utilise pour cela le score F_1, qui est la

moyenne harmonique de la précision et du rappel :

$$F_1(X,Y) = 2 \times \frac{P(X,Y) \times R(X,Y)}{P(X,Y) + R(X,Y)}$$

Note : le score F_1 est un cas particulier de la mesure F_β (pour $\beta = 1$) :

$$F_\beta(X,Y) = (1+\beta^2) \times \frac{P(X,Y) \times R(X,Y)}{\beta^2 P(X,Y) + R(X,Y)}$$

Un $\beta > 1$ donne plus de poids au rappel, alors qu'un $\beta < 1$ donne plus de poids à la précision.

Annexe B

Jeux de données : graphes utilisés

Sommaire

Cette annexe présente les jeux de données fournis en entrée aux algorithmes évalués dans les différentes expériences présentées dans ce manuscrit. Nous présentons tout d'abord différents générateurs de graphes artificiels puis deux graphes tirés de situations réelles et enfin deux instances de graphes spécifiques.

B.1 Générateurs de graphes

Dans cette section, nous détaillons les générateurs de graphes *artificiels* ou *synthétiques* (*benchmark graphs* en anglais). Le manque de graphes tirés de situations réelles, et surtout leur manque de diversité, ont mené à la création de plusieurs méthodes et algorithmes pour obtenir des graphes dont un certain nombre de propriétés sont paramétrables et donc connues, garanties. Bien évidemment, malgré un niveau de perfectionnement croissant au fil des années, ces benchmarks n'arrivent à l'heure actuelle pas encore à reproduire, tout du moins de façon réaliste et combinée, certaines propriétés des réseaux réels (Orman et al., 2013).

Nous commençons par présenter un premier modèle simple mais néanmoins très utilisé dans les premiers travaux de détection algorithmique de communautés sur graphes : générer les communautés en utilisant une loi de Bernoulli. Nous présentons ensuite trois générateurs très populaires : les deux premiers générant des graphes à structure communautaire statique, proposé par Lancichinetti et al. (2008) et Dutot et al. (2007), le troisième générant des graphes dynamiques, proposé par Granell et al. (2015). A la fin de cette annexe se trouve également présentation de deux instances de graphes utilisées dans certaines expériences du chapitre 7.

B.1.1 Génération par probabilités

Une première façon simple et intuitive de générer des graphes paramétrables comprenant des communautés est de proposer une adaptation de la définition informelle d'une communauté en distribution de probabilités, au sens de probabilité de présence d'arête entre tel et tel sommet. Cette méthode a été très utilisée (Newman & Girvan, 2004, Clauset, 2005) avant l'apparition de benchmarks plus riches et complets tels que le benchmark de Lancichinetti et al. (2008) (voir sous-section suivante).

On définit préalable le nombre total de sommets du graphe n et le nombre de communautés avec, pour chacune, le nombre de sommets qu'elle contient. On se munit ensuite de deux probabilités p_{in} et p_{out}.

On divise ensuite les sommets du graphe en sous-ensembles de sommets correspondant aux communautés définies précédemment. On scrute toutes les paires de sommets existantes : chaque paire de sommets d'une même communauté sera reliée par une arête avec la probabilité p_{in} (et non reliée avec $1 - p_{in}$) et chaque paire de sommets de deux communautés différentes sera reliée avec la probabilité p_{out} (et non reliée avec $1 - p_{out}$). Il est ainsi possible de paramétrer la densité des communautés ainsi que le degré moyen attendu des sommets.

Une autre façon de décrire cette méthode permet de contrôler le degré moyen attendu plus facilement. On définit ce degré comme un paramètre z, somme des arêtes intra-communautaires z_{in} et des arêtes inter-communautaires z_{out}. Les arêtes sont placées indépendamment et aléatoirement dans le graphe, en respectant en moyenne les proportions fixées par z, z_{in} et z_{out}. On peut ainsi faire varier z, ou le fixer et faire varier z_{out}, on a alors $z_{in} = z - z_{out}$.

Bien sûr, cette méthode colle à la définition intuitive des communautés comme ensemble « dont la connectivité interne est supérieure à la connectivité externe », mais elle ne permet pas de rendre des propriétés plus complexes telles que l'effet petit-monde ou l'assortatitivé.

B.1.2 Générateur de Lancichinetti, Fortunato, Radicchi (2008)

Devant le constat des limitations de la méthode précédente, Lancichinetti et al. (2008) ont proposé une méthode, ainsi qu'un programme exécutable l'implémentant.

Ce générateur produit un graphe à nombre de communautés variable de tailles variables, garantissant une distribution des degrés en loi de puissance dont l'exposant est paramétrable (sans échelle). Des optimisations permettent d'optimiser le temps de génération tout en gardant des garanties très fortes sur cette distribution des degrés.

Il permet également à un sommet d'appartenir à plusieurs communautés, et ainsi de générer une structure en communautés recouvrantes. Une limitation cependant, est que le nombre de communautés auquel appartient chacun de ces sommets est défini de façon exact et non en tant que maximum, ce qui est peu réaliste par rapport aux constatation opérées dans les réseaux réels (Xie et al., 2013).

Les nombreux paramètres définissables incluent :

N	nombre de sommets du graphe
k	degré moyen des sommets
$\max k$	degré maximal pour un sommet
μ	paramètre de mélange (voir ci-après)
t_1	(opposé de) l'exposant de la loi de distribution des degrés
t_2	(opposé de) l'exposant de la loi de distribution des tailles de communautés
$\min / \max c$	min et max pour les tailles de communautés
o_n	nombre de sommets multi-appartenants
o_m	nombre de multi-appartenances exact pour chacun des o_n sommets multi-appartenants

Le paramètre de mélange $\mu \in [0, 1]$ est défini comme le paramètre de *mélange* des sommets entre leur communauté et le reste du graphe, représentant les probabilités du modèle précédent (section B.1.1, p. 186). Concrètement, chaque sommet possède fraction $1 - \mu$ de ses voisins dans sa propre communautés et μ en dehors, dans le reste du graphe. Ainsi, un μ élevé crée des communautés très « mélangées » entre elles et moins facilement détectables, contrairement à un μ faible.

Les seuls paramètres obligatoires sont $N, k, \max k$ et μ, le reste pouvant être utilisé au choix de l'utilisateur pour préciser un peu plus la structure du graphe : la distribution des degrés est formée selon les paramètres restants spécifiés (ou leur valeur par défaut). Si l'ensemble des paramètres est incohérent (par exemple, $\min c > k$), aucun graphe n'est généré.

B.1.3 GraphStream : générateur de Dutot, Gain, Olivier, Pigné (2007)

GraphStream (Dutot et al., 2007) ne propose pas de modèle de génération comme le générateur de Lancichinetti et al. (2008) mais une implémentation de modèles génératifs de graphes existants (aléatoire, grille, attachement préférentiel, Watts-Strogatz...) en Java. Il dispose d'une boîte à outils relativement complète et extensible par un utilisateur, permettant de générer, exporter, charger, manipuler et visualiser tous types de graphes.

La génération se fait par ajout successif d'arêtes, conformément au modèle génératif choisi. Ainsi, GraphStream peut être utilisé pour générer des graphes statiques (état terminal d'une suite finie d'ajouts d'arêtes) ou un graphe dynamique si l'on sauvegarde chaque état du processus de génération.

Nous l'avons utilisé afin de créer des graphes artificiels d'attachement préférentiel utilisés pour certaines expériences du chapitre 7.

B.1.4 Générateur de Granell, Darst, Arenas, Fortunato, Gómez (2015)

Le dernier générateur que nous présentons est l'œuvre de Granell et al. (2015) et permet de construire des graphes dynamiques sous la forme d'une séquence de graphes, chaque instance représentant un état successif du graphe.

Le nombre de sommets est paramétré par l'utilisateur et ne change pas au cours de l'évolution, seules les arêtes apparaissent et disparaissent pour refléter le changement de la structure communautaire. Le nombre de communautés présentes au départ est également fixé par un paramètre et doit être un multiple de 2 (de 4 pour le motif mixed), ce qui est une première limitation du générateur de Granell et al. (2015).

La répartition originelle des arêtes est définie à l'aide de probabilités liées à la densité intra/intercommunautaire. En cela le graphe originel, à l'état t_0, se rapproche d'un graphe généré par le processus décrit section B.1.1, p. 186. Ce processus étant simpliste par rapport à l'état actuel des générateurs de graphes artificiels, on peut considérer que c'est une seconde limitation de ce générateur.

Les communautés vont ensuite évoluer suivant un motif, également paramétré, choisi parmi 3 possibilités :

— grow-shrink : certaines communautés vont grossir et d'autres rétrécir en conséquence.
— merge-split : certaines communautés vont fusionner et d'autres se scinder en deux.
— mixed : un mélange d'actions des deux motifs précédents.

L'évolution a lieu selon un nombre de pas de temps τ paramétré. Il faut noter que la taille totale de la séquence générée peut rapidement grossir puisqu'elle est proportionnelle au nombre de pas de temps. Ainsi un graphe de seulement 100 sommets de degré moyen 4 représente, en terme de nombre moyen d'arêtes, 2000 arêtes pour 5 pas de temps, 12000

arêtes pour 30 pas de temps et 40000 arêtes pour 100 pas de temps. Le temps de génération et la taille du fichier généré augmentent en conséquence.

Granell et al. (2015) présentent également des expériences dans leur article et mettent en avant le principal avantage de l'algorithme : les motifs proposés sont bien présents, cohérents et détectables par une méthode de détection communauté (les auteurs ont choisi d'illustrer avec l'algorithme de Mucha et al. (2010)).

B.2 Graphes tirés de situations et réseaux réels

Si les graphes générés artificiellement constituent la première source de données d'évaluation d'algorithmes de détection de communautés, ils ne sauraient constituer la seule. En effet, la génération est opérée depuis un modèle, certes paramétrables, mais défini et immuable. Ce modèle tente de capturer au mieux les phénomènes se déroulant dans les réseaux, mais des processus latents, cachés, sont à l'heure encore mal connus, ignorés ou tout simplement trop compliqués à reproduire et implémenter. Il est donc tout à fait possible de créer des algorithmes optimisés montrant d'excellentes performances sur les benchmarks artificiels, car capturant les propriétés et variables contenues dans ces benchmarks, mais beaucoup moins efficaces sur des graphes tirés de situations et réseaux réels. C'est pourquoi ces graphes sont un complément indispensable pour l'évaluation des algorithmes.

Nous présentons ici deux graphes tirés de réseaux sociaux réels, tout d'abord un graphe ancien et très célèbre contenant des communautés disjointes : le club de karaté de Zachary, puis un graphe plus récent contenant des communautés recouvrantes, tiré d'un relevé d'amitiés entre collégiens.

B.2.1 Club de karaté de Zachary

Ce graphe représente une situation vécue et décrite par Zachary (1977), qui l'a analysée d'un point de vue sociologique. Il s'agit du résultat de l'évolution des relations interpersonnelles entre les 34 membres d'un club de karaté, étudiées par Wayne Zachary pendant trois ans entre 1970 et 1972. Plus précisément, Zachary a étudié les interactions ayant eu lieu hors du contexte du club.

Durant la période de l'étude, un conflit a eu lieu entre le président/gérant du club (représenté par le sommet 34 dans le graphe) et le professeur de karaté (sommet 1). Ce conflit mena à la scission du club : la moitié environ des membres partirent former un nouveau club autour du professeur, les autres restèrent et trouvèrent un nouveau professeur, ou abandonnèrent le karaté. Zachary documente la décision de chacun des membres, et on peut donc considérer que deux communautés sont présentes : les membres faisant partie du nouveau club, et les autres (restant dans l'ancien club et abandonnant).

Le graphe résultant de ce réseau social comporte 34 sommets et 78 arêtes, chacune indiquant que les deux membres ont substantiellement interagit durant la période scrutée.

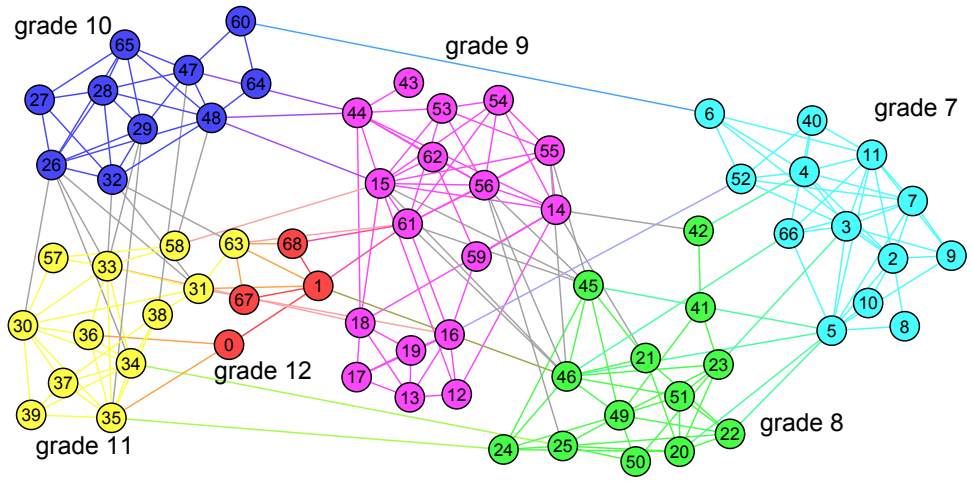


FIGURE B.1 – Réseau d'amitiés entre collégiens, figure tirée de (Xie et al., 2013)

Dans son article, Zachary opère une coupe max-flow–min-cut à l'aide de l'algorithme de Ford-Fulkerson (Ford, 1956), et prédit correctement le comportement de chaque membre, mis à part le membre identifié par le sommet 9.

Les algorithmes de détection de communautés poursuivent le même but : prédire correctement des communautés correspondant aux deux groupes présents après la scission, éventuellement plus précisément en se référant à des détails donnés par Zachary dans son article.

Le postulat assez fort derrière ce graphe est bien sûr que les interactions relevées sont suffisantes pour prédire le comportement d'un membre, i.e. son appartenance à telle communauté après la scission, ce qui est loin d'être évident. De plus, étant donné le très faible nombre de sommets du graphe, même un petit nombre d'erreurs de classification fait chuter des métriques comme l'ARI et la NMI de mesure significative.

Il est donc plus intéressant d'utiliser ce benchmark comme une évaluation visuelle et rapidement, « décorticable » des résultats d'un algorithme, que pour une évaluation rigoureuse de la qualité de l'algorithme.

B.2.2 Amitiés entre collégiens

Ce graphe est tiré d'une étude longitudinale [20] menée aux Etats-Unis par le National Institute of Child Health and Human Development sur l'évolution de la santé d'un groupe de personne, commencée alors que ces personnes étaient au collège et toujours en cours (Carrier, 2017). Cette étude concernent de nombreux paramètres de santé physique et mentale, et a donné lieu à de nombreuses publications [21].

Le graphe proposé (Fig. B.1), utilisé également par Xie et al. (2013), comporte 69 sommets et 440 arêtes, le degré moyen est 6.4. Il a été constitué en utilisant une partie

20. Qui suit une population donnée à long terme
21. `http://www.cpc.unc.edu/projects/addhealth/publications/database`

des questionnaires retournés remplis et retournés par les élèves participant eux-mêmes lors de la première phase de l'étude (1994-95). Chaque sommet représente un élève et chaque arête un lien d'amitié substantielle entre deux élèves, déterminé d'après les réponses aux questionnaires : contacts fréquents, visites de l'un chez l'autre etc.

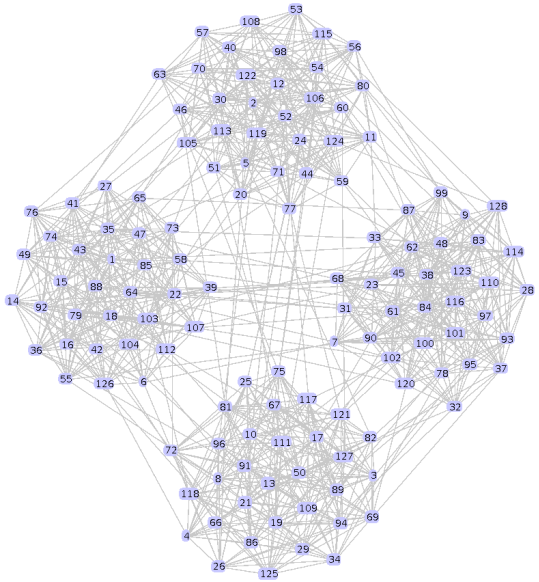
Xie et al. (2013) considèrent comme la « vérité terrain » le partitionnement en classes scolaires : la communauté de chaque élève est représentée par la classe du collège (grade 7-12) à laquelle il appartient.

Ce graphe est utilisé pour la détection de communautés recouvrantes, néanmoins comme nous l'avons exposé dans le chapitre 4, la notion de recouvrement et de multi-appartenance n'est pas présente de façon innée dans les graphes de terrain. Les auteurs proposent donc une liste de sommets recouvrants « évidents », tels que des sommets possédant de fortes connexions dans un autre groupe, en plus du leurs : les sommets 26, 32, 33, 45, 46 et 61. Ils notent également que le sommet 42 n'a pas d'appartenance particulière à une classe mais constitue un « pont » (bridge) entre les classes 7, 8, et 9. Cette situation peut être expliquée par l'abstraction du graphe : il ne représente que partiellement les résultats de l'enquête analysés.

B.3 Instances de graphes

Enfin, nous présentons deux instances de graphes utilisées pour des expériences du chapitre 7 : G_{test} et G_{cc}.

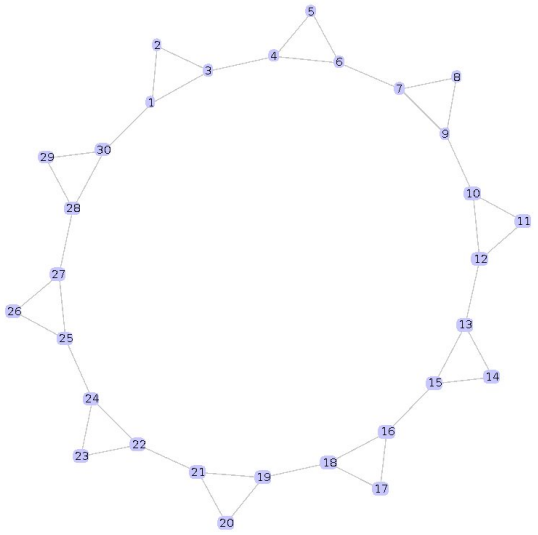
Graphe G_{test}



G_{test} est généré par LFR. Il ne comporte pas de communautés recouvrantes.

Paramètres de G_{test}	Nombre de sommets	128
	Nombre de paires de sommets non-voisins	14208
	Degré moyen	16
	Degré max	16
	Nombre de communautés	4
	Nombre min de sommets par communauté	32
	Nombre max de sommets par communauté	32
	Paramètre de mixage	0.1

Graphe G_{cc}



G_{cc} (*cc* comme *cliques chain*) est une suite de 10 3-cliques reliées les unes aux autres.

Paramètres de G_{cc}	Nombre de sommets	30
	Degré moyen	2.67
	Degré max	3
	Nombre de communautés	10
	Nombre min de sommets par communauté	3
	Nombre max de sommets par communauté	3
	Paramètre de mixage	-

Annexe C

Eléments théoriques autour de la détection de communautés orientée-sommet

Sommaire

Cette annexe présente des notions sortant du cadre de la thèse mais néanmoins utiles pour détailler davantage le fonctionnement des méthodes orientées sommet. Nous présentons tout d'abord des éléments de théorie des graphes liés à la connectivité, étendant les notions présentées dans le chapitre 1, p. 6. Nous discutons ensuite des marches aléatoires locales sur graphe, à la base de certaines méthodes de partitionnement de graphe orientées sommets, puis nous présentons les résultats de Gleich & Seshadhri, formalisant les communautés trouvées par ce biais.

C.1 Connectivité et graphes expanseurs

La *connectivité d'un graphe* est très liée à la conductance et au nombre isopérimétrique discutés dans la section 1.1.4, p. 9. Un modèle de graphe particulier disposant d'une forte connectivité est celui des graphes expanseurs (*expander graphs*, (Hoory et al., 2006)), dont nous discutons ensuite. Nous ne rentrons pas dans les détails de la théorie élaborée autour de ces graphes ni les propriétés liées car cela dépasse le cadre des travaux présentés dans la thèse.

Connectivité

La notion de connectivité d'un graphe, très riche, peut être introduite de façon simple :

Définition La *connectivité des sommets* (resp. arêtes) *d'un graphe* est le nombre de sommets (resp. arêtes) à supprimer de ce graphe afin de le rendre non-connexe.

En particulier, dans un contexte de partitionnement ou de structure communautaire, la connectivité d'un sous-ensemble S_G du graphe vers un autre ${S'}_G$ est le nombre d'arêtes de S_G reliant des sommets de S_G à des sommets de ${S'}_G$.

Plus la connectivité d'un graphe est haute, plus il existe de petits nombres de ses sommets reliés à un grand nombre d'autres sommets. Cette propriété est utile pour mesurer, par exemple, la circulation de l'information dans un réseau, afin d'anticiper si l'information ayant pour origine un sous-graphe pourrait facilement « en sortir » et se diffuser dans le reste du graphe, ce qui est plus susceptible de se produire si le graphe possède une bonne connectivité.

Un sous-graphe possédant un grand nombre isopérimétrique signifie qu'il dispose de beaucoup d'arêtes vers le reste du graphe : il possède donc une très bonne connectivité mais ne formerait probablement pas une bonne communauté (car intuitivement une communauté a peu de liens vers le reste du graphe). Au contraire, on cherche plutôt à obtenir des sous-graphes de nombre isopérimétrique minimum car ils témoignent de la présence de « goulets d'étranglement », peu d'arêtes les relient au reste du graphe.

Graphes expanseurs

Un *graphe expanseur* est, de façon simple, un graphe particulier, épars (peu dense), montrant une une forte connectivité [22]. On détermine généralement cette connectivité par le *taux d'expansion* d'un tel graphe, qui est son nombre isopérimétrique. Plus le taux d'expansion est élevé, plus le graphe comporte de petits sous-ensembles de sommets possédant de nombreuses connexions avec le reste du graphe.

Les graphes expanseurs sont très utilisés pour la constructions de bons codes correcteurs d'erreur par exemple ainsi que de réseaux diverses dont on veut assurer une bonne connectivité. En effet, de par leurs propriétés, les graphes expanseurs fournissent des garanties théoriques quant à la convergence de marches aléatoires (vers des distributions stationnaires), généralement très rapides. Pour plus de détails sur les graphes expanseurs, nous renvoyons à l'article d'état de l'art très complet de Hoory et al. (2006).

22. On requiert généralement que le graphe soit épars afin de ne pas considéré de situations triviale, en effet un graphe complet, par exemple, a par définition une très forte connectivité, mais ce n'est souvent pas un cas que l'on souhaite considérer.

C.2 Graphes de terrain, voisinages de sommets et conductance

Résultats de Gleich & Seshadhri

Les quelques résultats mathématiques relatifs aux valeurs de conductance atteignables par les algorithmes de détection de communautés proposés dans le chapitre 2 (section 2.5.2, p. 58 plus particulièrement) sont proposés pour n'importe quel graphe. Or, on pourrait espérer que les propriétés particulières des graphes de terrain permettent d'aller plus loin.

Nous présentons ici des travaux de David Gleich et Seshadhri Comandur, datant de 2011 (publiés à ACM SIGKDD 2012), reliant des propriétés d'un graphe à l'existence de coupes de faible conductance au voisinage des sommets. Ils sont très importants pour justifier le fonctionnement des algorithmes proposés dans cette thèse. Les preuves de ces théorèmes sont disponibles dans l'article.

Nous utilisons ici un graphe $G = (V, E)$, les notations usuelles introduites en début de manuscrit, ainsi que les notations suivantes :

$S \subset V$ un sous-ensemble de sommets du graphe,

∂S la coupe autour des sommets S (cf. section 1.1.4, p. 9),

$\kappa \in [0, 1]$ le coefficient de clustering global d'un graphe G (cf. section 1.2.3.3),

W l'ensemble des triplets non connectés de G,

W_v l'ensemble des triplets non connectés centrés sur le sommet v [23], et $|W_v| = \binom{d_v}{2}$

$p_v = \frac{|W_v|}{|W|}$,

f_d le nombre de sommets de degré d

Un premier résultat intéressant montre que, en moyenne, la conductance d'une coupe autour d'un sous-ensemble de sommets d'un graphe est liée à son coefficient de clustering global : plus le coefficient est élevé, plus la cette conductance est faible.

Lemme C.2.1 (Conductance des coupes de voisinage)**.**

$$\sum_v \left(p_v \frac{|\partial\Gamma(v)|}{|W_v|} \right) = \frac{\sum_v |\partial\Gamma(v)|}{|W|} = 2(1 - \kappa)$$

Ainsi, un haut coefficient de clustering global κ entraîne que la somme des conductance des coupes aux voisinages de tous les sommets du graphe sera basse. Il en résulte que l'espérance de la conductance d'une coupe au voisinage d'un sommet particulier est basse. Le théorème 2.5.1, p. 57 établit l'existence et borne la valeur de conductance d'une telle coupe.

Un autre théorème, que nous ne reproduisons pas ici, montre également que sous la même condition de distribution des degrés, il existe de larges cœurs dans le graphe.

23. C'est-à-dire, les ensembles de 3 sommets u, v, w tels qu'il existe une arête (u, v), une arête (v, w) mais pas d'arête (u, w)

Enfin, il est important de de noter que les bornes théoriques présentées sont *faibles*, car elles expriment un *pire cas* : la valeur prouvée de la conductance atteignable (théorème ci-dessus) n'est en pratique pas très intéressante pour obtenir des communautés structurellement intéressantes. Néanmoins, c'est une première garantie théorique et, de même que ce que l'on observe généralement en détection de communauté, les valeurs typiquement atteintes sont bien meilleures (i.e. plus faibles en ce qui concerne la conductance).

Tout en constituant une progression importante dans la compréhension des ressorts théoriques et mathématiques de la détection de communautés, ces travaux continuent également d'en illustrer l'une des difficultés majeures : la performance des algorithmes sur ce problème, pourtant au moins NP-difficile, souvent bien plus efficaces que les bornes théoriques des pires cas, reste difficile à comprendre et à expliquer.

Résumé

Les travaux présentés dans la thèse s'inscrivent dans le cadre de l'analyse des graphes de terrain (*complex networks*) et plus précisément de la tâche de détection de communautés, c'est-à-dire la reconnaissance algorithmique de sous-graphes particulièrement denses. Nous nous intéressons spécifiquement à l'implémentation d'une telle méthode dans un contexte fortement décentralisé et distribué : des réseaux MANET opportunistes formés par de petits objets connectés communiquant en pair-à-pair.

Afin de tenir compte des contraintes d'exécution d'algorithme dans de tels réseaux, les travaux présentés dans la thèse proposent des méthodes conçues selon le paradigme récent et actif nommé *orienté sommet*, en alliant le traitement de graphes Think-Like-a-Vertex aux méthodes de détection de communautés basées sur des leaders ou des graines : celles-ci présentent en effet des propriétés de décentralisation qui autorisent des implémentations parallèles et distribuées appropriées au cadre applicatif considéré.

Dans ce contexte, nous proposons d'une part un principe global de fonctionnement original que nous mettons en œuvre et déclinons dans trois algorithmes dédiés à trois configurations différentes de la tâche de détection de communautés : l'algorithme VOLCAN considère le cas de référence des communautés disjointes dans un graphe statique. Nous l'étendons ensuite avec l'algorithme LOCNeSs au cas des communautés recouvrantes, qui autorisent un sommet à appartenir à plusieurs communautés simultanément : cette généralisation donne plus de flexibilité à la détection et la rend plus appropriée au cadre applicatif considéré. Nous examinons également le cas des graphes dynamiques, c'est-à-dire dont les sommets et les arêtes évoluent au cours du temps, auquel est consacré l'algorithme DynLOCNeSs. Chacun des algorithmes est associé à une implémentation décentralisée et fait l'objet d'une étude théorique ainsi qu'expérimentale sur des données artificielles et réelles permettant d'évaluer la qualité des résultats fournis et de les comparer aux méthodes de l'état de l'art.

Nous considérons également, dans un cas particulier de réseau mobile ad-hoc spontané et décentralisé issu d'une application réelle de vêtements intelligents et communicants, une tâche de cheminement permettant d'identifier des interlocuteurs. Nous proposons une stratégie de recommandation utilisant la structure communautaire, modélisée et évaluée à travers un algorithme nommé SWAGG.

Mots-clés: fouille de graphes, détection de communautés, graphe dynamique, communautés recouvrantes, méthode locale, méthode décentralisée, méthode orientée sommet, think-like-a-vertex

Abstract

Our research is in the field of complex network analysis and mining, specifically addressing the communit detection task, ie. algorithms aiming to uncover particularly dense subgraphs.

We focus on the implementation of such an algorithm in a decentralised and distributed context : opportunistic MANET constituted of small wireless devices using peer-to-peer communication.

To tackle the implementation constraints in such networks, we propose several methods designed according to the novel and trending vertex-centred paradigm, by combining Think-Like-a-Vertex graph processing with vertex-centred community detection methods based on leaders or seeds : they show specific properties allowing dsitributed implementations suiting the opportunistic MANET case.

In this context, we first a global working principle and implement it in three different algorithms dedicated to three different configurations of community detection : the VOLCAN algorithm manages the classical disjoint community detection task in a static graph.

We extend it with the LOCNeSs algorithm, that is dealing with overlapping communities which means that one vertex can belong to several communities. It adds more flexibility to the method and more significance to produced results. We also tackle the dynamic graphe case (graph evolving over time), addressed by the DynLOCNeSs algorithm.

Each algorithm comes with a decentralised implementation and theoretical as well as experimental studies conducted both on real and synthetic benchmark data, allowing to evaluate the quality of the results and compare to existing state-of-the-art methods.

Finally, we consider a special case of opportunistic decentralised MANET developped as a part of a research project about smart and communicating clothing. We formalise a task of path finding between smart t-shirts holders and propose a recommandation strategy using community structure, that we model and evaluate through an algorithm named SWAGG.