

UNIVERSITY NAME

DOCTORAL THESIS

Conversations, Groupes et Communautés dans les Flots de Liens

Auteur :
Noé GAUMONT

Directeurs :
Clémence MAGNIEN Matthieu
LATAPY

*A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy*

in the

Research Group Name
Department or School Name

30 juin 2016

Table des matières

Remerciements	ii
Introduction	1
1 État de l'art sur la détection de communautés et les réseaux dynamiques	3
1.1 Définition dans les graphes	4
1.1.1 Groupes, partitions et couvertures	4
1.1.2 Comparaison de partitions et couvertures	5
1.2 Communauté dans les graphes	7
1.2.1 Parititons de nœuds	7
Méthodes utilisant un modèle pour la comparaison	7
Méthodes utilisant des marches aléatoires	9
1.2.2 Couverture de nœuds	10
Extension de méthodes existantes	11
Méthodes utilisant des fonctions de qualité locales	12
1.3 Extension temporelle des graphes	13
1.3.1 Extensions avec pertes d'informations temporelles	14
Séries de graphes	14
Tenseur 3D	15
Graphes multicouche	15
Bilan	16
1.3.2 Extension sans pertes d'informations temporelles	16
Graphe temporel	16
Flots de liens	17
2 Flots de liens : extensions temporelle des graphes	19
2.1 Définition	19
2.2 Sous-flot induit	19
2.3 Degré et densité	19
Liste des notations pour les flots de liens	21
3 Étude d'une archive de courriels	23
3.1 Prétraitement sur le jeu de données	23
3.2 Caractéristiques élémentaires des discussions	24
3.3 Étude des discussions en tant que sous-flots	26
3.3.1 Application de la Δ -densité	26
3.3.2 Répartition temporelle et structurelle des discussions	29
3.3.3 Flot quotient	31
3.3.4 Conclusion	32
3.4 Détection de structures denses	33
3.4.1 Méthode de détection	33
3.4.2 Comparaison des partitions	34
3.4.3 Conclusion	36

4 Détection de groupes denses (SNAM)	39
4.1 Calcul des groupes candidats	39
4.2 Calcul évaluation	39
4.3 Jeux de données	39
4.4 Application	39
4.5 Conclusion	39
5 Expected Nodes : communautés de liens dans les graphes statiques	41
5.1 Travaux existants	42
5.2 Définition d'Expected Nodes	45
5.3 Comparaison	48
5.3.1 Cas du graphe complet	48
5.3.2 Graphe LFR	49
5.4 Calcul et optimisation	52
5.5 Conclusion	53
5.5.1 Perspective	54
Amélioration de l'optimisation d' <i>Expected Nodes</i>	54
Cas d'utilisation d' <i>Expected Nodes</i>	55
6 Fonction de qualité	57
6.1 Définition	57
6.2 Générateur de flots de liens avec structure communautaire	57
7 Conclusion	59
Bibliographie	60

Introduction

Appuyer sur le fait que le temps est continu et que l'on veut garder toute l'information pour faire des descriptions super fines temporellement.

Chapitre 1

État de l'art sur la détection de communautés et les réseaux dynamiques

Sommaire

1.1	Définition dans les graphes	4
1.1.1	Groupes, partitions et couvertures	4
1.1.2	Comparaison de partitions et couvertures	5
1.2	Communauté dans les graphes	7
1.2.1	Parititons de nœuds	7
1.2.2	Couverture de nœuds	10
1.3	Extension temporelle des graphes	13
1.3.1	Extensions avec pertes d'informations temporelles	14
1.3.2	Extension sans pertes d'informations temporelles	16

Dans cette thèse, nous étudions deux axes de recherches liés aux graphes qui sont orthogonaux. D'une part, il s'agit de la détection de structures dans les graphes et plus particulièrement de communautés. Un communauté est un sous-ensemble de nœuds de manière à ce qu'ils soient fortement connectés. Il n'existe cependant aucune définition exact et la notion de communauté fortement connectée dépend du contexte et de la méthode. Malgré cette définition floue, des structures communautaires ont été trouvées dans de nombreux graphes dans plusieurs domaines tel que le réseaux constitué des région du cerveau [dRSKvdH14], le réseau de distribution d'eau [DDG⁺15] et un réseau d'interactions d'animaux [FW15]. Ces notions de communautés et les méthodes de détection sont définies dans la section 1.2.

D'autre part, il a été observé que la modélisation d'un réseau sous la forme d'un graphe peut poser problème notamment quand le réseau change au cours du temps [Hol15b]. D'une part, l'information temporelle est complètement perdue lorsque l'on manipule un graphe. Par conséquent, certaines structures peuvent devenir indétectable. Imaginons un graphe représentant les alliances politiques dans un pays. Si toutes les alliances politiques sont agrégées sur une trop grande durée temporelle, alors une personne politique changeant de parti politique sera faussement considérée comme influente car connectée à plusieurs partis alors qu'elle peut ne plus avoir de contact dans son ancien parti. Ainsi il n'est plus possible d'observer le glissement des alliances politiques dans ce genre de réseau si l'agrégation temporelle est trop importante [MRM⁺10].

D'autre part, si on prends en compte le temps, alors il est possible de détecter des structures plus fines que dans le cas statique. Il devient possible de détecter des instant où l'organisation

générale change [RB10], de comprendre à quels sont les instant où une personnes est importante [MT15], ou bien de détecter des groupes temporelles [CAH10]. Face à ces problèmes plusieurs extensions de la théorie des graphes ont été proposées et elles sont résumées dans la section 1.3.

Mais avant de présenter ces deux axes de recherches, nous revenons dans la section 1.1 sur quelques définitions formelles et notation des graphes que nous utilisons dans cette thèse.

1.1 Définition dans les graphes

Un graphe G est défini par un couple (V, E) où V est une ensemble de nœuds et E un ensemble de liens chaque lien est une paire de nœuds. Sauf mention contraire, nous considérons des graphes non-orienté, c'est-à-dire pour toute pair de nœuds $u, v \in V$ les liens (u, v) et (v, u) sont équivalents. Nous considérons également uniquement des liens non-pondérés, c'est-à-dire qu'un lien est soit présent soit absent. Enfin, nous ne considérons que des graphes simples, c'est-à-dire qu'il n'existe au maximum qu'un seul lien entre deux nœuds. Ceci est en opposition avec les graphes multiples où il peut exister plusieurs liens entre deux nœuds.

Par convention, nous définissons $n = |V|$ le nombre de noeuds et $m = |E|$ le nombre de liens. Il est possible de représenter un graphe G par une matrice carré A de taille n où la case $A_{i,j} \forall i, j \in \mathbb{N}^+ \leq n$ est égale à 1 si un lien relie les nœuds i et j et 0 sinon.

Degré Le degré d'un nœud i , noté d_i , est égale au nombre de liens reliés à i : $d_u = \sum_{j \leq n} A_{i,j}$.

Densité La densité, $d(V)$ des nœuds d'un graphe est la probabilité que deux nœuds soient

$$\text{reliés par un lien} : d(V) = \frac{2m}{n(n-1)}.$$

Degré moyen Le degré moyen $\tilde{d}(V)$ diffère de la densité est égale à : $d(\tilde{V}) = \frac{2m}{n}$.

Chemin Une chaîne ou chemin entre deux nœuds dans un graphe est une suite de liens $((u_1, v_1), \dots, (u_k, v_k))$ de tel sorte que $v_i = u_{i+1} \forall i \in [0, k-1]$.

Graphe connexe Les nœuds d'un graphe sont dit connexe s'il existe un chemin entre toute les pairs de nœuds.

Composante Connexe Une composante connexe est un ensemble de nœuds $V' \subseteq V$ qui est connexe et maximal, c'est-à-dire qu'il n'est pas possible d'ajouter un nœud dans V' tel que V' soit toujours connexe.

Arbre Le graphe connexe le moins dense est un arbre et est constitué de $n - 1$ liens pour n nœuds.

Clique Le graphe le plus dense est appelé un graphe complet ou un clique et est composé de $\frac{n(n-1)}{2}$ pour n nœuds.

Sous-graphe Un graphe $G' = (V', E')$ est un sous-graphe de G si et seulement si $V' \subset V$ et $|E' \cap E| = |E'|$.

Graphe induit Le graphe induit par un $V' \subset V$ est définit par V' et $E' = \{(u, v) \in E, u, v \in V'\}$.

1.1.1 Groupes, partitions et couvertures

En plus des nœuds et des liens, nous manipulons également des ensembles de liens et des ensembles de nœuds. Par convention, soit $X \subset V$ un ensemble de nœuds et $Y \subset E$ un ensemble de liens. Nous listons les notations utilisées dans le tableau 1.1.

Notation	Définition
$V(Y) = \{u, \exists(u, v) \in Y\}$	nœuds induits par les liens de Y
$n_X = X $	nombre de nœuds dans X
$l_{in}(X) = \{(u, v) \in E, u, v \in X\} $	nombre de liens entre les nœuds de X
$l_{out}(X) = \{(u, v) \in E, \cap(X \times V \setminus X)\} $	nombre de liens entre les nœuds de X et de $V \setminus X$
$l(X) = l_{in}(X) + l_{out}(X)$	nombre de liens reliés aux nœuds de X
$d_{in}(u, X) = \{(u, v) \in E, v \in X\} $	nombre de lien que partagent u avec X
$d_{in}(X) = \sum_{u \in X} d_{in}(u, X) = 2l_{in}(X)$	somme des degrés internes des nœuds dans X
$d_{out}(u, X) = \{(u, v) \in E, v \in V \setminus X\} $	nombre de lien que partagent u avec $V \setminus X$
$d_{out}(X) = \sum_{u \in X} d_{out}(u, X) = l_{out}(X)$	somme des degrés externes des nœuds dans X
$d(X) = d_{in}(X) + d_{out}(X)$	somme des degrés des nœuds dans X

TABLE 1.1 – Liste des notations utilisées pour un ensemble de nœuds X et un ensemble de liens Y .

Partitions et couvertures Nous définissons ici des structures de partitions et de couvertures appliquée au cadre spécifique des graphes. Une partition de nœuds, \mathcal{V} , est ensemble d'ensemble de nœuds : $\mathcal{V} = \{V_1, \dots, V_k\}$ tel que $V_i \subseteq V \forall i \in [1, k]$. Une partition est soumise à deux contraintes :

1. La partition doit *recouvrir* l'ensemble des nœuds : $\bigcup_i (V_i) = V$.
2. Les ensembles de nœuds doivent être disjoint : $V_i \cap V_j = \emptyset \forall i, j \in [1, k]$.

Afin de manipuler une partition, nous définissons $\mathcal{V}(u) = V_i$ si et seulement si $u \in V_i$.

Une couverture est une extension des partitions car elle relâche la contrainte sur l'intersection. Une couverture est parfois aussi appelé partition chevauchante. Ainsi, une couverture de nœuds, \mathcal{V} , est aussi un ensemble d'ensemble de nœuds. L'union des ensembles doit également être égale à V mais en revanche deux ensembles de nœuds peuvent partager un nœud : $\exists i, j \in [1, k] V_i \cap V_j \neq \emptyset$.

Nous avons détaillé les notions de partitions et couvertures de nœuds mais les même définitions valent pour les partitions et couvertures de liens.

1.1.2 Comparaison de partitions et couvertures

Une fois définis les partitions et les couvertures, il est intéressant de pouvoir les comparer. Le but est de calculer une similarité entre deux structures de telles sorte que la similarité égale 1 si les deux structures sont identiques et qu'elle soit égale à 0 ou -1 si elles sont complètement différentes. Il existe pour ce faire les méthodes tirant parti de la structure d'ensemble et les méthodes provenant de la théorie de l'information.

Approche ensembliste Il est possible de comparer deux ensembles X et Y en utilisant l'indice de Jaccard : $\mathbb{J}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$. Avec l'indice de Jaccard, il est possible de mesurer la similarité entre deux partitions \mathcal{X} et \mathcal{Y} :

$$sim(\mathcal{X}, \mathcal{Y}) = \frac{1}{|\mathcal{X}|} \sum_{X \in \mathcal{X}} \max_{Y \in \mathcal{Y}} \mathbb{J}(X, Y). \quad (1.1)$$

Avec cette formulation, la similarité n'est pas symétrique. C'est pourquoi la formule suivante lui est souvent préférée :

$$\text{sim}_{\text{moy}}(\mathcal{X}, \mathcal{Y}) = \frac{\text{sim}(\mathcal{X}, \mathcal{Y}) + \text{sim}(\mathcal{Y}, \mathcal{X})}{2} \quad (1.2)$$

Il existe d'autres méthodes pour symétriser la similarité. Une méthode possible est d'utiliser la moyenne harmonique. Cette méthode peut s'appliquer indifféremment aux partitions et aux couvertures.

Il existe également le Rand Index [Ran71] qui ne s'applique qu'aux partitions. Il mesure le nombre de pairs de nœuds qui sont classées de la même manière dans les deux partitions ; c'est à dire si pour deux nœuds u et v soit $\mathcal{X}(u) = \mathcal{X}(v)$ et $\mathcal{Y}(u) = \mathcal{Y}(v)$ soit $\mathcal{X}(u) \neq \mathcal{X}(v)$ et $\mathcal{Y}(u) \neq \mathcal{Y}(v)$. Plus formellement, soient a_{11} le nombre de pair de nœuds de telle sorte qu'ils soient dans le même ensemble dans les deux partitions, a_{00} le nombre de pair de nœuds de telle sorte qu'ils soient dans des ensembles différents dans les deux partitions et a_{10} (resp. a_{01}) le nombre de pair de nœuds de telle sorte qu'ils soient dans le même ensemble dans \mathcal{X} (resp. \mathcal{Y}) et dans deux ensembles différents dans \mathcal{Y} (resp. \mathcal{X}). Avec ces notations, le Rand Index est défini de la manière suivante :

$$RI(\mathcal{X}, \mathcal{Y}) = \frac{a_{11} + a_{00}}{a_{11} + a_{01} + a_{10} + a_{00}} \quad (1.3)$$

Il se peut que, par chance, deux partitions classent deux nœuds de la même manière. C'est pourquoi une version ajustée du Rand Index (ARI) a été proposée une version [HA85]. Le Rand index et sa version ajustée permettent de comparer des partitions. Une extension de l'ARI a été Porumbel *et al.* pour comparer des couvertures. Il s'agit de l'Omega Index [PHK11]

Approche venant de la théorie de l'information On peut considérer que l'assignation d'un élément à un ensemble est une variable aléatoire. Dans ce cas, la probabilité d'un élément d'être dans un ensemble $X \in \mathcal{X}$ est $P(X) = \frac{|X|}{|V|}$. De manière similaire, la probabilité jointe est $P(X, Y) = \frac{|X \cap Y|}{|V|}$. Avec ces définitions, il est possible de calculer l'*entropie* d'une partition, $H(\mathcal{X})$, l'*entropie conditionnelle*, $H(\mathcal{X}|\mathcal{Y})$ et l'*information mutuel* $I(\mathcal{X}, \mathcal{Y})$. Cette dernière est définie par $I(\mathcal{X}, \mathcal{Y}) = H(\mathcal{X}) - H(\mathcal{X}|\mathcal{Y})$. L'*entropie de Shanon* est définie par $H(\mathcal{X}) = -\sum_{X \in \mathcal{X}} P(X) \log(P(X))$ et l'*entropie conditionnelle* par $H(\mathcal{X}|\mathcal{Y}) = -\sum_{X \in \mathcal{X}, Y \in \mathcal{Y}} P(X, Y) \log P(X|Y)$. Afin de normaliser l'*information mutuel*, Danon *et al.* ont défini l'*information mutuel normalisée* (NMI_{shanon}) :

$$NMI_{shanon}(\mathcal{X}, \mathcal{Y}) = \frac{2I(\mathcal{X}, \mathcal{Y})}{H(\mathcal{X}) + H(\mathcal{Y})} \quad (1.4)$$

Lancichinetti *et al.* l'ont par la suite étendue pour prendre en compte des couvertures [LF09b]. Le choix de la normalisation dans le cas chevauchant semble cependant toujours ouvert [aMGH11, Zha15].

Résumé

Il est intéressant de noter que la littérature sur le problème de la comparaison de structure est assez restreinte comparé à la détection de communautés. En effet, il semble qu'uniquement 3 similarités soient utilisées : similarité se basant sur Jaccard, l'*Omega index* et la *NMI*. Surprenamment, tous les indices de similarité ont été étendus aux couvertures de manière assez convaincantes. Il est également important de noter l'*existence d'implémentations librement accessible* d'une majeure parties de ces métriques ^{a b}.

a. <https://github.com/aaronmcdaid/Overlapping-NMI>

b. <http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation>

1.2 Communauté dans les graphes

Ce champs de recherche est très vaste et il est illusoire de vouloir énumérer les méthodes existantes dans ce domaine car les caractéristiques voulues d'une communauté peuvent varier selon le contexte [CGP11, LLDM08, YL15, JBP⁺15]. Il y a tout de même deux grandes catégories qui permettent de séparer les méthodes existantes. Dans ces deux catégories, les méthodes existantes cherchent à capturer des communautés fortement connectées mais elles diffèrent sur ce qu'elles capturent comme structures communautaires. Les communautés d'une structure communautaire peuvent être disjointes, c'est-à-dire que deux communautés n'ont aucun noeuds en commun, ou alors recouvrantes, c'est-à-dire que deux communautés peuvent avoir plusieurs noeuds en commun. Dans le premier cas, on parle de partition des noeuds et dans le second on parle de couverture ou partitions chevauchantes de noeuds. Dans ces deux cas, il y a également une contrainte sur le fait que tous les noeuds doivent appartenir à au moins une communauté. Ces deux structures correspondent à deux visions possibles de l'organisation d'un graphe et du réseau sous-jacent. Nous présentons ces deux catégories dans les sous-sections suivantes. Il existe également une troisième catégorie qui est la détection de partition de liens qui est un problème à part et qui est présentée dans le chapitre 5.

1.2.1 Paritions de noeuds

Afin de mieux comprendre ce que peut capturer une partition de noeuds, il est plus facile de partir d'un exemple. Dans l'étude de Stehlé *et al.* [SVB⁺11], des enfants d'une école primaire ont eu pendant 2 jours des capteurs enregistrant lorsque deux enfants sont à une distance de moins de 1m50. Ce dispositif permet de mesurer les interactions entre élèves et de construire le graphe des relations entre élève à l'école. Un lien existe entre deux élèves si ils ont interagi au moins une fois ensemble. Une illustration du graphe obtenu est visible dans la figure 1.1. La classe de chaque élève est également connue. Comme chaque élève appartient à une et une seule classe, les classes forment une partition des élèves. Cette partition est une bonne structure communautaire car on remarque que les élèves d'une même classe parlent beaucoup entre eux mais ils parlent peu entre élèves de classes différentes. Cela se remarque particulièrement bien pour la classe 3A. Il existe beaucoup de liens entre les élèves de la classe 3A et aucun entre eux et les élèves de la classe 5A par exemple.

Afin de capturer des partitions de noeuds, beaucoup de méthodes existent. Il y a d'ailleurs régulièrement des état de l'art qui sont publiés [For10, PC13, MV13, HBG14]. Afin d'aider le lecteur, nous détaillons ici quelques unes des méthodes les plus utilisées.

Méthodes utilisant un modèle pour la comparaison

Comme une communauté est souvent définie comme devant être très connectée, le problème est de trouver une fonction capable d'évaluer la connectivité d'une communauté. Pour ce faire, deux *ingrédients* sont nécessaires. Tout d'abord, il faut définir une métrique qui mesure la connectivité, *e.g.* le nombre de liens dans un groupe. Puis, il est nécessaire de définir comment utiliser la métrique pour considérer qu'un groupe est très connecté. Il est possible d'utiliser une métrique en la normalisant par ses bornes minimum et maximum. Avec cette métrique normalisée entre 0 et 1, un groupe est considéré comme très connecté si son évaluation est supérieur à un certain seuil. Cette approche n'est cependant pas adaptée pour la recherche de communauté car elle ne tient pas compte de la structure du graphe². Prenons l'exemple du graphe constitué d'une clique et du nombre de liens comme métrique. Le nombre de lien d'un

1. Image provenant de <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0023176>.

2. C'est plus approprié dans la recherche des groupes les plus denses [BBC⁺15].

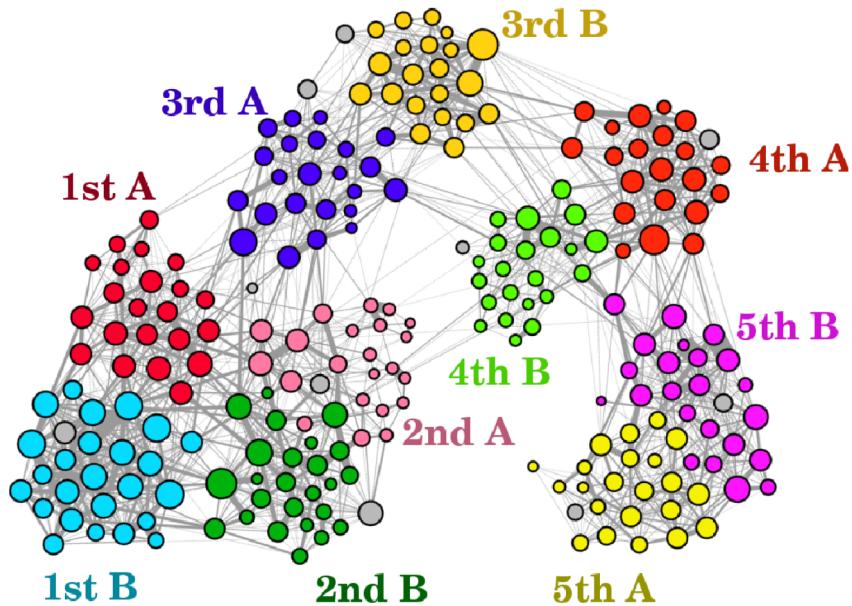


FIGURE 1.1 – Graphe de contact des enfants d'une école primaire. L'épaisseur du lien représente la durée de communication entre deux élèves. La couleur représente la classe de chaque élèves. Les professeurs sont en gris.¹

groupe est normalisé par le nombre minimum de liens, 0, et par le nombre de maximum de liens qui est obtenu par une clique. Dans ce graphe, tout groupe de nœuds est également une clique et par conséquent tout groupe de nœuds a un évaluation parfaite de 1. Chaque groupe serait donc une très bonne communauté selon cette évaluation. Or, le graphe constitué d'une unique clique ne possède pas de structure communautaire contrairement au graphe dans la figure 1.1. Il est donc nécessaire de trouver une autre approche.

Plutôt que de normaliser une métrique par ces valeurs minimum et maximum, il est intéressant de considérer l'écart à une valeur moyenne. L'idée est la suivante : si le graphe n'avait pas de structure communautaire qu'elle serait la valeur attendue de la métrique considérée ? Le problème est alors de définir des graphes qui n'ont pas de structures communautaires. L'ensemble des graphes similaires définit alors un *modèle nul*. Déetecter des communautés se traduit alors en trouver des groupes qui s'éloignent du modèle nul considéré où il n'existe pas de communauté. Ce changement est astucieux car il est relativement aisé de définir des graphes n'ayant pas de structure. Il suffit de créer des graphes complètement aléatoires [ER59] où les liens du graphes sont tirés de manière uniforme. Afin que les graphes aléatoires puissent être comparable au graphe initial, des contraintes sont généralement ajoutées et donnent lieu à différent modèle nul. Le premier modèle nul de graphe est celui de Erdős-Rényi [ER59] où le nombre de liens et le nombre de nœuds des graphes aléatoires doivent être les mêmes que dans le graphe initial. Un autre modèle très couramment utilisé est le modèle de configuration [BC78]. Dans ce modèle, la distribution des degrés est également fixe. Le modèle est de configuration est souvent utilisé car il a été souvent observé que les graphes provenant de données réelles ont une distribution des degrés très éloignés d'une distribution uniforme. C'est pourquoi l'ajout de la contrainte sur les degrés permet de considérer des graphes dans le modèle nul plus similaire au graphe initial. Ils existent bien évidemment d'autres modèles possibles considérant d'autres contraintes [New09].

Modularité La modularité [NG04] est une fonction qui associe à chaque partition de nœuds une valeur de qualité entre -1 et 1 . Plus la valeur de modularité d'une partition est élevée, plus

la partition est censée capturer une structure communautaire. La modularité est définie de la manière suivante pour une partition \mathcal{C} :

$$Q(\mathcal{C}) = \frac{1}{2M} \sum_{i,j \in V} \left(A_{ij} - \frac{d_i d_j}{2M} \right) \delta_{C(i)=C(j)}. \quad (1.5)$$

Il s'agit pour deux nœuds d'une même communauté de comparer la présence ou absence d'un lien, A_{ij} , à la probabilité que ces deux nœuds soient reliés dans le modèle de configuration, $\frac{d_i d_j}{2M}$. L'idée sous-jacente est que les nœuds d'une communauté devraient partager plus de liens qu'espéré dans le modèle de configuration. De très nombreux travaux ont par la suite étudié les caractéristiques de la modularité et son optimisation. Tout d'abords, il a été montré que l'optimisation de la modularité est un problème NB-Complet [BDG⁺07]. Il est donc nécessaire de recourir à des heuristiques afin de trouver rapidement une partition proche de l'optimum. Parmi l'ensemble des algorithmes existant, l'algorithme de Louvain [BGLL08] est un des plus rapide. Il existe également des variantes de cet algorithme [HBP15, Tra15]. D'autres travaux se sont attachés à l'étude de la modularité. Il a été montré que la modularité souffre du problème de *réolution limite* [FB07, LF11] car le modèle de configuration presuppose une répartition uniforme des tailles des communautés. Il a par ailleurs été montré que la modularité n'offre pas de maximum clair et que beaucoup de partitions différentes ont des évaluations proches [GdMC10]. Pour répondre à ces problèmes, il existe des variantes [RB06, DYB10].

Surprise La fonction Surprise [AM11, TAD15] est une autre fonction de qualité qui se base quant à elle sur le modèle de Erdös-Rényi pour évaluer la surprise d'observer un groupe de nœuds relié par l liens.

Stochastic Block Model Nous avons définis précédemment la notion de modèle nul permettant de se comparer à l'absence de communautés. Il est également possible de modéliser une structure communautaire puis de vérifier *a posteriori* si ce modèle pourrait être à l'origine du graphe observé. Le problème de détection de communauté est alors un problème d'inférence qui est traité avec des outils statistiques tel que le *stochastic block model* (SBM) [HLL83, NS01]. L'idée derrière le SBM est la suivante : la probabilité que deux nœuds soient reliés dépend uniquement de leur groupe respectif. Si le graphe a une structure communautaire, alors deux nœuds d'une même communauté devraient avoir une forte chance d'être connecté. À l'inverse, deux nœuds de deux communautés différentes devraient avoir une probabilité assez faible d'être connecté. Le SBM est défini par de nombreux paramètres : le nombre de groupe, l'assignation d'un groupe à chaque nœud et les probabilités d'interactions entre les groupes. Avec un jeu de paramètres donné, il est alors possible de calculer la vraisemblance que ce jeu de paramètre soit à l'origine du graphe. Trouver une partition de nœuds dans ce contexte est alors équivalent à trouver le jeu de paramètre qui est le plus vraisemblablement à l'origine du graphe.

Dans le SBM, tous les nœuds sont considérés comme équivalents en particulier vis-à-vis du degré ce qui n'est pas le cas dans beaucoup de graphes provenant de données réelles. C'est pourquoi une version tenant compte du degré des nœuds a été proposée : le Degree-correcte Stochastic Block Model (DBSM) [KN11]. Enfin d'après de récents travaux [New16], il semblerait que le DBSM et l'optimisation de la modularité soient liés.

Méthodes utilisant des marches aléatoires

Il existe de nombreuses autres approches que celles utilisant un modèle nul ou un modèle générateur. Notamment, il y a des méthodes utilisant les marches aléatoires [PL05, RB08]. Ces

méthodes tirent parti du fait que si une communauté est densément connectée alors un marcheur aléatoire devrait y rester assez longtemps. En particulier, la méthode Infomap [RB08] repose sur une idée très élégante ; une partition de nœuds est une carte du graphe et, en ce sens, elle doit aider sa lecture. Une carte est efficace si elle permet de mieux comprendre l'objet d'étude en réduisant sa complexité. Dans une carte d'un pays, les départements découpent l'espace en zones compactes et la plupart du temps la majorité des routes se retrouvent à l'intérieur des départements. Ainsi un voyageur se déplaçant aléatoirement sur les routes a peu de chance de sortir d'un département. Pour décrire, *a posteriori*, l'ensemble des routes prises par ce voyageur, il suffit alors de donner le département initiale puis la liste des routes empruntées. Il n'est pas nécessaire de répéter le département à chaque fois si le voyageur n'en est pas sortit. En ce sens, la découpe d'un pays en département permet de réduire la complexité du voyage. Il s'agit donc d'un problème lié à la théorie de l'information et de sa compression. De manière similaire, Rosval *et al.* utilise des marcheurs aléatoires se déplaçant sur les nœuds du graphe et les communautés forment des zones du graphe. Si les communautés sont bien formées et que les marcheurs aléatoires restent bloqués à l'intérieur, alors la description de leur marche aléatoire sera courte. La longueur de cette description devient alors la signature de la partition et plus la signature est courte, meilleur la partition est.

La méthode a également été étudiée pour voir si elle souffre de *réolution limite* comme la modularité [KR15]. Il semble que cet effet existe également dans Infomap mais qu'il soit beaucoup moins prononcé.

Autres méthodes Il existe bien d'autres méthodes pour détecter des communautés en tant que partition de nœuds. Il y a les méthodes spectrales [DM04, MT09] qui se base sur les vecteurs propres de la représentation d'un graphe sous la forme d'une matrice. De manière moins formelle, les méthodes sont des algorithme de propagations de labels (LPA) [RAK07, LLJT14]. Dans ces méthodes, chaque nœuds a initialement un label puis à chaque itération chaque nœud prend comme label un des label de ses voisins. En général, un nœud prend comme label celui qui est le plus présent parmi ses voisins. Au bout d'un certain nombre d'itération ou l'équilibre, il ne reste que beaucoup moins de labels qui représentent les communautés.

Résumé

Il existe de très nombreuses méthodes pour la détection de communautés en tant que partition de nœuds. Ils semblent que les méthodes d'optimisation de **modularité**, la méthode **infomap** et les **Stochastic Block Model** soient les plus utilisées dans la littérature.

1.2.2 Couverture de nœuds

Jusqu'à maintenant nous avons considérons les communautés comme des partition de nœuds. Or, les partitions sont très restrictives et ne peuvent pas capturer toutes les situations possibles. Reprenons l'exemple d'un graphe reflétant des interactions de personnes comme dans la section 1.2. Il existent des communautés qui sont disjointes comme le travail et la famille mais bien souvent des personnes appartenant à plusieurs groupes, voire l'exemple dans la figure 1.2. Ainsi le groupe de personnes faisant du sport ensemble et le groupe des personnes travaillant ensemble peuvent ne pas être disjoint. Si tel est le cas, alors il n'est plus possible de représenter des communautés avec une partition. Il est nécessaire de manipuler une couverture de nœuds. Ainsi dans l'exemple dans la figure 1.2, les nœuds rouges devraient appartenir deux groupes au lieu d'un seul.

Une fois encore, la littérature est très vaste dans ce domaine et nous ne ferons pas une liste exhaustive des méthodes existantes. Pour une liste plus exhaustive, il existe de nombreux état-de-l'art dans le domaine [DGG12, Kan14, XKS13, BCS15, HDF14].

3. Image provenant de https://en.wikipedia.org/wiki/Clique_percolation_method.

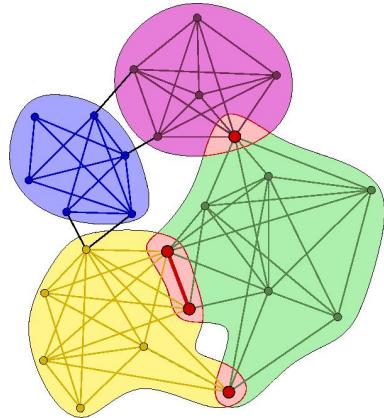


FIGURE 1.2 – Exemple graphe avec une structure communautaire chevauchante représenté par les couleurs³.

Une des premières méthodes de détection de couverture de nœuds est la *Clique Percolation Method* (CPM) [PDFV05]. L'algorithme CPM repose sur le principe de transitivité qui serait à l'origine des communautés : si i et j sont reliés par un lien alors le nœud k qui est déjà relié à i a une forte chance d'être également connecté à k . Il s'agit de la formalisation du proverbe "Les amis de mes amis sont mes amis". Si ce principe est réellement à l'origine des communautés, alors elles doivent être composées de plusieurs cliques. C'est pourquoi CPM cherche l'ensemble des cliques d'une taille k donnée, en générale $k < 10$ pour des raisons de coût de calcul, puis fusionne toutes les cliques qui partagent suffisamment de nœuds, en générale $k - 1$. Comme cette méthode est relativement couteuse, Kumpula *et al.* [KKKS08] ont repris le même mécanisme en optimisant le mode de calcul.

Extension de méthodes existantes

La majorité des méthodes existantes pour les partitions ont été adaptées pour manipuler les couvertures de nœuds. Il existe plusieurs extensions de la modularité [SCCH09, NMCM09]. Cependant ces extensions ne reposent plus sur un modèle nul car elles introduisent des termes de normalisations. Par conséquent, ces extensions sont beaucoup moins utilisées que la modularité initiale.

Stochastic Block Model En revanche, le SBM s'adapte très bien aux couvertures de nœuds. Une extension du SBM est le Mix-Membership Stochastic Block Model (MMSBM)[ABFX08] qui permet à un nœuds d'avoir plusieurs groupes. Dans [GB13], les auteurs utilisent la même méthodes mais en échantillonnant le graphe initiale afin de réduire le cout de calcul. Il existe d'autres méthodes à base de modèle génératif [BKN11, YL13]. Ces méthodes se basent sur des graphes d'affiliation[Bre74]. Un graphe d'affiliation est un graphe biparti entre les nœuds d'une part et les communautés de l'autre part. Dans la méthode de Ballet *et al.*, ce graphe d'affiliation est pondéré et la pondération représente la propension d'un nœuds à créer des liens dans un groupe donné tant dis que dans Yang *et al.* il s'agit du facteur d'appartenance du nœuds au groupe. Une fois le graphe d'affiliation défini, il est nécessaire de savoir recréer la matrice d'adjacence et, en ce sens, ces méthodes se rapprochent des méthodes de factorisation non négative de matrices [LS99]. Le but de la factorisation non-négative d'une matrice donnée est d'être capable de trouver deux matrices dont les entrées sont non-négatives tel que leur combinaison permettre de retrouver la matrice initiale.

Jusque récemment ce genre de technique généralisant le SBM ne pouvaient s'appliquer qu'à des graphes relativement petit. Il semble cependant que les méthodes récentes [GB13, YL13] arrivent à traiter des graphes ayant énormément de liens, de l'ordre 10^{12} liens.

Propagation de label Des méthodes ont étendus la propagation de label aux couvertures de nœuds [Gre10, XSL11]. Le principal changement est qu'un nœud ne stocke plus un unique label mais soit plusieurs labels soit des fréquences d'apparition de label. Ainsi à la fin de l'algorithme, il suffit de choisir les labels les plus fréquents. Cependant ce mécanisme de propagation change radicalement de l'idée initiale. En effet, la diffusion d'un label n'est plus directement contrainte par la diffusion des autres labels. Dans cette nouvelle configuration, il est possible qu'un label soit présent dans tous les nœuds. De plus selon les auteurs de ces méthodes, des communautés non connexes peuvent apparaître ce qui nécessite l'ajout d'un mécanisme de post-processing.

Infomap Il s'agit sûrement de la méthode étant le moins "déformé" avec le SBM par l'adaptation aux partitions chevauchantes. En effet, il suffit de relâcher la contrainte sur le fait qu'un nœud n'appartienne qu'à un seul groupe. Il est toujours possible de décrire un trajet par un nom de groupe puis une liste de nœuds visités dans le même groupe. Ainsi, la notion de longueur de description est toujours valide même dans contexte. Ce travail d'extension a été fait par Esquivel et al. [ER11].

Méthodes utilisant des fonctions de qualité locales

On a vu avec la modularité qu'il est assez difficile de définir une fonction de qualité évaluant une couverture de nœuds en fonctions des caractéristiques des groupes. En revanche, les LPA tirent parti de l'information locale pour créer des groupes locales mais ne permettent pas leur évaluation. L'idée est en quelque sorte mélanger ces deux aspects. Il s'agit d'algorithmes qui initialement considèrent de très petites communautés puis essayent de les étendre itérativement en ajoutant des nœuds. Afin de ne s'étendre un groupe indéfiniment comme cela peut être le cas avec les LPA, un ajout n'est fait que si l'améliore une fonction de qualité locale. Ainsi dans ce type d'algorithme, il y a principalement deux critères importants : le choix des communautés initiales et le critère d'évaluation. Dans la majorité des cas, chaque nœud constitue une communauté. OSLOM[LRRF11] diffère de cette situation car OSLOM utilise comme point de départ une partition trouvée par l'algorithme de Louvain ou par infomap.

Les fonctions de qualité applicable sont très nombreuses et nous ne mentionneront que trois. Tout d'abord, il est possible d'utiliser le degré relatif [LWP08] d'un groupe comme critère. Il s'agit du ratio entre le nombre de l_{in} liens internes à un groupe de nœuds et du nombre de liens totale $l_{in} + l_{out}$: $\frac{l_{in}}{l_{in} + l_{out}}$. En effet, plus le degré relatif est élevé, plus le groupe est dense comparé à son voisinage et plus il a de forte chance d'être une bonne communauté. Cette formulation est assez proche de la conductance ou coupe normalisée[SM00] :

$$\phi = \frac{l_{out}}{\min(2(l_{in} + l_{out}), m - 2(l_{in} - l_{out}))} \quad (1.6)$$

Ces deux notions se rapportent à la notion de densité vis-à-vis de leur voisinage. Il est également possible d'évaluer une communauté locale par rapport aux nombres de triangles présents de la communauté. C'est ce que mesure la cohesion [FCF11] pour un groupe de taille k :

$$cohesion = \frac{\Delta_3}{\binom{k}{3}} \times \frac{\Delta_3}{\Delta_3 + \Delta_2}, \quad (1.7)$$

où Δ_3 est le nombre de triangle dont les 3 nœuds sont à l'intérieur du groupe et Δ_2 est le nombre de triangle dont uniquement 2 nœuds sont à l'intérieur du groupe.

Différentes méthodes d'initialisation ainsi que fonctions de qualité sont détaillées dans l'article de Kanawati [Kan14]. Ces méthodes semblent très prometteuses car elles permettent de traiter efficacement de très grand graphe tout comme les LPA mais elles profitent des fonctions de qualité locales qui permettent d'une certaines manières d'évaluer le résultat obtenu.

Résumé

La littérature sur la détection de couverture de nœud est encore très récente. Il est donc délicat de commencer à tirer des conclusions. Cependant, il semble que les extensions de la modularité ne soient pas réellement capable de trouver des couvertures de nœuds. En plus d'infomap, c'est surtout les méthodes utilisant des modèles génératifs et celles utilisant des fonctions de qualité locales qui semblent les plus à même de capturer des couvertures de nœuds pertinentes. Bien que souvent limité à de petit exemple, les modèles génératifs semblent maintenant capable de traiter des graphes de tailles importantes. Les méthodes utilisant des fonctions de qualité locales sont quant à elle très rapide mais le choix de la fonction de qualité semble encore délicat.

1.3 Extension temporelle des graphes

Les graphes sont utilisés pour représenter une situation ou résoudre un problème réel. Dans le chapitre précédent, nous nous sommes attachés à présenter une partie des méthodes considérant le problème de la détection de communautés recouvrantes ou non. Cependant toutes ces méthodes reposent sur la validité de la représentation du problème par un graphe. Cette hypothèse est justifiée dans énormément de cas. Ce n'est pas contre pas le cas lorsque l'objet d'étude évolue durant la capture.

Afin d'illustrer ce propos, nous appuyons sur le même exemple de réseau de personnes que nous avons présenté dans la sous-section 1.2.1 et qui est illustré dans la figure 1.1. Dans cet exemple, deux élèves sont reliés dans le graphe si ils ont interagi au moins une fois au cours de la capture. Dans ce cas, on parle de graphe agrégé car la dimension temporelle est complètement oubliée. Comme la capture n'a duré que deux jours, il est fort probable que les élèves n'aient pas changé d'habitude durant la capture. En étudiant le graphe agrégé, il est possible de mettre en avant l'organisation générale des élèves mais déjà de l'information a été perdue. En effet, il n'est pas possible de différencier le comportement observé le matin de celui observé le soir car l'ensemble des interactions sont agrégées dans le graphe. Ce n'est pas tout, imaginons que la capture ait duré plusieurs mois voir plusieurs années. Dans ce cas de figure, il ne sera plus possible de dégager un comportement général des élèves car ils auront changé durant ce laps de temps. En particulier, de nouveaux groupes d'amis se seront formés et d'autres auront disparu. Tout ces changements impactent le graphe agrégé et tendent au fur et à mesure à le rapprocher à une grande clique ce qui le rend complètement inexploitable. Le temps est donc une dimension qu'il est nécessaire de prendre en compte.

Le domaine de l'épidémiologie sont de très bon exemples de l'importance du temps. Un modèle épidémique modélise la propagation d'une maladie dans une population en fonction des contacts qui existent. Il est donc tout naturel de s'appuyer initialement sur un graphe où les nœuds représentent des personnes et les liens représentent les interaction entre personnes. En plus du graphe, il est nécessaire d'ajouter l'états de chaque nœuds, *e.g.* nœud sain ou nœud infecté. Ainsi, un nœud sain ne peut devenir infecté que s'il est relié à un nœud infecté dans le graphe. Ce genre de modèle mettent donc en avant un chemin de diffusions, *e.g.* une suite de personnes transmettant sa maladie à l'autre.

Afin d'illustrer ce phénomène, les premiers travaux [VBB08] s'appuient sur un graphe d'interaction de personnes agrégeant toute l'information temporelle. Or, la prise en compte du

temps dans ce contexte est primordiale car il est possible que le chemin d'infections simulé dans le graphe agrégé ne soit pas réalisable si l'on prend en compte le temps. Imaginons 3 personnes A, B, C tel que A et B interagissent à l'instant 1, B et C interagissent à l'instant 2. Dans le graphe agrégé, il est possible pour C d'infecter A via B alors que dans la réalité cela n'est possible. L'ajout du temps impacte donc fortement les résultats obtenus dans le contexte épidémiologique et ce phénomène est d'ailleurs très étudié [GPBC15, KKP⁺11, JPKK14, HK14, HL14, SWP⁺14, PJHS14].

Une fois reconnue l'importance du temps, il est nécessaire de trouver un nouveau formalisme étendant la théorie des graphes pour tenir compte du temps. Nous présentons maintenant différentes extensions possibles de la théorie des graphes, dans les sous-sections 1.3.1 et 1.3.2. Nous détaillons également comment la recherche de communautés se transpose dans ces nouveaux formalismes et plus généralement quelles sont les problèmes qu'ils permettent de résoudre. Des états de l'art dans ce domaine ont d'ailleurs déjà été esquissés [BBC⁺14, CA14, HKW14].

1.3.1 Extensions avec pertes d'informations temporelles

Séries de graphes

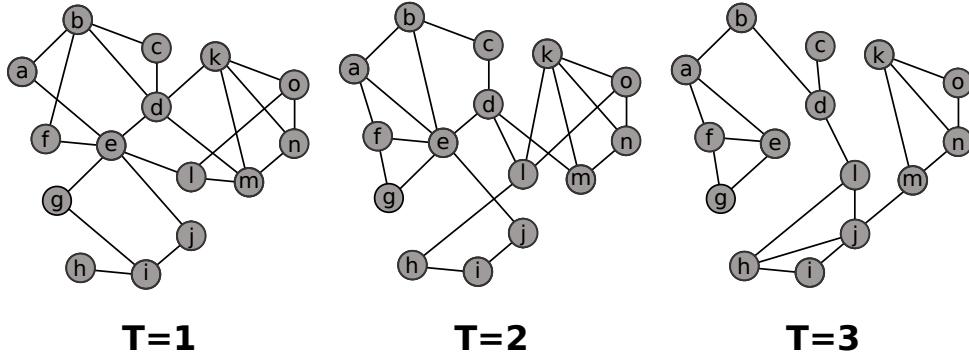


FIGURE 1.3 – Exemple de série de graphes sur trois intervalles de temps.

La première solution qui a été apportée ne prends le temps en compte que partiellement. Il s'agit de manipuler une série de graphe où chaque graphe représente le système durant un intervalle de temps donné. Ainsi, il est possible d'appliquer les outils de la théorie des graphes sur chaque intervalle de temps. Cependant chaque intervalle de temps est représenté par un graphe agrégé. Il y a donc une perte d'information. Plus formellement, une série de graphe est défini par $\mathcal{G} = \{G_i\}_{i < T}$ où T est un entier, voir l'illustration 1.3.

Cette définition initiale laisse le choix sur la découpe du temps en intervalle. Il est possible de choisir des intervalles de tailles égales ou non et disjoints ou chevauchants [WFAG12]. Utiliser des intervalles à durée variable permet de mieux tenir compte de la dynamique. Par exemple lors de l'étude d'interactions de personnes, il est très courant d'observer une très faible activité la nuit et une plus forte le soir. Un graphe agrégant ce qui se passe sur un intervalle de 5h sera vide dans le premier cas et peut être trop dense dans le second. La détection d'intervalle pertinent est donc un vaste sujet de recherche [RB10, KKB⁺12, RPB13, CBW13, PC15, DVR16].

La notion même de temps est importante. Albano *et al.* [AGL14] propose même d'utiliser une autre mesure que la seconde ou la journée mais plutôt le nombre de changement comme mesure du temps. Ainsi, le temps n'avance pas si aucun changement n'a lieu et au contraire il avance beaucoup si énormément de changement apparaissent. Cette manière de procéder se rapproche des travaux de Lamport [Lam78] dans les systèmes distribués.

Détection de communautés Dans ce contexte de série de graphe, il y a eu assez tôt des méthodes de détection de communautés [HKKS04, SFPY07, LCZ⁺08, APU09]. Il est, en effet, assez naturel d'appliquer une méthode de détection statique sur chaque graphe puis d'essayer de faire du suivi de communautés. Le suivi de communauté consiste en étant une série de graphe et une série de partition comprendre comme un communauté donnée à évoluer. Palla *et al.* [PBV07] ont été parmi les premiers à décrire les évolutions possibles d'une communauté. Muni d'indicateur de similarité tel que l'indice de Jaccard, voir 1.1.2, il est possible de trouver les communautés les plus proches aux instant précédent et suivant. À partir de ces informations, il est défini 5 types d'évolutions d'une communauté :

Naissance il n'existe pas de communauté proche précédemment.

Agrandissement La communauté continue d'exister et s'agrandit.

Fusion Deux communautés fusionnent pour donner lieu à une nouvelle communauté.

Division Une communauté se sépare en deux nouvelles communautés à l'étape suivante.

Mort Une communauté cesse d'exister dans les graphes suivants.

Ce type de méthode souffre souvent de l'instabilité des méthodes de détections [AG10, HBG14]. Si les partitions changent complètement entre deux graphes consécutifs, alors il est difficile de faire un réel suivi de communautés. C'est pourquoi des méthodes essayent de forcer une certaine stabilité de la partition en ajoutant un coût de transition [CKT06, CKU13, KBV15]. Une approche détournée pour garder une certaine stabilité est d'utiliser la partition trouvée précédemment comme base de recherche pour l'intervalle suivant [LRRF11, CLR16].

Tenseur 3D

Les tenseurs 3D ne sont pas en soi différents des séries de graphes. Il est possible de voir un graphe comme une matrice carrée d'adjacence. Il est donc normal de concevoir une série de graphes comme un tenseur appartenant à \mathcal{R}_{nnT} . Ce changement de point de vue permet ainsi d'appliquer les méthodes d'algèbre linéaire, notamment la décomposition de tenseur. C'est la méthode proposée par Gauvin *et al.* [GPC14] pour étudier la structure communautaire des interactions d'élèves.

Graphes multicouche

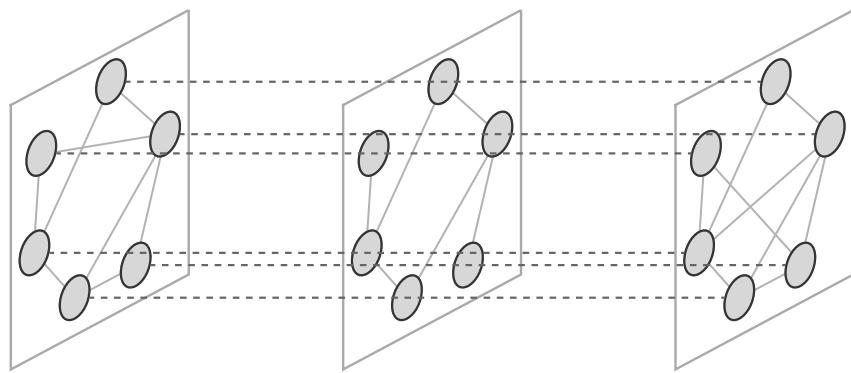


FIGURE 1.4 – Graphe multicouche avec trois couches représentant le temps. Les liens pleins (resp. pointillés) sont les liens intra-couches (resp. inter-couches).

La construction de graphes multicouche (*multilayer* ou *multiplex*) est proche de l'idée des séries de graphes. Tout comme les séries de graphes, des nœuds et des liens sont créés à chaque intervalle pour représenter ce qui s'est déroulé l'intervalle de temps. Cependant, le graphe

multicouche ajoute des liens entre les nœuds de deux intervalles si'ils sont identiques, voire l'illustration 1.4. Par conséquent, un graphe multicouche est un graphe où il existe deux type de liens dans un graphe multicouche : les liens intra-couches et les liens inter-couches. Les liens intra-couches représentent un connexion entre entre deux nœuds durant un intervalle de temps. Les liens inter-couches représentent un lien entre deux nœuds sur deux intervalles différents. Ces derniers sont utilisés pour identifier un même nœud sur plusieurs intervalles et sont en générales limités à relier deux couches consécutives.

Les graphes multicouches représentent très bien les donnée évoluant dans le temps mais ils modélisent aussi très bien d'autres situations. Par exemple, ils permettent de représenter facilement les différent moyens de transport dans une ville où chaque moyen de transport (bus, voiture, métro ...) est représenté par une couche. Plusieurs travaux [DSRC⁺13, KAB⁺14, BBC⁺14] décrivent les graphes multicouches et leurs applications.

Détection de communautés Grâce au formalisme de graphe multicouche, il est possible de traiter le temps de manière un peu plus fine que dans les séries de graphes car il permet de mieux suivre l'évolution des nœuds. Comme un graphe multicouche est un graphe, il est possible d'adapter les méthodes existantes pour tenir compte des différents type de liens. C'est le cas d'infomap [DLAR14], de la modularité [MRM⁺10, BPW⁺13, BPW⁺16] et du SBM [SSTM15, Pei15].

Bilan

Les séries de graphes, les tenseurs et les graphes multicouches permettent de prendre en compte le temps tout en autorisant l'utilisation de méthode conçu sur un graphe statique. Cette liberté d'utilisation à un coût. Ces approches reposent sur une découpe du temps en sous-intervalles durant lesquelles il est possible d'agrégé temporellement pour obtenir un graphe statique. Il peut être délicat de définir un intervalle de temps qui permet de construire des graphes qui aient du sens. La construction des graphes agrégés entraîne une perte d'information temporelle et cela impacte la précision temporelle des structures communautaires qui sont manipulables. Il n'est pas envisageable d'augmenter le nombre d'intervalle de temps car cela induirait d'une part des graphes agrégés avec très peu de liens et d'autre part le temps de calcul serait très fortement impacté.

1.3.2 Extension sans pertes d'informations temporelles

Graphe temporel

Les graphes temporels (*Time Varying Graph* ou *Evolving Graph*) permettent de tenir compte de l'ensemble de l'information temporelle. Pour cela au lieu de considérer des intervalles de temps, ils considèrent l'ensemble des modifications qui affectent le graphe : les ajouts et retraits de liens. En pratique, cela revient à considérer sur chaque lien une fonction de la présence en fonction de temps qui vaut 1 à un instant t si le lien existe à cet instant et 0 sinon. Ce formalisme est présenté dans différents travaux [CFQS11, WZF14].

Détection de communautés Dans un graphe temporel, une structure de graphe existe à chaque instant. Il est donc possible de calculer après chaque modification l'évolution d'une métrique. Par exemple, il est possible de calculer après l'ajout d'un lien le nouveau degré interne des nœuds impacté par ce changement.. En fonction de l'évolution de cette métrique, il est alors décidé d'ajouter ou retirer un nœud voir même de fusionner deux communautés. Li *et al.* [LHB⁺12] se base sur le nombre de liens que partage un nœuds avec les communautés environnantes. Ainsi, un nœud est toujours dans la communauté avec laquelle il partage le plus de liens.

Shanget *et al.* [SLX⁺14] et Cordeiro *et al.* [CSG16] se basent sur l'évolution de la modularité. Cependant, ces approches ne permettent pas l'ensemble des évolutions possibles de communauté, en particulier l'apparition d'une nouvelle communauté. C'est pourquoi l'évolution de la structure courante peut mener à une structure ayant une faible qualité. Une autre approche a été proposée par Cazabet *et al.* [CAH10]. Elle se utilise sur une métrique locale basé sur le nombre de chemins de longueur 2 existant entre un nœud et une communauté. Après chaque modification, ils considèrent également la possibilité de créer une nouvelle communauté sous la forme d'une petite clique.

Flots de liens

Dans les approches précédentes,
proche[MM15]

[HS13, Hol15b, Hol15a] plein d'étude possible Formalisation algébric [BP15]. bursty [KKBK12, KKP⁺11, MSPS15, SBB10] [MSMA08, MSCA09] heavy tail, [RB13]

motif [KKK⁺11, KKKS13]. Le formalisme de flot de liens est défini plus en profondeur dans le chapitre 2. randomwalk[SBBPS12]

[GVFS⁺16] centralité [CVW⁺15, KA12, PSG⁺13, PB15, SWG15, TYY16] densité [VL14]

Résumé

L'extension temporelle des graphes est un champ de recherche très récent et il est difficile d'avoir du recul sur les possibilités qu'offre chaque méthodes. Il semble tout de même se dessiner différentes catégories.

C'est le bordel plein de notation et tout ça.

Chapitre 2

Flots de liens : extensions temporelle des graphes

Sommaire

2.1 Définition	19
2.2 Sous-flot induit	19
2.3 Degré et densité	19

2.1 Définition

Un flot de liens est défini comme un triplet : $\mathcal{L} = (T, V, E)$, où $T = [\alpha, \omega]$ est un intervalle de temps, V un ensemble de n nœuds et $E \subseteq T \times T \times V \times V$ un ensemble de m liens. Les liens de E sont des quadruplets (b, e, u, v) , signifiant que la paire de nœuds (u, v) est connectée sur l'intervalle $[b, e] \subseteq [\alpha, \omega]$. Nous considérons un flot non orienté, i.e. $(b, e, u, v) = (b, e, v, u)$, et sans boucle, i.e. $u \neq v$.

Nous dénotons la durée du flot par $\bar{L} = \omega - \alpha$. $\beta(E) = \min_{(b,e,u,v) \in E}(b)$ et $\psi(E) = \max_{(b,e,u,v) \in E}(e)$ sont respectivement l'apparition du premier lien et la disparition du dernier lien.

Un flot de liens est simple si pour tout $(b, e, u, v) \in E$ et $(b', e', u, v) \in E$, $[b, e] \cap [b', e'] = \emptyset$. La simplification d'un flot de liens $\sigma = L$.

$V(E') = u_{(b,e,u,v) \text{ in } E'}$ sommets induits par un ensemble de liens.

$\xi(L, \Delta)$ ajout de Δ à chaque lien.

Que des flots avec durées ou bien delta densité?

2.2 Sous-flot induit

Sous flot induit par un ensemble de lien $E' : L(E') = ([\beta(E'), \psi(E'), V(E'), E'])$.

Sous flot induit par un ensemble de paire nœuds $S \in V^2 : L(S) = ([\beta(E'), \psi(E')], V', E')$ avec $E' = \{(b, e, u, v) \in E, (u, v) \in S\}$. Par convention, on note $L(v) = L(\{v\} \times V)$.

Sous flot induit par un intervalle de temps $T' = [\alpha', \omega']$, $T' \subseteq T : L_{\alpha'.. \omega'} : L_{\alpha'.. \omega'} = ([\alpha', \omega'], V(E'), E')$ avec $E' = \{(b', e', u, v), \exists (b, e, u, v) \in E, b' = \max(b, \alpha'), e' = \min(e, \omega')\}$. Par convention, on note $L_{t..t} = L_t$

Il est aussi possible de combiner ces notions. Par exemple avec $V' \subset V$, $L_{\alpha'.. \omega'}(V'^2)$ est le sous flot correspondant au lien entre les nœuds de V' sur l'intervalle $[\alpha', \omega']$.

2.3 Degré et densité

Beaucoup de notions sont développées autour de cet objet. Degré temporelle d'un nœud u :

$$d_t(u) = |L_t(v)| = |\{(b, e, u, v) \in E, b \leq t \leq e\}| \quad (2.1)$$

Par extension pour un ensemble de sommets V' :

$$d_t(V') = |L_t(V'^2)| = |\{(b, e, u, v) \in E, u, v \in V', b \leq t \leq e\}| \quad (2.2)$$

Degré interne maintenant ?

$$d_t(E') = |\{(b, e, u, v) \in E', b \leq t \leq e\}| \quad (2.3)$$

Il est possible d'intégrer sur l'ensemble du temps

$$D_{\alpha..w}(v) = \int_{\alpha}^{\omega} d(v, t) dt = D(v) \quad (2.4)$$

Par convention, on notera le degré moyen de v : $d(v) = \langle d(v, t) \rangle = \frac{D(v)}{\omega - \alpha}$. Il en va de même pour les autres degrés.

C'est le cas de la densité :

$$\delta(L) = \frac{2 \sum_{l \in E}}{n(n-1)(\bar{L})} \quad (2.5)$$

Liste des notations pour les flots de liens

Symbol	description
L	Flot de liens
T	intervalle de temps
V	ensemble de nœuds
E	ensemble de liens : (b, e, u, v)
n	nombre de nœuds
$ L , E $	nombre de liens dans le flot
$\beta(E)$	temps d'apparition du premier lien
$\psi(E)$	temps de disparition du dernier lien
$\xi(L, \Delta)$	Flot de liens où chaque lien dure Δ
$L(V'^2)$	sous flot induits par les nœuds de V'
$L_{t..t'}$	sous flot induits par l'intervalle $[t, t']$
$d_t(v)$	degré de v à l'instant t
$d(v)$	degré moyen v sur T
$\delta(L)$	densité du flot
$\delta_\Delta(L)$	densité du flot ou chaque lien dure Δ

Chapitre 3

Étude d'une archive de courriels

Sommaire

3.1	Prétraitement sur le jeu de données	23
3.2	Caractéristiques élémentaires des discussions	24
3.3	Étude des discussions en tant que sous-flots	26
3.3.1	Application de la Δ -densité	26
3.3.2	Répartition temporelle et structurelle des discussions	29
3.3.3	Flot quotient	31
3.3.4	Conclusion	32
3.4	Détection de structures denses	33
3.4.1	Méthode de détection	33
3.4.2	Comparaison des partitions	34
3.4.3	Conclusion	36

Nous nous intéressons ici à une archive de courriels publiquement disponibles¹. Cette archive contient l'ensemble des courriels échangés par différents utilisateurs pour résoudre un problème survenu lors de l'utilisation de Debian. Typiquement, une personne ayant un problème lors de l'installation envoie un courriel à la liste afin de demander de l'aide. Toute personne inscrite sur la liste reçoit ce courriel et peut y répondre ce qui donne lieu à une discussion visible par tous. Ces discussions ont déjà été étudiées dans le passé [DLCA07, SSA06, WWL⁺14] mais cela a été fait en utilisant des méthodes statiques uniquement.

Or, ces données se représentent naturellement sous forme de flot de liens en associant chaque personne à un nœud et chaque courriel entre deux personnes à un lien dans le flot de liens à l'instant où le courriel a été envoyé. L'avantage de ces données de communications est que nous connaissons la discussion (*thread*) dans laquelle a lieu chaque message. Une discussion est un ensemble de courriels dont tous les messages répondent à un message précédent de la discussion excepté pour le premier qui a initié la discussion et que nous appelons *racine*. Ainsi, nous étudions la structure des discussions dans le flot de liens représentant les courriels envoyés sur la liste.

Utiliser le formalisme de flot de liens est particulièrement intéressant car cette ligue de diffusion existe depuis 1994. L'aspect temporel des discussions est donc important.

3.1 Prétraitement sur le jeu de données

Bien qu'accessible sur internet, ce jeu de données nécessite un ensemble de traitements avant de pouvoir exploiter les 724985 courriels que contenait l'archive en janvier 2015. Tout d'abord, les données ne sont pas sous la forme d'un flot de liens avec la structure des conversations. Les données sont accessibles via le site internet et ne sont pas structurées. Pour avoir ces informations sous la forme d'un flot de liens, un script d'extraction a été développé [URL](#).

1. <https://lists.debian.org/debian-user/>

Lors de l'extraction, 2269 courriels n'ont pas pu être pris en compte car certaines informations étaient manquantes ou mal formées, typiquement à cause de la date ou d'un fuseau horaire non reconnu.

Une fois les informations brutes récupérées, il faut les transformer en un flot de liens cohérent. Pour chaque message m , nous extrayons son auteur $a(m)$, l'instant $t(m)$ auquel le message a été posté², le message auquel il répond $p(m)$ trouvé via le champ IN-REPLY-TO, son destinataire $a(p(m))$ et la discussion $D(m)$ dans laquelle il apparaît. Comme les messages *racines* ne répondent à aucun autre, nous imposons $p(m) = m$. L'ensemble de liens du flot est donc $\{(t(m), a(m), a(p(m)))\}_m$. Nous ne prenons pas en compte la direction des liens.

Une fois le flot créé, il est encore nécessaire de vérifier sa cohérence. Un message peut être filtré pour différentes raisons : le courriel apparaît avant le message auquel il est censé répondre, le message auquel il répond n'est pas présent dans l'archive, l'auteur et le destinataire sont la même personne. Cette dernière condition permet notamment d'éviter la présence de boucles dans le flot. Cela concerne principalement les *racines*. Il s'agit de vérifications simples auquel il faut ajouter les vérifications sur la cohérence de la structure des discussions. Ainsi, une discussion est entièrement retirée du jeu de données s'il manque la *racine*, si un de ses messages a été retiré à l'étape précédente. Après ces vérifications, environ 7% des discussions sont retirées. À cela, il faut également tenir compte de notre temps d'observation qui est partiel. En effet, une discussion dont le dernier message a lieu 1 semaine avant la fin de la capture peut ne pas être terminée. De même, une discussion durant très longtemps n'a qu'une faible probabilité d'être capturée en entier. Pour corriger ces biais, nous filtrons également les discussions ayant débutées trop récemment ou durant trop longtemps. La limite pour considérer une discussion trop récente ou trop longue a été fixé à 4 ans (1.26×10^8 s) car nous avons constaté qu'uniquement quelques discussions dépassées ce seuil sur la distribution de durées des discussions dans la figure 3.1. Toutes les discussions qui ont débutées moins de 4 ans avant la capture du jeu de données ne sont donc pas prises en compte.

Une fois tout ces messages filtrés, nous obtenons un flot de liens avec 316 569 liens entre 34 648 personnes pendant presque 19 ans et 116 999 discussions. Mis à part les 237 664 messages de début de discussion, ce sont 168 482 courriels qui ont été filtrés soit environ 23%. La majorité des courriels filtrés l'ont été car ils appartiennent à une discussions trop récente.

3.2 Caractéristiques élémentaires des discussions

Les caractéristiques les plus élémentaires des discussions sont le nombre de courriels, le nombres de personnes, le nombre de paires de personnes distinctes en interaction directes et leur durée. Dans la figure 3.1, sont présentées les distributions cumulatives inverses de ces quantités et on remarque qu'elles sont toutes hétérogènes. On remarque que les données filtrées ne diffèrent pas qualitativement des données brutes.

La distribution des durées des discussions montre que la majorité des discussions dure environ une journée ou moins (10^5 secondes équivaut à moins de 28 heures). Par ailleurs, on remarque qu'il n'existe que quelques discussions durant plus d'un an.

Ces premières observations sont nécessaires mais pas suffisantes pour comprendre les caractéristiques d'une discussion. Nous avons également étudié la corrélation entre ces différentes notions et une partie d'entre elles sont présentées dans la figure 3.2.

La corrélation entre la durée et le nombre de courriels, dans la figure 3.2 partie gauche, met en évidence que plus une discussion est grande en nombre de courriels plus elle dure longtemps, ce qui est attendu. Par contre, on observe que les petites discussions ont des durées très variables. Dans la partie droite de la figure 3.2 présentant la corrélation entre le nombre de courriels et d'auteurs, on observe un autre fait attendu [DLCA07] qui est qu'une discussion est

2. Cet instant est convertit en *timestamp* en tenant compte des fuseaux horaires.

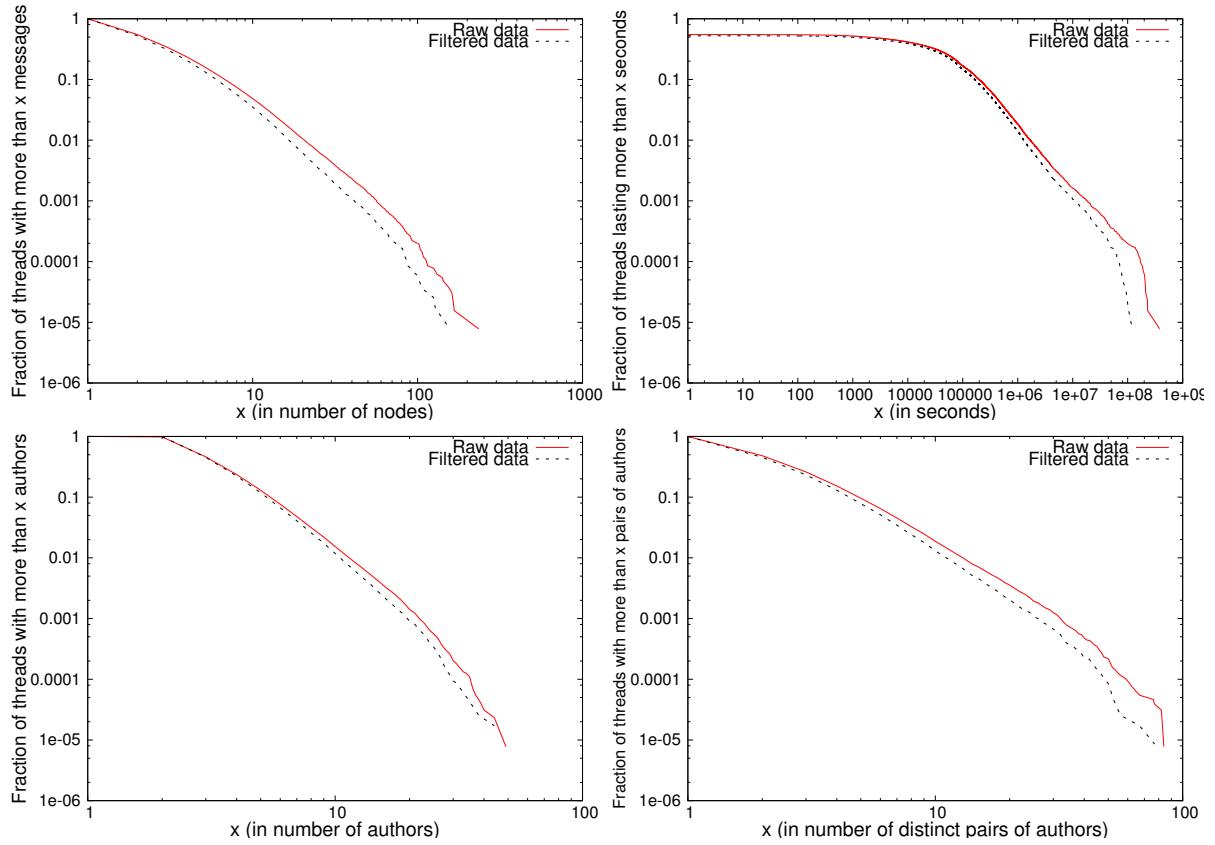


FIGURE 3.1 – Distribution cumulative inverse de différentes caractéristiques pour les données brutes (ligne pleine) et filtrées (ligne pointillé). En haut à gauche : nombre de courriels dans une discussion ; en haut à droite : durée d'une discussion ; en bas à gauche : nombre de personnes dans une discussion ; en bas à droite : nombre de paires d'auteurs distinct dans une .

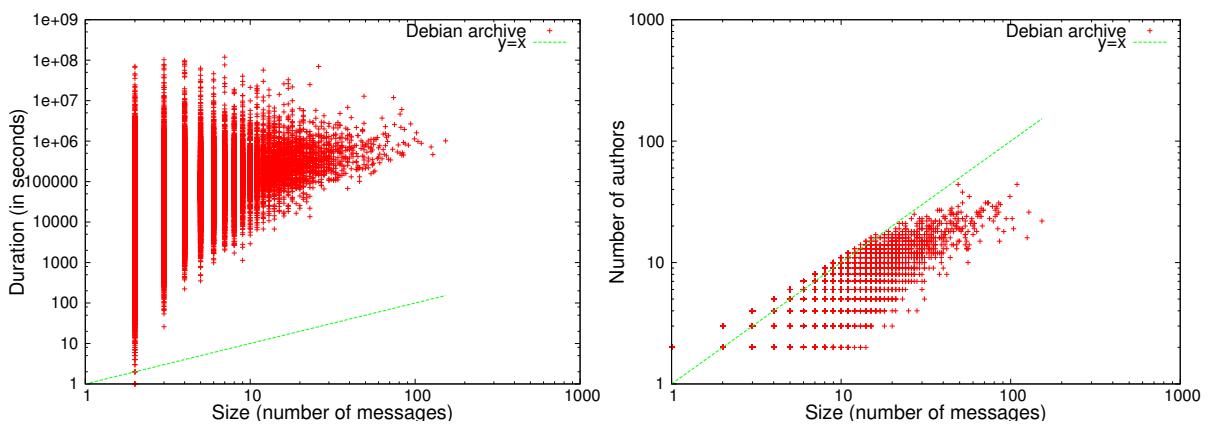


FIGURE 3.2 – Gauche : Corrélation entre le nombre de courriels et la durée d'une discussion. Droite : Corrélation entre le nombre de courriels et le nombre d'auteurs dans une discussion.

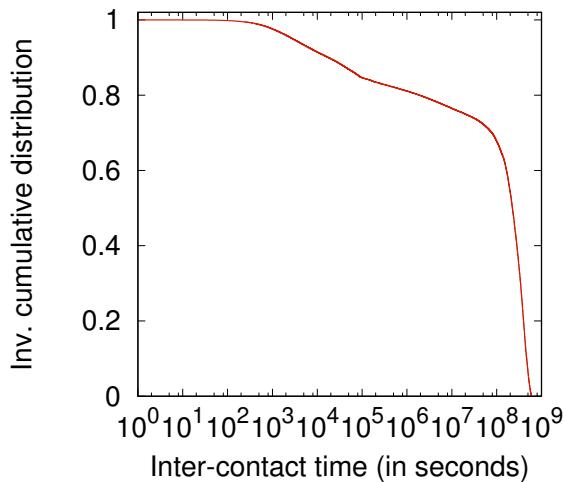


FIGURE 3.3 – Distribution des temps inter-contacts dans le fil de discussions.

constituée, en général, de plus de messages que de participants. Ainsi lors d'une discussion, c'est un petit nombre de personnes qui échangent potentiellement beaucoup de messages.

Enfin, il est intéressant d'observer la dynamique des échanges entre deux personnes. Soit $\tau(u, v) = (t_{i+1} - t_i)_{i=0..k+1}$ la séquence des temps inter-contacts des k liens entre les noeuds u et v , où t_0 est le temps entre α et le premier lien et t_{k+1} est le temps entre le dernier lien et Ω . Il s'agit du temps écoulé avant que deux personnes se contactent à nouveau, indépendamment peu importe la conversation. Dans la figure 3.3 est représentée la distribution cumulative inverse du temps inter-contacts. 21% des temps inter-contacts sont inférieurs à 30 jours (2.6×10^6 s). Ce chiffre bien que relativement faible est tout de même important car il s'agit de discussions ouvertes où tout le monde peut participer. En particulier, une personne peut envoyer une demande d'aide à un moment donné et ne plus jamais échanger avec les mêmes personnes. Or, on observe que 21% des contacts sont renouvelés en moins de 30 jours. La participation est donc relativement élevée.

3.3 Étude des discussions en tant que sous-flots

3.3.1 Application de la Δ -densité

Jusqu'à maintenant aucune notion intrinsèquement liée aux flots de liens n'a été utilisée pour caractériser les discussions. Le but est d'évaluer si cette structure de flot peut se rapprocher d'une structure communautaire. Comme dit précédemment, les communautés sont souvent définies comme étant des structures devant être densément connectées. C'est pourquoi nous nous attachons à étudier la densité des discussions.

Comme ces données se modélisent par un flot de liens où les liens n'ont pas de durée, nous étudions la Δ -densité pour différentes valeurs de Δ entre 1 seconde et 20 ans. Tout d'abord dans la figure 3.4 est représentée la Δ -densité globale du le flot. En couvrant un spectre aussi large de Δ , on observe que la Δ -densité est croissante avec Δ mais surtout on observe bien la convergence de Δ -densité vers 3.139×10^{-4} , la densité du graphe agrégé, lorsque Δ est proche de $\omega - \alpha$.

Cependant, la Δ -densité du flot n'apporte que peu d'informations en elle-même. Elle est surtout utile pour comparer les valeurs de Δ -densité des sous-flots que sont les discussions. Ainsi dans la figure 3.5, est présentée la distribution cumulative inverse de la Δ -densité des discussions pour différentes valeurs de Δ . On remarque que les différentes valeurs de Δ ne

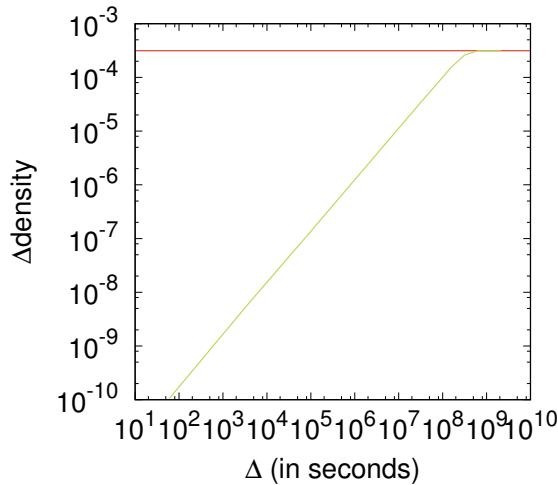


FIGURE 3.4 – Évolution de la Δ -densité (en vert) du flot de liens pour Δ de 60 seconde à 20 ans. En rouge, la densité dans le graphe agrégé.

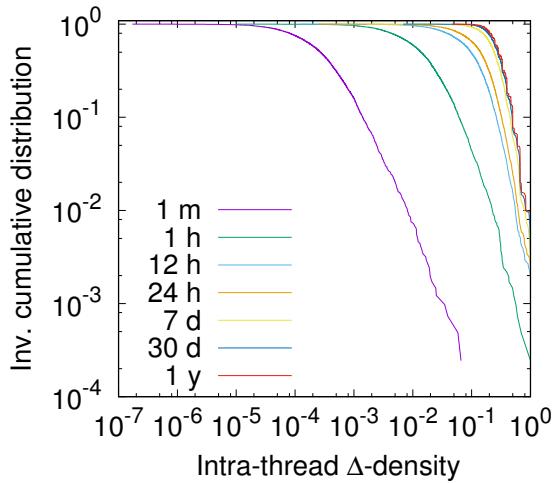


FIGURE 3.5 – Distribution cumulative inverse de la Δ -densité des discussions pour différentes valeurs de Δ s.

semblent pas influencer qualitativement la distribution de Δ -densité. Cette courbe met surtout en évidence que les discussions sont des structures beaucoup plus denses que le flot. En effet, la densité médiane des discussions est, selon la valeur de Δ , entre 2.69×10^{-4} et 0.28 alors que le flot a une Δ -densité variant entre 1.05×10^{-10} et 3.42×10^{-5} . La Δ -densité des discussions est donc en moyenne 10^5 fois plus élevée que celle du flot. Bien que notable, ce fait est attendu notamment car le flot dure beaucoup plus longtemps et concerne beaucoup plus de nœuds que les discussions.

Afin d'aller plus loin dans l'étude de cette structure, il faut revenir à une définition plus précise de ce qu'est une bonne communauté. En soi, une valeur de densité n'est pas suffisante pour définir une structure communautaire. En effet, une discussion ayant une densité de 0.8 peut ne pas être une communauté tandis qu'une autre ayant une densité proche de zéro peut être une communauté. Il faut définir un point de comparaison pour effectivement affirmer qu'une structure est particulièrement dense. La prise en compte de la densité globale est un début mais n'est pas suffisante.

Une autre définition d'une communauté est qu'elle devrait être plus densément connectée à l'intérieur qu'avec les autres communautés adjacentes. Pour un graphe $G = (V, E)$ et une

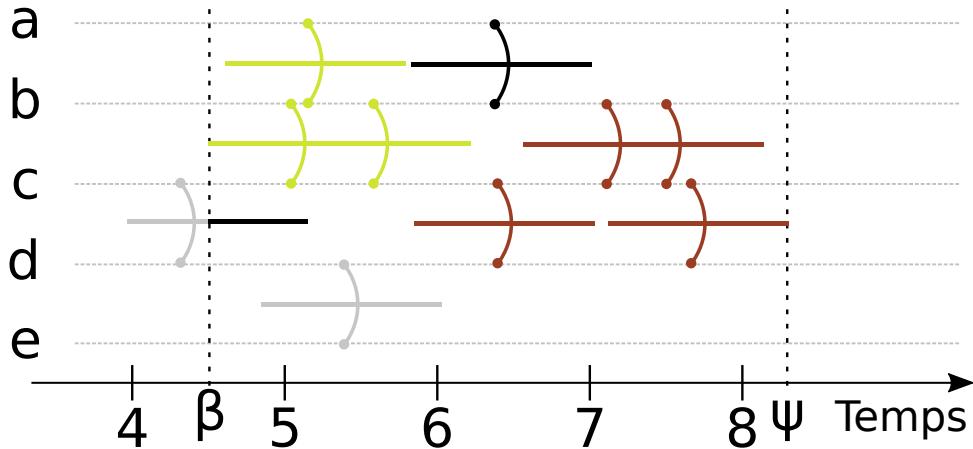


FIGURE 3.6 – Le flot entre les discussions **verte** et **rouge** est constitué des liens ou partie de liens en **noire**. Les liens en **gris** ne sont pas pris en compte.

communauté C_i de la partition $C = \{C_j\}_{j_1..k}$ de V en k communautés, cela se traduit par le calcul de la densité entre les communautés, $\delta^{inter}(C_i)$:

$$\delta^{inter}(C_i) = \frac{1}{|C|-1} \sum_{j,i \neq j} \frac{|\{(u,v) \in E \text{ t.q. } u \in C_i \text{ and } v \in C_j\}|}{|C_i| \cdot |C_j|}. \quad (3.1)$$

Il s'agit tout simplement de la probabilité qu'un lien existe entre un nœud de C_i et un nœud d'une autre communauté. Encore une fois, cette notion n'a pas de sens direct dans le formalisme de flot de liens et il est nécessaire de l'adapter. Pour ce faire, nous définissons la Δ -densité inter discussions entre deux discussions D_i et D_j : $\delta_\Delta^{inter}(D_i, D_j)$. Soit $E_\Delta = \xi(E, \Delta)$ et $L_{inter}(D_i, D_j) = (T', V', E')$ avec $V' = V(D_i \cup D_j)$, $T' = [\beta(D_i \cup D_j), \psi(D_i \cup D_j)]$, et $E' = E_\Delta \setminus (D_i \cup D_j)$. Avec ces définitions, $\delta_\Delta^{inter}(D_i, D_j)$ est égale à $\delta_\Delta^{inter}(D_i, D_j) = \delta(L_{inter}(D_i, D_j))$. Il s'agit donc de la densité du flot inter discussions qui est constitué des liens entre les nœuds induits par D_i et D_j qui n'appartiennent ni à D_i ni à D_j . Dans la figure 3.6, un exemple de flot inter discussion est représenté.

Afin d'obtenir la Δ -densité inter discussions entre D_i et tout les autres discussions, nous utilisons la moyenne des densité inter discussion entre D_i et les autres discussions, soit :

$$\delta_\Delta^{inter}(D_i) = \frac{1}{|C|-1} \sum_{j,i \neq j} \delta_\Delta^{inter}(D_i, D_j). \quad (3.2)$$

La distribution cumulative inverse de la Δ -densité inter discussions est présentée dans la figure 3.7 pour différentes valeurs de Δ . Bien que similaire, le comportement de la Δ -densité inter discussions diffère qualitativement de celui de la Δ -densité. La Δ -densité inter discussions croît également en fonction de Δ mais il y a toujours une différence notable entre $\Delta = 1 \text{ mois}$ et $\Delta = 1 \text{ an}$ ce qui n'est pas le cas pour la Δ -densité. Cette différence est normale car lors du calcul de Δ -densité le nombre de liens considérés est fixe peut importe Δ alors qu'il croît avec Δ lors du calcul de Δ -densité inter discussions. Cet effet est visible dans la figure 3.6. Le lien (c, d) qui apparaît peut avant β n'est pas pris en compte si Δ est proche de 0 alors qu'il est en parti pris en compte lorsqu'un Δ plus grand est considéré, ce qui est le cas dans la figure.

Un autre facteur est aussi la duré considérée qui est plus longue que la durée des discussions.

Afin de comparer plus aisément Δ -densité et Δ -densité inter discussions, la corrélation entre ces deux mesures est présentée dans la figure 3.8 pour différentes valeurs de Δ . On remarque que les discussions sont effectivement plus denses intérieurement qu'avec les autres

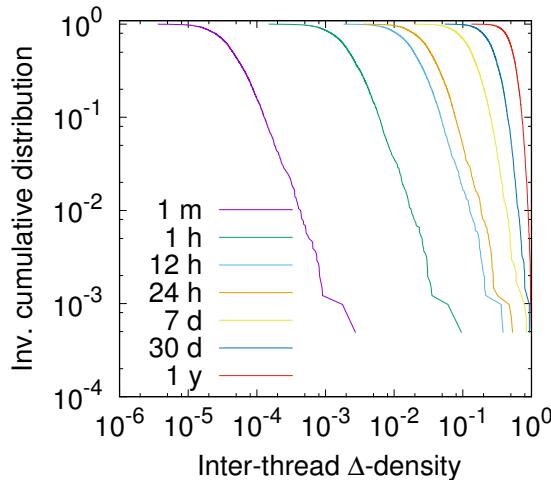


FIGURE 3.7 – Distribution cumulative inverse de la Δ -densité inter discussions pour différentes valeurs de Δ s.

discussions. La différence est de plusieurs ordres de grandeur lorsque Δ est petit et elle diminue lorsque Δ croît. Pour $\Delta = 20 \text{ ans}$ dans la figure 3.8d, la différence n'est plus visible car à cette échelle de temps, l'ancrage temporel des discussions n'est plus décisif. On remarque tout de même que pour $\Delta = 1 \text{ an}$, la différence reste notable.

3.3.2 Répartition temporelle et structurelle des discussions

Nous avons étudié la densité des discussions et entre les discussions mais il est également intéressant d'observer comment ces discussions sont réparties topologiquement et temporellement. Pour étudier la répartition des discussions dans le temps, nous construisons un graphe d'intervalle [LB62] $X = (V_X, E_X)$ représentant le chevauchement temporel. Chaque discussion du flot devient un nœud de V_X et le lien (i, j) existe dans E_X si les discussions D_i et D_j correspondantes ont eu lieu au même instant, *i.e.* $[\alpha_i, \omega_i] \cap [\alpha_j, \omega_j] \neq \emptyset$. De manière similaire, nous définissons le graphe de chevauchement topologique $Y = (V_Y, E_Y)$. Les nœuds de ce graphe représentent encore une fois les discussions du flot et un lien existe entre deux discussions si au moins une personne a participé aux deux, *i.e.* $V(D_i) \cap V(D_j) \neq \emptyset$.

Ces deux graphes sont constitués de 116 999 nœuds et d'environ 2 millions de liens pour le graphe de chevauchement temporel et d'environ 63 millions de liens pour le graphe de chevauchement topologique. Par construction, ces graphes contiennent beaucoup d'informations sur les relations entre les discussions.

Dans la figure 3.9(gauche), est représentée la corrélation entre le degré d'une discussion dans le graphe de chevauchement temporel X et sa durée. Il y a une corrélation évidente entre ces deux notions lorsque les discussions ont une durée supérieur à 10^5 secondes. Plus une discussion dure longtemps, plus elle a de chance d'avoir lieu en même temps que beaucoup d'autres discussions. On observe également que, même pour les discussions durant moins d'un jour (8.6×10^4 s), il peut y avoir jusqu'à une centaine d'autres discussions actives sur la même période.

La figure 3.9(droite) présente la corrélation entre le degré d'une discussion dans le graphe de chevauchement topologique Y et son nombre de participants. La corrélation est moins nette mais il y a tout de même une tendance. Par contre, on remarque de manière frappante que même une petite discussions peut partager des nœuds avec les énormément d'autres discussions.

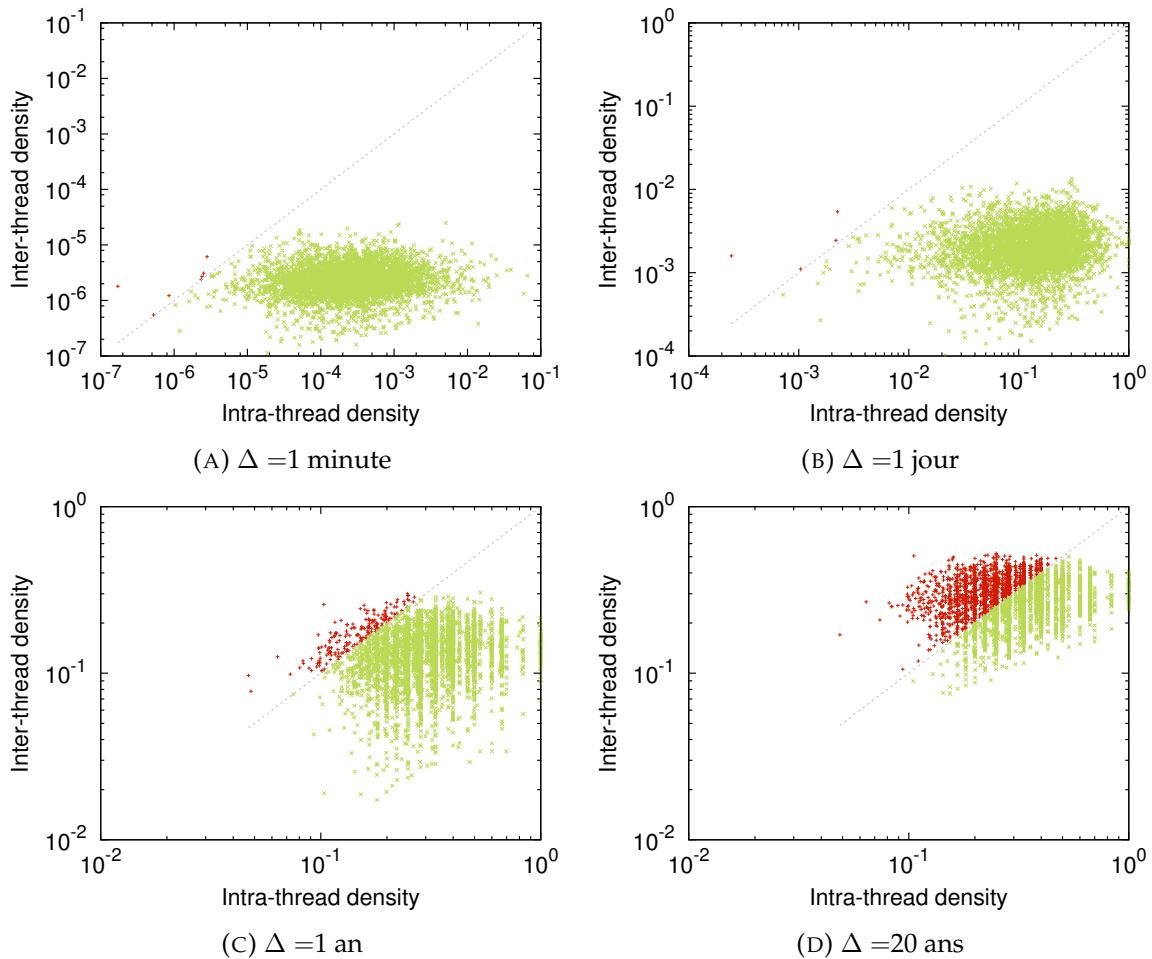


FIGURE 3.8 – Corrélations entre Δ -densité et Δ -densité inter discussions pour différentes valeurs de Δ . Une discussion est en vert (resp. rouge) si elle a une Δ -densité plus (resp. moins) élevée que sa Δ -densité inter discussions.

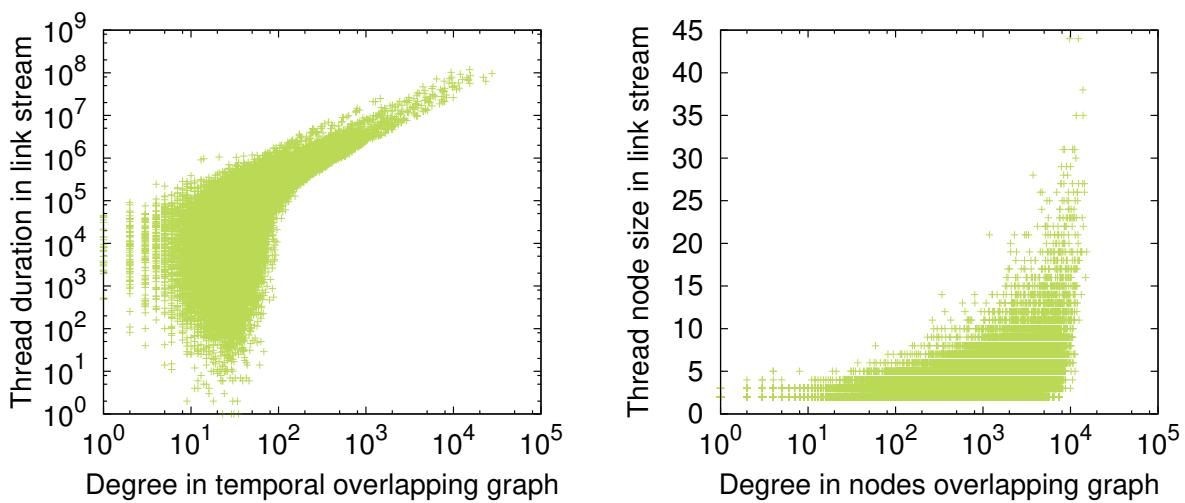


FIGURE 3.9 – Gauche : Corrélation entre le degré des discussions dans le graphe de chevauchement temporel et leur durée. Droite : Corrélation entre le degré des discussions dans le graphe de chevauchement topologique et leur nombre de participants.

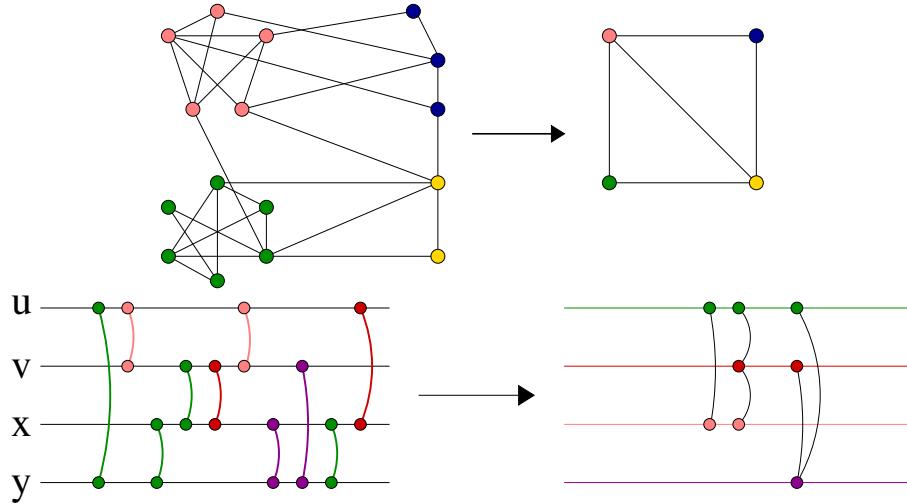


FIGURE 3.10 – Haut : Exemple de graphe ayant une structure communautaire et son graphe quotient associé. Bas : Exemple d'un flot de lien avec une structure ainsi que son flot quotient associé.

3.3.3 Flot quotient

Le graphe quotient est une autre notion clef pour étudier les relations entre les communautés d'un graphe $G = (V, E)$. Soit une partition $C = \{C_i\}_{1..k}$ des nœuds de G en k communautés, chaque communauté est représentée dans le graphe quotient \bar{G} par un nœud dans V . Il y a un lien entre deux communautés C_i et C_j dans E si il existe au moins un lien entre un nœud de C_i et un nœud de C_j . Voir une illustration sur la figure 3.10. Il est possible d'ajouter un poids sur les liens de \bar{G} égale au nombre de liens reliant les communautés. Le graphe quotient permet de facilement étudier, dans un graphe, les relations entre les communautés.

Nous étendons ici cette notion de graphe quotient aux flots de liens. Nous définissons le flot quotient, $Q = (T_Q, V_Q, E_Q)$, induit par une partition $P = \{P_i\}_{1..k}$ en k sous-flots de la manière suivante. Chaque sous-flot P_i est représenté par un nœud dans V_Q . Il existe un lien (t, P_i, P_j) dans E_Q si il existe $(t_1, u, v) \in P_i$, $(t_2, u, v') \in P_j$ et $(t_3, u, v'') \in P_i$ avec $t_1 \leq t_2 \leq t_3$. En d'autre termes, il y a un lien dans E_Q si un nœud u a un lien dans P_j qui apparaît entre deux autres de ses liens du groupe P_i .

Le flot quotient induit par les discussions dans le jeu de données contient 12 281 269 liens impliquant 68 524 discussions différentes. Comme le jeu de données contient 116 999 discussions, il y a donc 48 475 discussions sans lien et qui ne seront pas prises en compte par la suite. Ce nombre de discussions non-reliées est élevé comparé à ce qui est obtenu dans un graphe. En effet dans un graphe, un nœud de degré 0 correspond à une communauté qui est une composante connexe (ou une union de composantes connexes). En ajoutant l'information temporelle, les discussions sont séparées par le temps dans flot. C'est pourquoi un grand nombre de discussions n'ont pas de liens dans le flot quotient. Ce phénomène est d'autant plus vrai pour les petites discussions.

Il faut aussi noter qu'il y a environ 20 fois plus de liens dans le flot quotient que dans le flot initial. Cela est normal car un lien dans le flot peut donner lieu à plusieurs liens dans le flot quotient. Ce cas est visible dans la figure 3.10. Le lien (x, y) du groupe violet du flot à gauche donne lieu au lien (*violet, rouge*) et au lien (*violet, vert*) dans le flot quotient à droite.

La figure 3.11 présente la Δ -densité du flot de liens initial et du flot quotient pour différentes valeurs de Δ . Le flot initial et le flot quotient ont le même comportement de densité mais le flot quotient est moins Δ -dense que le flot initial. Ce résultat diffère par rapport à ce qui est obtenu dans un graphe. Cela est dû au nombre de nœuds qui augmente dans le flot quotient. Mais le

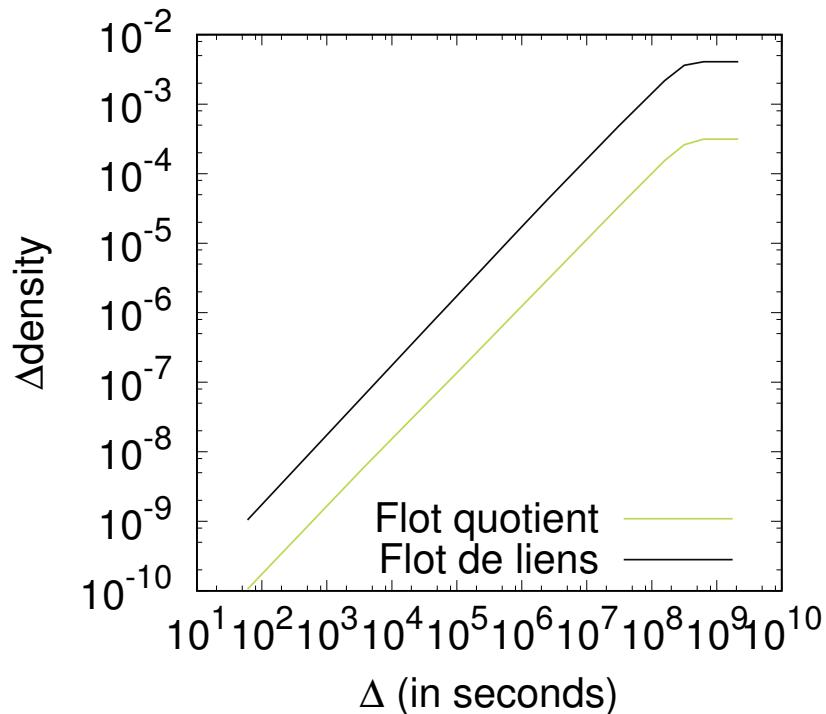


FIGURE 3.11 – Δ -densité du flot de liens et du flot de liens quotient en fonction de Δ pour $\Delta = 1mn, 1h, 12h, 1j, 7j, 30j, 1\text{ an}$ et 20 ans .

flot quotient contient tout de même beaucoup de liens. En effet, le degré moyen dans le flot quotient est moyenne 25 fois plus élevé que dans le flot.

3.3.4 Conclusion

Nous avons utilisé le modèle de flot de liens pour étudier une archive de courriels provenant du projet Debian. Grâce au modèle de flot de liens, nous avons étudié des notions clefs pour mieux comprendre la répartition temporelle et topologique des discussions. Nous avons étudié la notion de Δ -densité sur les discussions en elles mêmes. Puis, nous avons étudié les relations entre les discussions avec la Δ -densité inter discussions, les projections en graphe de chevauchement temporel ou topologique et le flot quotient.

Cette étude repose en grande partie sur la notion de Δ -densité qui nécessite un paramètre fixé arbitrairement. Nous avons à chaque fois testé un ensemble de valeurs de Δ variant d'une seconde jusque parfois 20 ans et, lors de ces tests, aucune valeur Δ caractéristique n'a pu être identifiée. Il semble donc que la Δ -densité soit relativement robuste vis-à-vis de Δ dans ce contexte.

Nous avons tout d'abord observé que les discussions forment une structure plus dense que le flot de liens. De manière encore plus forte, nous avons constaté, grâce à la Δ -densité inter discussion, que les discussions sont plus denses en interne qu'en externe. C'est une caractéristique importante des communautés que l'on trouve dans les graphes mais qui n'avait pas été observée dans un contexte temporel. À partir de ces observations, nous avons également observé les relations entre les discussions. Via le graphe de chevauchement temporel, nous avons validé le fait que différentes discussions ont lieu en même temps et que par conséquent une agrégation temporelle entraînerait une perte d'information. De même via le graphe de chevauchement topologique, on remarque que la structure est très recouvrante sur les noeuds, rendant ainsi l'utilisation de partitions statiques de noeuds difficilement envisageable pour décrire les discussions.

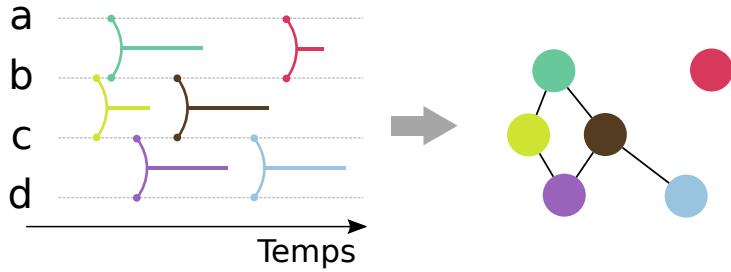


FIGURE 3.12 – Transformation d'un flot de liens avec 4 nœuds (a-d) et 6 liens à gauche en un graphe à droite à 6 nœuds. La couleur d'un nœuds dans le graphe indique le lien du flot qu'il représente.

3.4 Détection de structures denses

À partir du constat que les discussions forment une structure particulière, il est naturel d'essayer de les retrouver automatiquement. Pour y parvenir, il faut un moyen capable de trouver des sous-flots-denses dans le flots. C'est à dire une méthode capturant des groupes de liens qui soient proche temporellement et topologiquement. Il serait tentant d'optimiser directement la densité dans le flot mais ce n'est pas envisageable car un groupe constitué d'un unique lien a une densité de 1. Il faut donc trouver une autre méthode. C'est pourquoi nous avons construit une autre projection du flot en un graphe statique afin d'y appliquer une méthode de détection de communautés. Le problème est alors de réussir à créer une transformation de telle sorte que les informations temporelles et topologiques ne soient pas complètement détruites.

3.4.1 Méthode de détection

Afin de créer une transformation du flot vers un graphe, nous définissons un autre flot de liens, \mathcal{L} , dont les liens ont une durée. À partir de \mathcal{L} , nous créons un graphe non-orienté et non-pondéré $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Chaque lien du flot est représenté par un nœud. Deux liens (b, e, u, v) et $(b', e', u', v') \in \mathcal{L}$ sont connectés dans le graphe s'ils partagent un nœud et si les intervalles s'intersectent, i.e. $\{u, v\} \cap \{u', v'\} \neq \emptyset$ et $[b, e] \cap [b', e'] \neq \emptyset$, voir figure 3.12. Ainsi, un lien dans le graphe représente une connexion structurelle et temporelle entre deux liens du flot de liens. Les groupes denses dans le graphe représentent donc des groupes de liens connectés temporellement et topologiquement dans le flot.

Il faut donc trouver une manière d'ajouter une durée à chaque lien pour créer \mathcal{L} . Lors du calcul de la densité dans la section 3.3.1, nous avions ajouté une durée arbitraire Δ . Ici, il n'est pas très pertinent d'appliquer la même logique. En effet, si on utilise un Δ faible, alors il n'y aura que très peu de liens dans \mathcal{E} et les nœuds représentant les liens d'une discussions ne seront pas forcément connexes. Il paraît illusoire d'espérer retrouver les discussions dans \mathcal{G} si elles ne sont même pas connexes. Si Δ est très grand alors toute information temporelle est perdue et cela revient à calculer le line graphe du graphe agrégé. C'est pourquoi nous adoptons une autre manière d'ajouter une durée sur les liens.

Pour chaque message m , nous connaissons $p(m)$, le message auquel il répond dans la discussion. Nous définissons alors les liens de \mathcal{L} qui ont une durée de la manière suivante : $(t(p(m)), t(m), a(m), a(p(m))_m)$. Ainsi, deux messages, m_1 et m_2 , se succédant dans une discussion sont par définition reliés topologiquement car $a(m_1) = a(p(m_2))$. Ces deux messages

sont aussi reliés temporellement car nous avons la relation suivante :

$$\begin{aligned}[t(p(m_1)), t(m_1)] \cap [t(\mathbf{p}(\mathbf{m}_2)), t(m_2)] &= \\ [t(p(m_1)), t(m_1)] \cap [t(\mathbf{m}_1), t(m_2)] &= [t(m_1)] \neq \emptyset.\end{aligned}$$

Par construction, une discussion est donc représentée dans \mathcal{G} par un ensemble connexe de nœuds. Un fois \mathcal{G} construit, on peut appliquer un algorithme de détection de communautés.

Avec cette construction, \mathcal{G} contient plus d'1 millions de liens entre 316 569 nœuds pour les 116 999 discussions présentes. Sur ce graphe, nous avons appliqué l'algorithme de Louvain [BGLL08] qui optimise la modularité. D'autres algorithmes peuvent également être appliqués s'ils capturent des groupes de nœuds disjoints et qu'ils passent à l'échelle. Les groupes trouvés par Louvain sont des communautés dans \mathcal{G} . Par conséquent, ils sont censé être densément connectés dans \mathcal{G} . Comme un lien de \mathcal{G} correspond à une connexion temporelle et topologique dans le flot, on peut espérer qu'ils correspondent à des groupes denses dans le flot.

3.4.2 Comparaison des partitions

Avant de comparer la structure des discussions, D , et la partition, \mathfrak{D} , trouvée par la méthode de Louvain sur \mathcal{G} , il est nécessaire de décrire cette dernière. Dans la figure 3.13, les distributions cumulatives inverses du nombre de liens, du nombre nœuds et de leur durée sont présentées pour les groupes de \mathfrak{D} . Pour rappel, les même données sont représentées pour les discussions. On remarque tout de suite que \mathfrak{D} contient des groupes beaucoup plus gros en nombre de nœuds et de liens alors qu'ils ont des durées similaires.

Ces deux structures sont donc très différentes mais cela pourrait être dû à l'algorithme de Louvain qui n'est pas adapté pour trouver des groupes denses. C'est pourquoi, nous avons également observé la densité des groupes de D et \mathfrak{D} dans le flot \mathfrak{L} . Le résultat est visible dans la figure 3.13d. Comme les liens de \mathfrak{L} ont une durée, il est possible d'utiliser directement la densité au lieu de la Δ -densité utilisée précédemment. On remarque que les groupes de \mathfrak{D} , bien que plus gros, sont plus denses que les groupes de D . Cependant la distribution cumulative inverse cache les effets de la taille sur la densité. Or à nombre de liens égale (entre 2 et 160), on remarque que les groupes trouvés par Louvain sont plus denses en moyenne ,0.46 contre 0.38. La médiane est également plus élevée : 0.34 contre 0.33. En revanche, les plus gros groupes ($|\mathfrak{D}_j| > 160$) trouvés par Louvain ont une densité plus faible ce qui peut être dû à leur taille.

Si les groupes de \mathfrak{D} sont plus denses, c'est peut être car ils regroupent plusieurs discussions de D dans un groupe. Pour comparer deux partitions, l'indice de Jaccard est classiquement utilisé pour calculer la *précision* et le *rappel* qui sont définis de la manière suivante :

$$\text{précision}(\mathfrak{D}_j) = \max_i \frac{|\mathfrak{D}_j \cap D_i|}{|\mathfrak{D}_j|}, \quad \text{rappel}(D_i) = \max_j \frac{|\mathfrak{D}_j \cap D_i|}{|D_i|}.$$

Dans la figure 3.14, est présenté la *précision* des groupes et le *rappel* des discussions en fonction de leur taille. Chaque point représente la moyenne du *précision* (resp. *rappel*) pour les groupes (resp. discussions) d'une taille donnée. On voit qu'il y a un important rappel et ce même pour les grandes discussions, ce qui veut dire qu'en générale une discussion D_i est totalement incluse dans un groupe \mathfrak{D}_j . En revanche, la précision est très faible car un groupe \mathfrak{D}_j contient plusieurs discussions, ce qui est cohérent avec la taille très importante des groupes de \mathfrak{D}_j .

Il semble donc que la partition \mathfrak{D} soit proche de D mais que ses groupes soient plus gros. Pour circonvenir à ce problème, nous appliquons de manière récursive l'algorithme de Louvain sur chaque graphe induit par un groupe \mathfrak{D}_j . Ce processus permet de subdiviser de manière récursive chaque groupe \mathfrak{D}_j . Soit $\mathfrak{D}_{j'}(h)$ un groupe trouvé au niveau h , par construction, il est

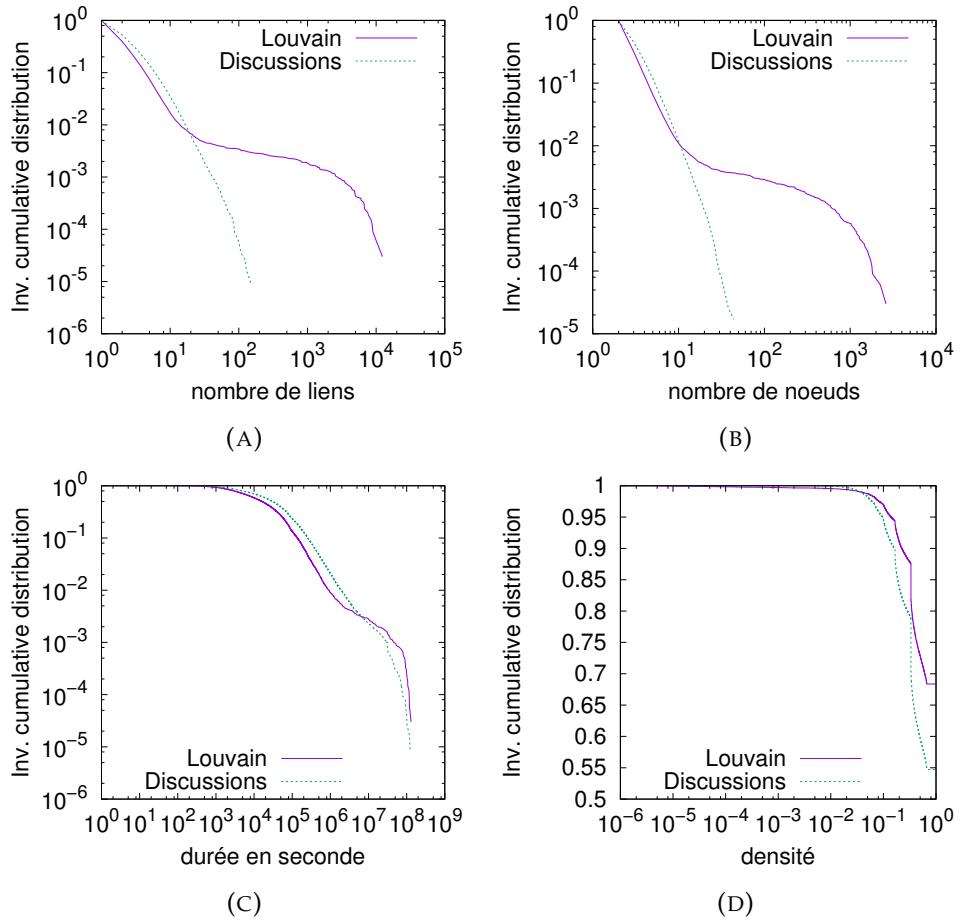


FIGURE 3.13 – Distribution cumulative inverses du nombre de liens (a), du nombre de nœuds (b), de la durée (c) et de la densité (d) pour les groupes trouvés par Louvain et les discussions.

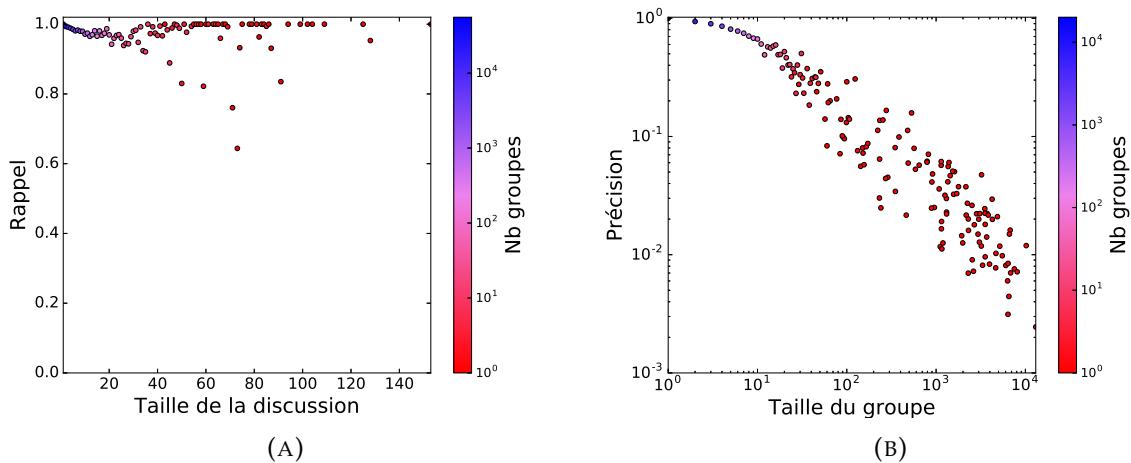


FIGURE 3.14 – (A) Rappel des discussions vis-à-vis des groupes trouvés par Louvain. (B) Précision des groupes trouvés vis-à-vis des discussions. Chaque point représente la moyenne du rappel (resp. précision) pour les discussions (resp. groupes) ayant la même taille. La couleur du point indique le nombre de groupes ayant la même taille.

inclus dans un groupe trouvé au niveau $h - 1$, c'est à dire $\mathfrak{D}_{j'}(h) \subseteq \mathfrak{D}_j(h - 1)$. Le niveau 0 est la première partition trouvée par l'algorithme de Louvain dans \mathfrak{G} .

Soit $D_i \in D$, notons $\mathfrak{D}_{\tilde{j}}(h)$ avec $h \in \mathbb{N}$ le groupe trouvé par la méthode de Louvain au niveau h qui soit le plus proche de D_i au niveau h , c'est-à-dire $|\mathfrak{D}_{\tilde{j}}(h) \cap D_i| = \max_j |\mathfrak{D}_j(h) \cap D_i|$. Avec ces définitions, on observe la relation suivante : $\mathfrak{D}_{\tilde{j}}(h) \cap D_i \subseteq \mathfrak{D}_{\tilde{j}}(h - 1) \cap D_i$. La définition de *rappel* de l'équation 3.4.2 n'est donc pas adaptée pour les niveaux inférieurs et nous l'adaptons de la manière suivante :

$$\text{rappel}(D_i, h) = \max_j \frac{|\mathfrak{D}_j(h) \cap D_i|}{|D_i \cap \mathfrak{D}_{\tilde{j}}(h - 1)|}. \quad (3.3)$$

Ainsi, le *rappel* au niveau h prends en compte le maximum d'élément qu'il est possible de trouver à ce niveau. La définition de *précision* ne pose quant à elle pas de problème. La figure 3.15 représente le *rappel* adapté et la *précision* pour le premier et deuxième niveau de récursion de la même manière que pour la figure 3.14. On remarque que, dès le premier niveau, le rappel baisse et que ce phénomène s'amplifie fortement au niveau suivant. Cela implique que les discussions ne sont plus incluses dans un groupe mais au contraire réparties dans plusieurs. La précision quant à elle augmente légèrement mais cela est dû à la baisse de la taille des groupes trouvés. Il semble donc qu'il ne soit pas possible avec cette approche de retrouver automatiquement les discussions.

3.4.3 Conclusion

Nous avons avec cette méthode mis en évidence des groupes denses. Les groupes trouvés sont plus gros et plus denses que la structure des discussions. Cependant, ces observations ne remettent pas en cause les conclusions faites dans la section 3.3 pour plusieurs raisons. Tout d'abord, les flots de lien étudiés ne sont pas exactement les-mêmes ($L \neq \mathfrak{L}$). Ce changement de flot est nécessaire pour le fonctionnement de la méthode de détection. Ensuite, les deux structures ne sont pas complètement différentes car les groupes trouvés semblent en fait agréger plusieurs discussions. Malheureusement, nous n'avons pas réussi avec notre méthode à isoler chaque discussion malgré notre approche récursive. Pourtant, il semble que la structure trouvée ai du sens au vu des valeurs de densité des groupes.

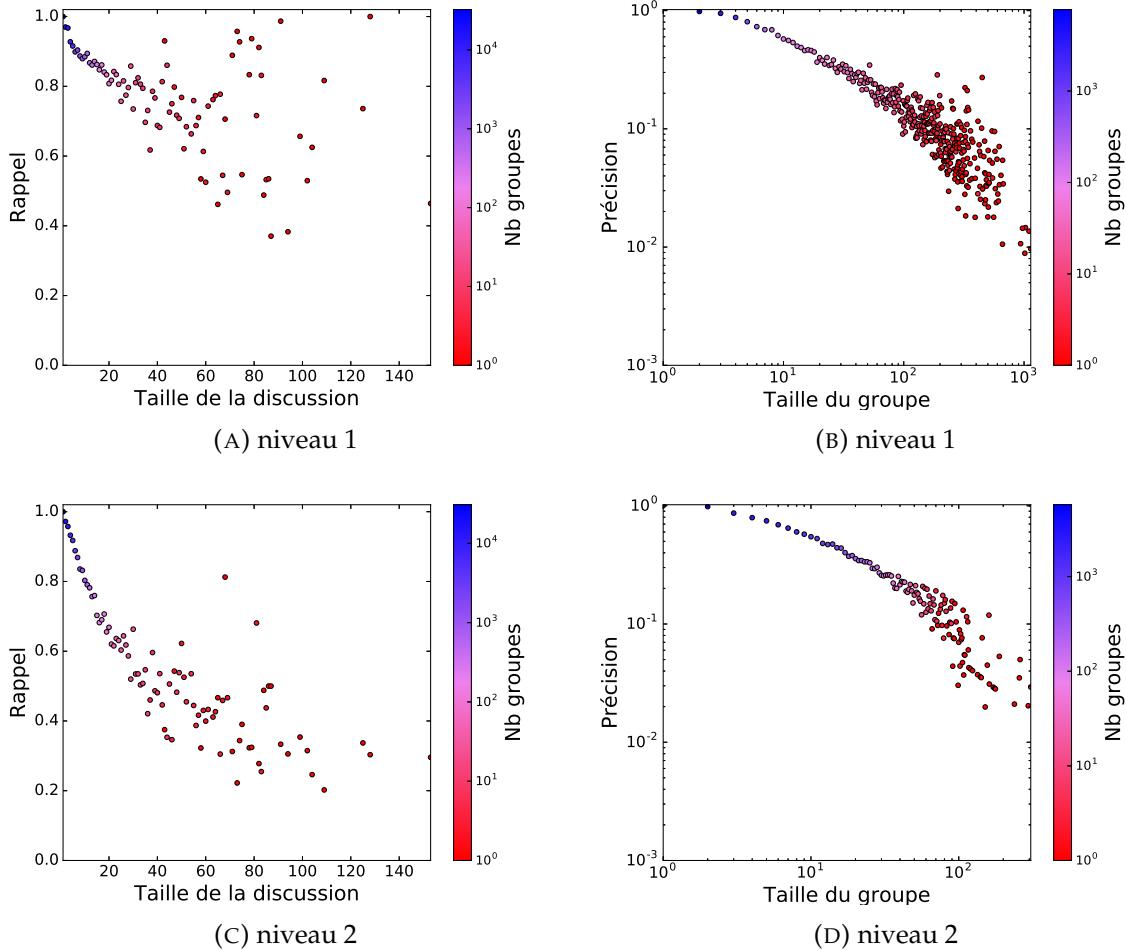


FIGURE 3.15 – (A,B,C) Rappel des discussions vis-à-vis des groupes trouvés par Louvain à différent niveaux récursif. (B,D,F) Précision des groupes trouvés vis-à-vis des discussions. Chaque point représente la moyenne du rappel (resp. précision) pour les discussions (resp. groupes) ayant la même taille. La couleur du point indique le nombre de groupes ayant la même taille.

Chapitre 4

Détection de groupes denses (SNAM)

Sommaire

4.1	Calcul des groupes candidats	39
4.2	Calcul évaluation	39
4.3	Jeux de données	39
4.4	Application	39
4.5	Conclusion	39

4.1 Calcul des groupes candidats

4.2 Calcul évaluation

4.3 Jeux de données

4.4 Application

4.5 Conclusion

Chapitre 5

Expected Nodes : communautés de liens dans les graphes statiques

Sommaire

5.1	Travaux existants	42
5.2	Définition d'Expected Nodes	45
5.3	Comparaison	48
5.3.1	Cas du graphe complet	48
5.3.2	Graphe LFR	49
5.4	Calcul et optimisation	52
5.5	Conclusion	53
5.5.1	Perspective	54

Les structures communautaires dans les graphes ont été beaucoup étudiées lorsqu'elles concernent les noeuds, voire chapitre 1. Dans une moindre mesure, elles ont également été étudiées lorsqu'elles concernent les liens. Par exemple dans un réseau social, chaque personne a plusieurs centres d'intérêts : famille, sport, politique... Lorsque deux personnes interagissent, la communication a lieu dans un contexte bien particulier. Bien que les personnes aient plusieurs centres d'intérêts, la raison de leur communication est souvent unique. Il semble donc qu'une information importante soit intrinsèquement liée au lien. Via la recherche de partitions de liens d'un graphe, c'est cette information que nous cherchons à capturer.

Afin d'illustrer ce que capture une partition, prenons l'exemple d'un réseau de personnes fictif où le contexte de l'interaction est connue. Certaines personnes communiquent ensemble car elles sont de la même **famille**, pratiquent le même **sport**, jouent au **go** ensemble ou bien encore car elles **travaillent** ensemble. Nous illustrons cet exemple dans la figure 5.1 où les

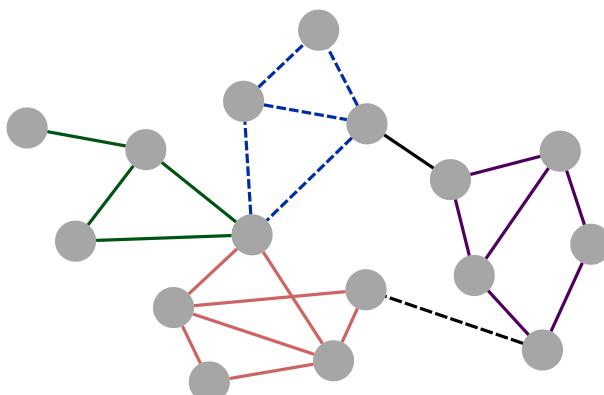


FIGURE 5.1 – Exemple de réseau de personnes avec une structure communautaire sur les liens qui est représentée par la couleur et le style de chaque lien.

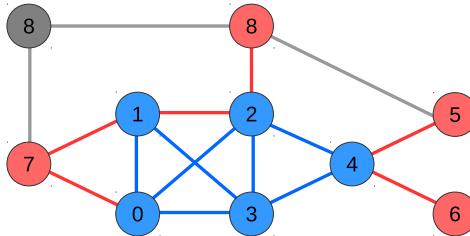


FIGURE 5.2 – Exemple d'un groupe de liens L (en bleu). Les liens rouges sont les liens adjacents L_{out} connectant les nœuds internes V_{in} (en bleu) aux nœuds adjacents V_{out} (en rouge).

interactions sont colorées en fonction de leur contexte. Il est intéressant de noter que les interactions d'un même type se regroupent ensemble. Un nœud peut avoir des interactions de plusieurs types, c'est le cas du nœud central qui a des interactions de type **famille**, **sport** et **go**. De même, certaines interactions font le lien entre différent groupes. C'est le cas du lien pointillé **noir** qui relie le **sport** et le **travail** ce qui pourrait se traduire par le financement de l'équipe par l'entreprise. Ainsi, les partitions de liens peuvent capturer des situations assez variées.

Il est également possible de manipuler des partitions chevauchantes ou couvertures. Dans ce cas, chaque nœud peut appartenir à plusieurs communautés. Pour répondre à ce problème de nombreux algorithmes ont été proposés pour la détection et l'évaluation de couvertures de nœuds, voir la section 1.2.2. Les couvertures sont une généralisation des partitions et aucune méthode ne fait encore consensus pour les évaluer car leur structure est complexe, voir la sous-section 1.2.2. Les partitions de liens, quant à elles, restent des objets plus simples à manipuler. De plus, elles permettent de mettre en avant une autre structure ayant également du sens.

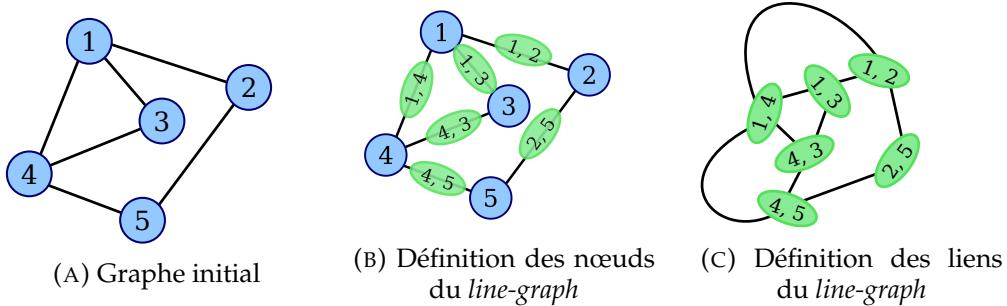
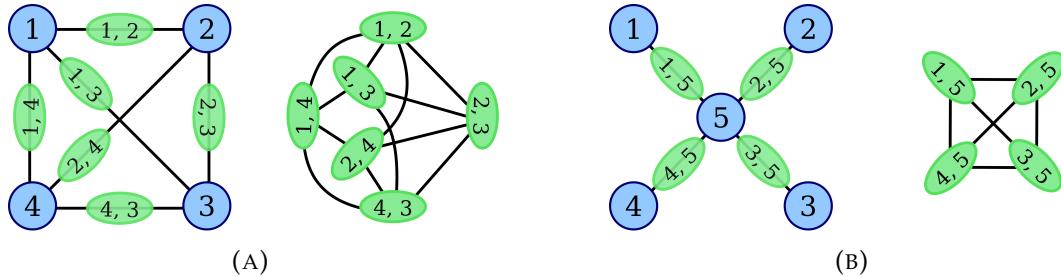
Il apparaît donc que les partitions de liens sont des objets à part entières pertinent à étudier. Pour ce faire, il est nécessaire d'adapter les outils d'analyses pour évaluer directement les partitions de liens. Nous développons ici une approche similaire à ce qui est fait pour les partitions de nœuds et la modularité [NG04]. Le but est de créer une fonction de qualité permettant d'évaluer une partition de liens d'un graphe.

5.1 Travaux existants

Les notations utilisées sont les suivantes. Soit $G = (V, E)$ un graphe non-orienté avec V l'ensemble des nœuds de taille n et $E \subseteq V \times V$ l'ensemble des liens de taille m . Le degré d'un sommet u de G est noté $d_G(u)$. Une partition des liens en k groupes est notée $\mathcal{L} = (L_1, L_2, \dots, L_k)$ avec $L_i \subseteq E \forall i, L_i \cap L_j = \emptyset \forall i \neq j$ et $\bigcup_i L_i = E$. Pour un groupe de liens $L \in \mathcal{L}$, on pose $V_{in} = \{u \in V, \exists (u, v) \in L\}$ l'ensemble des nœuds internes au groupe L , $V_{out} = \{u \in V \setminus V_{in}, (u, v) \in E \wedge v \in V_{in}\}$ représente les nœuds adjacents au groupe L et enfin $L_{out} = \{(u, v) \in E \setminus L, u \in V_{in} \vee v \in V_{in}\}$ l'ensemble des liens adjacents au groupe L (voir Figure 5.2).

Une approche naïve serait de transformer le graphe initial en un *line-graph*. Un *line-graph* est un graphe où chaque lien du graphe initial est transformé en un nœud dans le *line-graph*. Deux nœuds du *line-graph* sont reliés si les liens correspondants ont au moins un nœud en commun, voir la figure 5.3. Comme un *line-graph* est un graphe classique, on peut y appliquer toutes les méthodes déjà existantes, e.g. les algorithmes de détection de communautés. Ainsi, une partition des nœuds du *line-graph* représente une partition des liens du graphe initiale. Sur l'exemple 5.3, les liens $(1, 4)$, $(1, 3)$ et $(3, 4)$ du graphe forment un triangle dans le *line-graph* et pourraient être capturés comme étant une communauté. Ces liens dans le graphe initial forme également un triangle et peuvent être considérés comme une communauté valide.

1. Image provenant de https://en.wikipedia.org/wiki/Line_graph.

FIGURE 5.3 – Exemple de construction du *line-graph*¹.FIGURE 5.4 – Exemple de construction du *line-graph* d'une clique en (A) et d'une étoile en (B).

Cependant pour que ce type de méthode fonctionne, il est nécessaire que le *line-graph* résultant puisse être analysé comme un graphe. En particulier, il est nécessaire que la notion de communauté dans le *line-graph* est un sens dans le graphe initial. Or, un *line-graph* a une structure très différente du graphe initial.

Prenons pour l'exemple : la clique qui est la meilleure communauté possible et l'étoile qui est une des pires communautés possible. Le but est d'observer comment ses structures sont transformées dans le *line-graph*. Ces situations sont représentées dans la figure 5.4. L'étoile dans la figure 5.4b est transformée en une clique de 4 noeuds. Une des pires structures communautaires d'un graphe est transformé dans le *line-graph* en la meilleure structure communautaire. En effet, chaque noeuds de degré k du graphe initial donne lieu à une clique de taille k dans le *line-graph*. Les cliques du *line-graph* ne sont donc pas forcément des communautés dans le graphe. Dans le cas de la clique 4 dans la figure 5.4a, on remarque que le *line-graph* est composé de 6 noeuds et de uniquement 12 liens. Plus généralement une clique de taille n dans le graphe initial donne lieu dans le *line-graph* à $\frac{n(n-1)}{2}$ noeuds et $(n-1)n$ liens. Ainsi, plus la clique est grande dans le graphe, moins la structure résultante dans le *line-graph* est dense. Les cliques du graphe sont donc moins denses que ses étoiles, lorsqu'on les observe dans le *line-graph*.

Il n'est donc pas innocent d'utiliser le *line-graph* pour trouver des communautés de liens. Il est nécessaire d'adapter les outils à ce type de graphe.

Il existe différents types de méthodes pour la détection et l'évaluation de partitions de liens. Yuet al. [YWW13] ont étendu l'algorithme de propagation de label sur les liens ce qui permet de capturer une partition de liens. Cependant cela ne permet d'évaluer le résultat obtenu.

Il y a également les méthodes évaluant une partition de liens via la transformation de la partition en une couverture de noeuds [HWW⁺13, LRK⁺14, WLWT10]. Un transformation classiquement utilisée est qu'un noeud dans la couverture prend comme communautés l'ensemble des communautés de ses liens, voir la figure 5.5. Il serait tentant de considérer que les partitions de liens et les couvertures de noeuds sont équivalentes. Ainsi pour évaluer une partition de liens, il suffirait de transformer la partition en couverture. Or, ce changement n'est pas anodin.

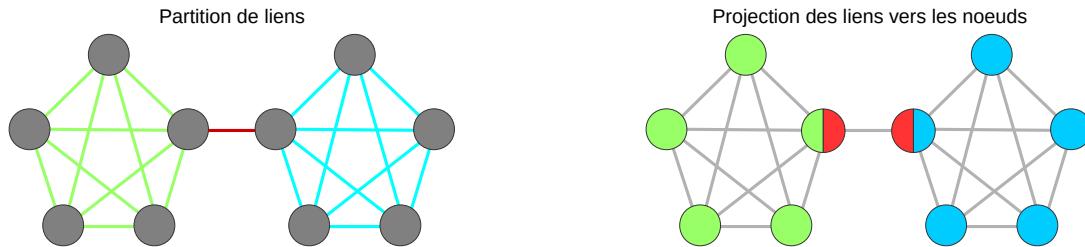


FIGURE 5.5 – Transformation d'une partition de liens à gauche en couverture de nœuds à droite. La couleur représente un groupe.

D'une part, les couvertures de nœuds permettent de modéliser beaucoup plus de situations car il n'y a aucune contrainte sur les couvertures. D'autre part, il n'est pas trivial de transformer une partition de liens en couverture de nœuds, et *vice versa*.

Dans l'exemple de la figure 5.5, il n'est évident pas que la communauté **rouge** constituée des deux nœuds centraux soit une communauté légitime. Selon le contexte, elle peut être considérée comme un artefact de la transformation. La transformation d'une partition n'est donc pas un acte neutre. Cet aspect a d'ailleurs été mis en avant par Esquivel *et al* [ER11]. Face à ce problème, nos travaux ainsi que quelques méthodes existantes proposent des méthodes évaluant directement les partitions de liens.

Ahn *et al.* [ABL10] sont parmi les premiers à avoir proposé une méthode détectant les communautés de liens. Leur méthode *link clustering* est une méthode hiérarchique d'agglomération. Elle construit un dendrogramme en agglomérant de manière itérative les groupes de liens en fonction de leur similarité calculée par l'indice de Jaccard. Afin de décider de la coupe du dendrogramme et de la partition résultante, la fonction *Partition Density* est utilisée. Pour une partition de liens donnée \mathcal{L} , la *Partition Density* est définie de la manière suivante :

$$D(\mathcal{L}) = \frac{\sum_{L \in \mathcal{L}} |L| D(L)}{m} \quad D(L) = \frac{|L| - \min_D(|V_{in}|)}{\max_D(|V_{in}|) - \min_D(|V_{in}|)}, \quad (5.1)$$

où $\min_D(N) = N - 1$ est le nombre minimum de liens pour relier N nœuds et $\max_D(N) = \frac{N(N - 1)}{2}$ est le nombre maximum de liens qu'il puisse exister entre N nœuds. Malgré son nom la *Partition Density* n'est pas une densité mais le nombre de liens du groupe normalisé par le nombre de liens minimum et maximum pour un groupe de $|V_{in}|$ nœuds.

Après simplification, on obtient la formule suivante :

$$D(L) = 2 \frac{|L| - (|V_{in}| - 1)}{(|V_{in}| - 1)(|V_{in}| - 2)}. \quad (5.2)$$

Par convention, un groupe constitué d'un unique lien, et qui n'a donc que deux nœuds internes, a une qualité nulle.

D'autres chercheurs [LZW⁺13, SCF⁺13] ont par la suite utilisé la *Partition Density* comme fonction à optimiser dans des algorithmes génétiques. Leurs solutions semblent pour l'instant difficilement utilisable car leurs algorithmes reposent sur de nombreux critères et sont limités à de petits graphes.

Par ailleurs, la *Partition Density* ne peut pas être directement appliquée aux graphes pondérés. Une première proposition de Kim [Kim14] a été faite dans ce sens.

Evans *et al.* [EL09] proposent trois fonctions de qualité pour évaluer les partitions de liens. Leurs fonctions de qualité sont basées sur trois marches aléatoires qui se déroulent sur les liens du graphe. L'approche est similaire à la modularité car la modularité peut également être définie à l'aide de marche aléatoire sur les nœuds du graphe [DYB10]. Leurs trois fonctions

de qualité peuvent être calculées et optimisées sur le graphe mais les auteurs ont montré que l'on pouvait, de manière complètement équivalente, utiliser la modularité sur des *line-graph* pondérés (LG_1 , LG_2 , LG_3). Ainsi, il suffit de construire le *line-graph* approprié puis d'utiliser un algorithme existant d'optimisation de la modularité tel que l'algorithme de *Louvain* [BGLL08].

Pour construire les *line-graphs* LG_1 , LG_2 et LG_3 , nous définissons $B \in \mathcal{M}_{n,m}$ la matrice d'incidence du graphe G : un élément $B_{i\alpha}$ de cette matrice $|V| \times |E|$ est égale à 1 si le lien α est relié au nœud i et 0 sinon. Les matrices LG_1 , LG_2 et LG_3 sont alors définies de la manières suivante :

	$x = 1$	$x = 2$	$x = 3$
$LG_x(\alpha, \beta)$	$B_{i\alpha}B_{i\beta}(1 - \delta_{\alpha\beta})$	$\sum_{i \in V, d_G(i) > 1} \frac{B_{i\alpha}B_{i\beta}}{d(i) - 1}$	$\sum_{i,j \in V, d(i)d_G(j) > 0} \frac{B_{i\alpha}A_{ij}B_{j\beta}}{d(i)d(j)}$

Soit $k_x(\alpha) = \sum_\beta LG_x(\alpha, \beta)$ le degré pondéré dans le line graphe LG_x du nœud représentant le lien α et $W_x = \sum_{\alpha, \beta \in |E|} LG_x(\alpha, \beta)$ la somme des poids des liens. Pour $x \in \{1, 2, 3\}$, la fonction de qualité $Evans_x$ est définie de la manière suivante :

$$Evans_x(\mathcal{L}) = \frac{1}{W_x} \sum_{L_i \in \mathcal{L}} \sum_{e_1, e_2 \in L_i^2} LG_x(e_1, e_2) - \frac{k_x(e_1)k_x(e_2)}{W}. \quad (5.3)$$

Kim *et al.* [KJ11] ont exploré une extension du concept de *Minimum Length Description* (MDL) introduit par Rosvall *et al.* [RB08] qui est une méthode provenant de la théorie de l'information. Cette extension de la MDL évalue directement une partition de liens, contrairement à l'extension proposée par Esquivel *et al.* [ER11]. Un avantage de leur méthode est de pouvoir comparer la qualité d'une partition de liens et d'une partition de nœuds avec leur MDL respective. Cependant, leur méthode ne semble favoriser les communautés de liens que dans des cas très limités.

Résumé

Il existe des méthodes pour capturer et évaluer des partitions de liens dans un graphe. Deux d'entre elles semblent faire consensus pour l'instant. D'une part, la *Partition density* est une fonction de qualité comparant le nombre de liens observé avec les nombres minimum et maximum de liens possibles entre les mêmes nœuds induits. D'autre part, les fonctions de qualité $Evans_x$ se basent quand à elles sur des marches aléatoires sur les liens et d'une certaine manière sur un processus similaire à la modularité. Il n'existe cependant aucune méthode utilisant une comparaison d'une métrique à ce qui est attendu dans un modèle nul. Or, ce processus a été à l'origine de la modularité.

5.2 Définition d'Expected Nodes

Une idée souvent utilisée lors de la détection de communautés de nœuds est qu'une communauté devrait avoir beaucoup de connexion en interne. Pour évaluer ce genre de définition intuitive, il est nécessaire de définir à quoi comparer le nombre de connexions interne. Le choix qui est fait avec la modularité et d'autres méthodes est de définir un modèle nul aléatoire où il n'existe pas de structure communautaire. Le but est de construire un graphe aléatoire qui partage un certain nombre de caractéristiques avec le graphe initial mais dont la structure communautaire a été détruite lors du mélange.

Il existe de nombreux modèles et celui utilisé dans la modularité est le modèle de configuration [BC78]. Dans ce modèle, le nombre de nœuds et leurs degrés sont fixes mais la répartition des liens est aléatoire. Ainsi pour chaque nœud, ses voisins sont tirés de manière aléatoire avec une probabilité proportionnelle à leur degré. Comme les liens sont mélangés dans ce modèle, on suppose qu'il n'existe plus de structure communautaire dans le graphe.

Pour notre fonction de qualité *Expected Nodes*, nous utilisons également le modèle de configuration. Avant d'aller plus loin et de définir formellement *Expected Nodes*, il est utile d'avoir une définition informelle de la fonction de qualité.

Le but est d'évaluer un groupe de liens. Afin qu'un groupe de liens soit évalué comme une bonne communauté, les liens devraient induire un nombre relativement faible de nœuds internes. En effet, plus le nombre de nœuds internes est faible, plus le groupe de liens ressemble à une clique. De manière similaire à la modularité, nous utilisons le configuration modèle pour calculer le nombre de nœuds interne espéré dans le modèle de configuration. Si le groupe de liens a moins de nœuds internes qu'espéré alors ça indique que le groupe de liens est plus dense et qu'il devrait donc avoir une évaluation élevée.

Il est donc nécessaire de calculer l'espérance du nombre de nœuds interne, μ_G , d'un groupe de liens, L , dans le modèle nul. Un nœud de G est interne si au moins un de ses liens appartient à L . Ainsi pour calculer μ_G , il faut tirer aléatoirement et sans remise $2|L|$ demi-liens parmi les $2|E|$ demi-liens du graphe aléatoire avec la même distribution de degrés. Soit B_u la variable aléatoire correspondant au nombre de fois où le nœud u est tiré. Cette variable suit une loi hypergéométrique $B_u \sim \mathcal{G}(2|E|, d_G(u), 2m)$. Avec cette notation, on définit μ_G de la manière suivante :

$$\mu_G(|L|) = \sum_{u \in V} \mathbb{P}(B_u \geq 1) = \sum_{u \in V} 1 - \frac{\binom{2|E|-d(u)}{2|L|}}{\binom{2|E|}{2|L|}}. \quad (5.4)$$

Voici quelques propriétés de la fonction $\mu_G(|L|)$:

- La fonction μ_G dépend uniquement de la séquence de degrés $\{d_G(v)\}_{v \in V}$ et du nombre de liens.
- Pour une distribution de degrés donnée, la fonction $\mu_G(|L|)$ est une fonction croissante de $|E|$.
- Si $L = E$, alors le nombre de nœuds attendus est bien égal à $|V|$.
- On a $\mu_G(1) \leq 2$, en effet le modèle nul n'interdit pas la présence de boucle.

Avec μ_G , nous pouvons définir la qualité *interne*, Q_{in} d'un groupe de liens L :

$$Q_{in}(L) = \frac{\mu_G(|L|) - |V_{in}(L)|}{\mu_G(|L|)}. \quad (5.5)$$

Avec cette formulation, pour un groupe de taille $|L|$, plus le nombre de nœuds internes est faible, plus Q_{in} sera élevée.

Q_{in} permet d'évaluer la qualité interne d'un groupe mais il faut aussi tenir compte du voisinage. En effet, observer une clique à l'intérieur d'une autre clique n'est absolument pas surprenant. C'est pourquoi, nous définissons également une qualité externe. Le but est d'évaluer comment sont répartis les liens et nœuds adjacents. Pour ce faire, nous allons également comparer le nombre de nœuds adjacents observé au nombre espéré dans le modèle de configuration. Cependant à l'inverse de la qualité interne, la qualité externe est **mauvaise** si jamais le nombre de nœuds adjacent est plus faible qu'espéré. En effet, si il y a beaucoup de liens adjacents pour peu de nœuds, alors cela indique que le voisinage du groupe est dense et devrait être inclus dans le groupe. Le cas idéal est que chaque lien adjacent soit relié à un nœud différent.

Soit $\bar{d}(L, u) = \sum_{v \in V} \mathbf{1}_{(u,v) \in E \setminus L}$ le degré de u limité aux liens adjacents et $\bar{d}(L) = \sum_{u \in V_{in}(L)} \bar{d}(L, u)$. L'espérance du nombre de nœuds adjacents est calculé comme le nombre de nœuds tirés lorsque $\bar{d}(L)$ demi-liens sont choisis aléatoirement et sans remise dans le modèle de configuration où les liens de L ont été préalablement retirés. Ce graphe aléatoire a la distribution de degrés suivante : $\{d_{G \setminus L}(u)\}_{u \in V}$ où $G \setminus L = (V, E \setminus L)$. Dans ce cas, on ne tire pas aléatoirement un lien mais uniquement un demi-lien car l'autre demi-lien est un des demi-liens reliés aux nœuds internes. L'espérance du nombre de nœuds adjacents se définit de la manière suivante :

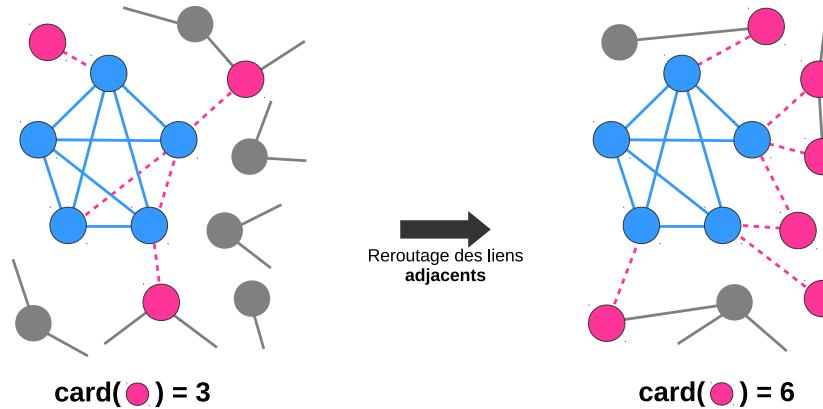


FIGURE 5.6 – Groupe de liens L en bleu et ces liens adjacents en rose pointillés dans le graphe initial à gauche. À droite, une réalisation du modèle de configuration où L a été figé.

$$\mathbb{E}[\bar{d}(L)] = \mu_{G \setminus L}(\bar{d}(L)/2). \quad (5.6)$$

Une illustration de ce processus est présentée dans la figure 5.6. Sur cette illustration, le groupe L a un très mauvais voisinage et cela se reflète par un nombre de noeuds adjacents observés plus faible qu'espéré.

Comme il est intéressant de pénaliser les groupes ayant de mauvais voisinage mais qu'un bon voisinage n'est pas suffisant pour définir une bonne communauté, nous bornons à 0 la qualité externe :

$$Q_{ext}(L) = \min \left(0, \frac{|V_{out}(L)| - \mu_{G \setminus L}(\bar{d}(L)/2)}{\mu_{G \setminus L}(\bar{d}(L)/2)} \right). \quad (5.7)$$

Enfin, nous définissons *Expected Nodes* pour un groupe L :

$$Q(L) = 2 \frac{|L|Q_{in}(L) + |L_{out}|Q_{ext}(L)}{|L| + |L_{out}|}. \quad (5.8)$$

La qualité interne est due aux liens de L et la qualité externe aux liens adjacents. C'est pourquoi nous pondérons Q_{in} par $|L|$ et Q_{ext} par $|L_{out}|$.

Nous détaillons maintenant certaines propriétés des formules 5.7 et 5.8 découlant des propriétés de μ_G en nous appuyant sur des exemples. En s'intéressant aux noeuds adjacents V_{out} , on pénalise la présence de noeuds adjacents fortement connectés avec les noeuds incidents à L . Prenons le cas extrême où L est une clique qui est incluse dans une plus grande clique alors que le reste du graphe est quelconque. Dans cette situation, $Q_{in}(L)$ est maximum car le groupe est une clique. En revanche, $Q_{ext}(L)$ va fortement pénaliser la qualité du groupe car il y a dans ce cas beaucoup de liens adjacents pour très peu de noeuds adjacents. Ainsi, la Q_{ext} permet de pénaliser les noeuds adjacents ayant beaucoup de liens avec L . L'idée est que ces liens adjacents devraient être intégrés à L .

Bien évidemment, la solution n'est pas de systématiquement d'intégrer les liens adjacents car intégré un lien adjacent peut également faire baisser Q_{in} . Par exemple, un lien reliant deux groupes denses disjoints peut avoir une qualité positive. Cette situation est visible dans la figure 5.5 partie gauche. Ce lien tout seul a d'une part une qualité interne positive car $\mu_G(1) \leq 2$ et d'autre part il a une qualité externe nulle car chaque noeud adjacent n'est relié qu'à un seul lien adjacent ce qui est le cas idéale. Dans une situation plus générale, la qualité d'un lien isolé

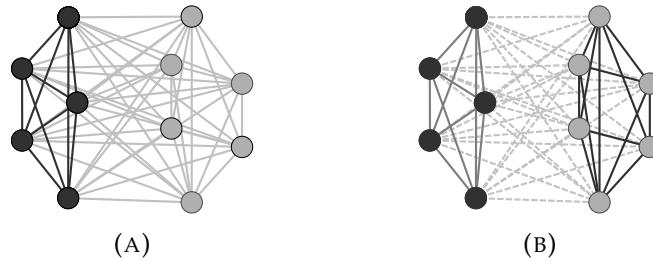


FIGURE 5.7 – Deux partitions de liens pour un graphe complet à 10 nœuds avec $p = 5$: (a) partition en deux groupes et (b) partition en trois groupes. Les nœuds noirs sont les nœuds appartenant à V' et la couleur d'un lien correspond à son groupe.

dépend du nombre de triangles dans lequel il se trouve. Moins le nombre de triangle est élevé, meilleure sera la qualité externe.

Enfin il est intéressant de noter que la qualité du groupe contenant tout les liens est nulle. En effet comme dit précédemment, $Q_{in}(E)$ égal à zéro, voir l'équation 5.4. De plus, si le groupe contient tout les liens alors il n'y a plus de liens adjacents et donc seule $Q_{in}(E)$ influe sur $Q(E)$.

Nous définissons *Expected Nodes* pour une partition de liens \mathcal{L} comme la moyenne pondérée de la qualité de chaque groupe :

$$Q_G(\mathcal{L}) = \frac{\sum_{L \in \mathcal{L}} |L| Q(L)}{|E|}. \quad (5.9)$$

D'autres choix de pondération pour Q_{in} , Q_{ext} et Q_G ont été testés en utilisant le nombre de nœuds au lieu du nombre de liens mais elles ont été abandonnées lors des tests qui sont présentés dans la section 5.3.

5.3 Comparaison

Nous évaluons maintenant *Expected Nodes* en utilisant deux jeux de test. Sur ces jeux de tests, nous appliquons également des fonctions de qualités reconnues : *Partition Density* [ABL10] et les fonctions de qualité proposées par Evans *et al.* [EL09] que nous nommons *Evans1*, *Evans2* et *Evans3*. Pour chaque graphe de test, nous créons empiriquement plusieurs partitions de liens et nous évaluons chaque partition avec toutes les fonctions de qualité.

5.3.1 Cas du graphe complet

Le premier jeu de test est assez simple puisqu'il s'agit d'un graphe complet. Le but est de vérifier que *Expected Nodes* n'a pas un comportement dégénéré. Nous étudions un graphe complet de 100 nœuds². Sur ce graphe, nous définissons plusieurs partitions. La première est la partition triviale où tout les liens sont dans un unique groupe. Nous définissons également deux familles de partitions : une séparant les liens en deux groupes et une séparant les liens en 3 groupes. Soit V' un ensemble de p nœuds où p est un paramètre $p < |V|$. Les deux familles de partitions placent les liens de $V' \times V'$ dans un groupe. Pour la partition en 2 groupes, tout les autres liens sont mis dans un second groupe. Pour la partition en 3 groupes, les liens de $V \times V \setminus V'$ sont dans un second groupe et le reste dans un troisième. Ces répartitions sont illustrées dans la figure 5.7.

Comme le graphe est un graphe complet, la meilleur solution est d'avoir un seul groupe contenant l'ensemble des liens, *i.e.* la partition triviale devrait avoir une meilleure évaluation

2. Nous avons obtenus des résultats similaires pour un graphe de 500 nœuds.

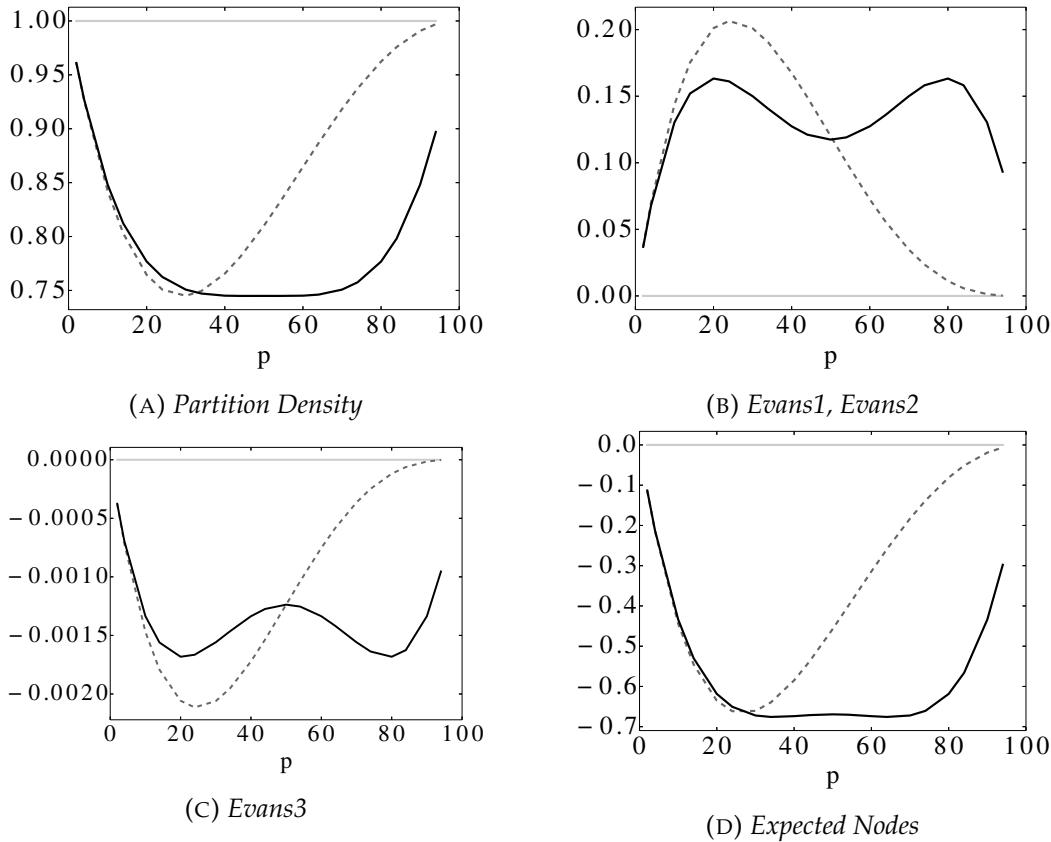


FIGURE 5.8 – Évaluation des 5 fonctions de qualité sur un graphe complet de 100 nœuds pour trois type de partitions. Les partitions testées sont présentées dans la section 5.3.1. Par définition, les résultats pour *Evans1* et *Evans2* sont identiques. Les lignes en gris, noir et pointillées représentent respectivement la partition triviale, les partitions en deux groupes et les partitions en trois groupes.

que les autres partitions. La figure 5.8 présente les résultats. Pour chaque valeur de p et chaque fonction de qualité, nous calculons les évaluations des partitions en deux et en trois groupes ainsi que l'évaluation de la partition triviale. L'évaluation de la partition triviale n'est pas dépendante de p et n'est calculée qu'une seule fois. Tout d'abord, la construction de cet exemple est symétrique pour la partition en deux groupes. En effet, la partition en deux groupes lorsque p égal 24 est complètement équivalente à la partition en deux groupes lorsque p égal 76. Par ailleurs, il est possible de définir formellement la qualité de chacune de ses partitions selon toutes les fonctions de qualité. Cependant, la complexité des équations, surtout pour *Expected Nodes* et *Evans_x*, rend le résultat difficilement interprétable.

De manière assez surprenante, les fonctions *Evans1* et *Evans2* ne passent pas ce test car elles évaluent la partition en deux ou trois groupes comme meilleure que la partition triviale. Selon la *Partition Density*, *Expected Nodes* et *Evans3*, la partition triviale est la meilleur des partitions. La fonction *Evans3* diffère légèrement des deux autres car elle a une amplitude plus faible ($\approx 10^{-3}$).

5.3.2 Graphe LFR

Nous utilisons maintenant un jeu de test plus évolué. Il n'existe pas à notre connaissance de générateur de graphe aléatoire avec une structure communautaire sur les liens. C'est pourquoi, nous utilisons le générateur proposé par Lancichinetti *et al.* [LF09a]. Ce générateur aléatoire permet de générer des graphes ayant une structure communautaire chevauchante sur les nœuds.

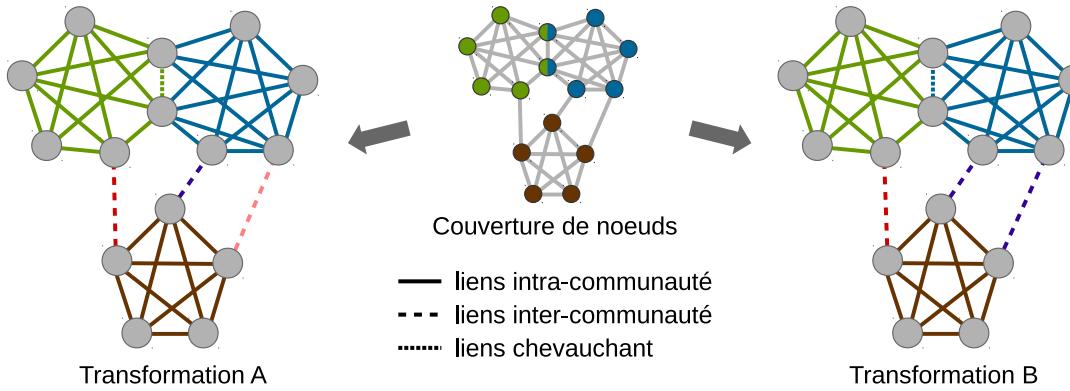


FIGURE 5.9 – Construction de *TA* et *TB* depuis une couverture de nœuds. La couleur des liens indique leur groupe.

Comme nous voulons évaluer une partition de liens, il est nécessaire de transformer cette vérité de terrain. Nous introduisons deux transformations de la couverture des nœuds en deux partitions de liens, *TA* et *TB*, voir figure 5.9.

Reprendons l'exemple d'un réseau d'interactions de personnes où chaque personne appartient à différents groupes. Afin de simplifier l'exemple, nous considérons qu'il n'existe que deux groupes : *travail* et *amis*. Le but est de déterminer le type d'une interaction à partir des groupes des personnes.

Si deux personnes partagent un seul groupe, par exemple *travail*, alors il est clair que l'interaction entre ces deux personnes devraient également être de type *travail*.

Si deux personnes ne partagent aucune communauté, l'une est dans *travail* et l'autre dans *amis*, alors plusieurs choix sont possible pour le type de l'interaction. Soit l'interaction a un type *travail-amis* car on considère que toutes les interactions reliant deux personnes des groupes *travail* et *amis* sont de même types. Soit l'interaction a son propre type car on considère qu'elle est unique.

Enfin, deux personnes peuvent partager plus qu'une communauté si les deux sont dans les communautés *travail* et *amis*. Dans ce cas, l'interaction peut être due à l'un de ces deux groupes indépendamment.

On définit maintenant de manière plus formel cette transformation qui également illustrée dans la figure 5.9. Soit $u, v \in V$, $C_{u,v}$ désigne l'intersection des communautés de u et v dans la couverture et $U_{u,v}$ désigne leur union. Nous définissons le groupe d'un lien $(u, v) \in E$ dans *TA* et *TB* de la manière suivante :

intra-communauté si $|C_{u,v}| = 1$ alors (u, v) est dans la communauté $C_{u,v}$;

inter-communauté si $|C_{u,v}| = 0$ alors dans *TA*, (u, v) appartient à sa propre communauté.

Dans *TB*, le lien appartient à la communauté $U_{u,v}$, qui contient l'ensemble des liens (u', v') tel que $U_{u',v'} = U_{u,v}$;

chevauchant si $|C_{u,v}| > 1$ alors le lien (u, v) appartient aléatoirement à une des communautés appartenant à $C_{u,v}$.

Pour générer le graphe, nous avons appliqué un jeu de paramètre classiquement utilisé dans la littérature [For10]. Ainsi, nous avons généré des graphes de 500 nœuds ayant un degré moyen de 25, un degré max de 50 et 10 nœuds appartenant à deux communautés et des communautés ayant une taille comprise entre 20 et 100. Le degré est tiré selon une loi exponentielle de paramètre -2 et la taille des communautés a -1 comme paramètre. Enfin, 90% des liens se sont à l'intérieur d'une communauté et les 10% restants sont répartis de manière aléatoire. Avec ces paramètres, il y a en moyenne 5620 liens intra-communautés, 625 liens inter-communauté et seulement 5 liens chevauchant.

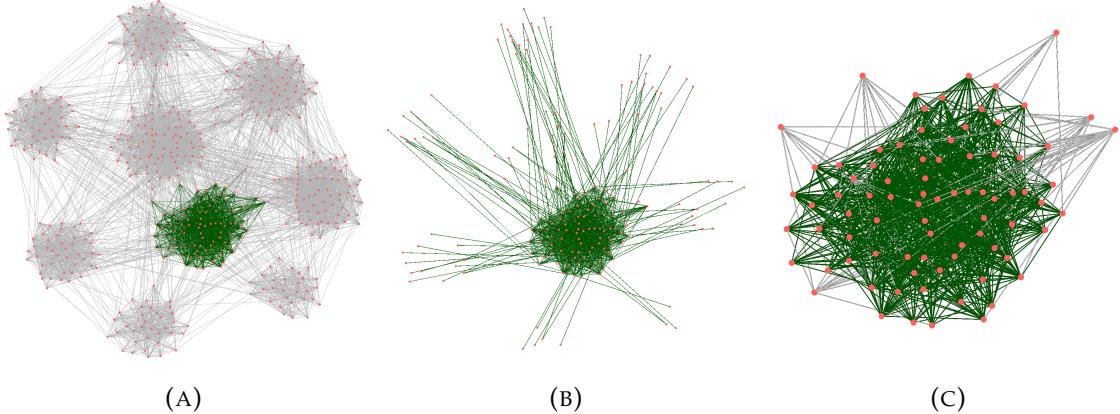


FIGURE 5.10 – Exemple de graphe généré par le LFR en (A) avec une communauté de la vérité de terrain mise en avant en vert. En (B) zoom sur une groupe détecté par $E2$ dont les liens sont en verts. En (C) zoom sur une groupe détecté par LC dont les liens sont en verts.

Pour chaque graphe généré, nous testons les partitions TA et TB mais aussi la partition LC trouvée par *link clustering* [ABL10] et les partitions EX trouvées par les méthodes de Evans *et al.* [EL09]. Ces algorithmes optimisent respectivement *Partition Density*, *Evans1*, *Evans2* et *Evans3*. Ces partitions sont ensuite évaluées par les fonctions de qualité *Partition Density*, *Evans1*, *Evans2*, *Evans3* et *Expected Node*. Une illustration d'un graphe généré et des exemples de groupes capturés par LC et $E2$ sont présentés dans la figure 5.10.

Dans LC , TA , TB et EX , il y a 720, 650, 70 et 11 groupes en moyenne. Afin d'observer la ressemblance de ces partitions, nous avons utilisé la NMI [DDGDA05]. Il apparaît que les partitions TA et TB sont les plus proches. Nous remarquons les trois fonctions de qualité $EvansX$ et les partitions respectives EX sont similaires. Ensuite, nous remarquons que la partition EX diffère de TA et de TB uniquement sur les liens inter-communautés. En effet si ils ne sont pas pris en compte lors de la comparaison, alors EX , TA et TB sont équivalentes. Il semble en effet que les liens inter-communautés soient arbitrairement distribués entre les plus grosses communautés adjacentes, ce qui est visible dans la figure 5.10b. Enfin, la partition LC , bien que proche de TA et TB , est un peu plus différente. Ces 720 groupes semblent plus petits mais aussi plus denses que ceux de TA ou TB . En particulier, les liens intra-communautés peuvent être séparés en plusieurs groupes, comme dans la figure 5.10c. Les quatre partitions sont donc différentes et mettent en avant différentes caractéristiques.

Nous procédons maintenant à l'évaluation de ces partitions par les différentes fonctions de qualités. Comme le processus de génération de graphe est aléatoire, les évaluations présentées dans la figure 5.11 représentent 30 générations. On remarque tout d'abord que ni TA ni TB n'a la meilleure évaluation selon $EvansX$ (figures 5.11c à 5.11e) ou *Partition Density* (figure 5.11a) même si TA et TB représentent nos vérités de terrain. Dans le cas de $EvansX$, c'est les partitions EX qui obtiennent les meilleures évaluations. Cela prouve l'efficacité de l'algorithme pour optimiser $EvansX$ mais remet en cause la pertinence des critères EX pour mettre en avant la vérité de terrain. Notre fonction *Expected Nodes* se comporte différemment des 4 autres. Tout d'abord, c'est la vérité de terrain TA qui obtient la meilleure évaluation puis il s'agit de TB ou LC selon les générations. Notre mesure semble donc bien mettre en avant la vérité de terrain générée. De plus, *Expected Nodes* évalue différemment les partitions TA et TB contrairement à *Partition Density* et $EvansX$. C'est un point important car, dans la partition TB , les liens inter-communautés peuvent donner lieu à des groupes non-connexes dans TB , voir figure 5.9. Ce phénomène est très pénalisé par *Expected Nodes*. Enfin selon *Expected Nodes*, les partitions EX ont une mauvaise partition et cela est également dû aux liens inter-communautés. En effet

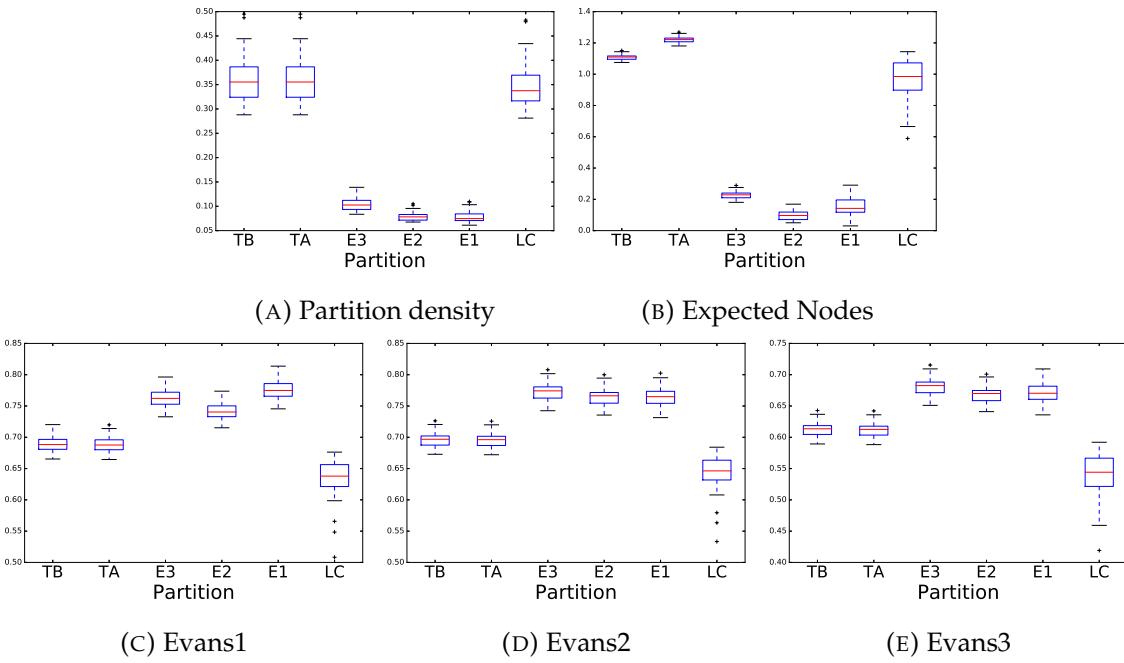


FIGURE 5.11 – Boîte à moustache des évaluations des trois fonctions de qualité pour les différentes partitions de liens. La boîte représente le premier et troisième quartile ainsi que la médiane. Les moustaches s'étendent sur 1.5 fois l'écart interquartile. Les croix sont les points au delà des moustaches.

en les fusionnant à une communauté adjacente, cela augmente fortement le nombre de nœuds interne ce qui fait baisser Q_{in} .

Avec ce test, nous mettons en évidence les limites des fonctions de qualités existantes. Les fonctions de qualité $EvansX$ ne peuvent pas permettre de capturer les vérités de terrain générée par LFR. En effet, l'algorithme de Louvain est capable de trouver des partition ayant une évaluation plus élevé que la vérité de terrain. La *Partition Density*, quant à elle, souffre surtout de son incapacité à différencier clairement des partitions différentes. Cela se matérialise par des évaluations très similaires pour les vérités de terrain et la partition *LC*. Il en résulte que la *Partition Density* est sensible aux faibles perturbations. Lors de ces tests, *Expected Nodes* ne semble pas souffrir de ces défauts. Pour ces raisons, nous pensons que *Expected Nodes* est une mesure qui permet de bien évaluer les partitions de liens.

5.4 Calcul et optimisation

Jusqu'à maintenant, nous avons évalué *Expected Nodes* sans nous attacher ni à son calcul ni à son optimisation. Nous discutons maintenant de la complexité de calcul d'*Expected Nodes* pour une partition donnée. Le calcul de la qualité interne 5.5 nécessite d'évaluer la probabilité qu'un nœud soit tiré. Ce calcul de probabilité nécessite d'évaluer pour un nœud u : $1 - \frac{\binom{2|E|-d(u)}{2|L|}}{\binom{2|E|}{2|L|}}$, ce qui peut être assez couteux. Ce calcul correspond à une loi hypergéométrique. Or sous certaines conditions, une loi hypergéométrique peut être approché par une loi binomiale ce qui simplifie le calcul à : $1 - (1 - \frac{|L|}{|E|})^{|L|}$ ce qui est plus rapide. Lors de nos tests, le biais induit par cette approximation reste assez faible, de l'ordre de 0.01% en moyenne. Ce changement de calcul est équivalent à considérer un tirage avec remise au lieu de tirage sans remise. De plus cette probabilité ne dépend que du degré du nœud et du nombre de liens dans le groupe.

Donc, tout les nœuds ayant le même degré donnent lieu à la même probabilité. Si l'on considère que l'évaluation de la probabilité pour un nœud peut se faire en $O(1)$ alors, pour un groupe donné, il est possible de calculer sa qualité en $O(|\{d_G(v)\}_{v \in V}|)$. Cette formulation est très efficace lorsque beaucoup de nœuds ont le même degré. Enfin comme la qualité d'un groupe ne dépend que de sa taille, on peut calculer la qualité d'une partition \mathcal{L} en $O(\{|L_i|\}_{L_i \in \mathcal{L}})$.

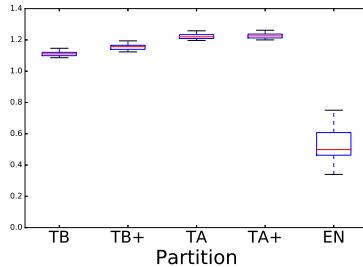
Il est donc assez rapide de calculer Q_{in} . En revanche, la situation est complètement différente pour Q_{ext} . Le processus de calcul est similaire. On peut également appliquer l'approximation de la loi hypergéométrique par une loi binomiale mais deux nœuds de même degré ne vont plus forcément avoir la même probabilité d'être tirés. En effet, Q_{ext} n'utilise pas le graphe initial mais le graphe $G \setminus L_i$. Pour chaque nœud, il faut évaluer son degré dans ce nouveau graphe. Le calcul de Q_{ext} pour un groupe se fait donc en $O(n)$. Pour l'évaluation d'une partition, il n'est pas non plus possible de considérer comme équivalents des groupes de même taille. Le coût pour évaluer une partition est donc en $O(|\mathcal{L}|n)$. Le code pour évaluer une partition de liens d'un graphe avec *Expected Nodes* mais aussi avec *Partition Density*, *Evans1*, *Evans2* et *Evans3* est disponible en ligne : <https://github.com/ksadorf/ExpectedNodes>.

Malgré ce coût élevé, nous avons développé un premier algorithme d'optimisation glouton de *Expected Node*. Le principe de fonctionnement est le suivant. Chaque lien est initialement dans son propre groupe. Puis à chaque itération, on considère deux types de modification de la solution courante. Soit la meilleure fusion de deux communautés soit le meilleur changement de communauté d'un seul lien. On fusionne ou change de communauté un lien si cela améliore la qualité de la partition. Les fusions considérées sont les fusions entre des communautés adjacentes. Les changements de liens se font également que avec une communauté adjacente. On continue de modifier la solution courante tant qu'elle est améliorable par un de ces mouvements. Malheureusement cette approche souffre de deux problèmes majeurs. Tout d'abord le calcul du gain est couteux et rend impossible l'étude de grands jeux de données. Ensuite dans nos tests dans les graphes générés par LFR, il semble que cette méthode reste bloquée sur des optimum locaux bien plus faibles que la vérité de terrain, voir la figure 5.12a où la qualité de la partition trouvée par notre algorithme, noté *EN*, est présentée. Il faudrait donc tester d'autres heuristiques d'optimisation mais aussi travailler sur une méthode de calcul du gain plus optimisée.

Malgré ces limitations, nous avons tout de même tiré parti de cet algorithme naïf afin de tester si une partition donnée peut être améliorée. En effet même si l'algorithme n'est pas adapté pour trouver une partition ayant une qualité élevée en partant de zéro, il peut modifier une partition donnée pour l'améliorer. Nous avons donc utilisé les partitions *TA* et *TB* comme partition de départ de l'algorithme et nous avons observé si l'algorithme est capable d'améliorer les vérités de terrain. Les changements qu'apportent notre algorithme se portent principalement sur les liens inter-communautés. Dans le cas de *TA*, chaque lien inter-communauté constitue une communauté. Or, il se peut que plusieurs liens inter-communautés soient connectés aux mêmes nœuds. Dans ce genre de situation, notre algorithme fusionne ces liens dans une communauté augmentant légèrement la qualité globale. Après optimisation, nous constatons que les partitions *TA* et *TB* peuvent être légèrement améliorées mais qu'elles sont très proches du maximum local lorsque l'on considère notre algorithme d'optimisation, voir la figure 5.12a où les partitions *TA+* et *TB+* sont les versions améliorées de *TA* et *TB*.

5.5 Conclusion

Nous considérons de nouveaux critères pour l'évaluation des partitions de liens tenant compte de la répartition des nœuds internes et externes d'un groupe. À partir de ces critères, nous définissons une mesure de qualité, *Expected Nodes*, basée sur la différence entre le nombre de nœuds induits par un groupe de lien et le nombre de nœuds attendu dans un modèle nul.



(A) Expected nodes

FIGURE 5.12 – Boite à moustache des évaluations selon Expected Nodes pour différentes partitions. TA et TB sont les vérités de terrains, EN est la partition trouvée par notre algorithme et enfin $TA+$ et $TB+$ sont les versions améliorées par notre algorithme de TA et TB .

Ce processus est similaire à celui qui a donné lieu à la modularité. En plus de s'inspirer de la modularité, *Expected Nodes* ne prends pas uniquement en compte la répartition de ses nœuds induits mais également celle de nœuds et liens adjacents. Ce changement permet d'avoir une évaluation plus fine d'une communauté. Pour montrer la pertinence de cette nouvelle fonction de qualité, nous évaluons quatre fonctions de qualité de la littérature. Nous montrons que, sur nos jeux de tests, *Expected Nodes* semble être la plus à même de capturer la vérité de terrain et de différencier des partitions différentes. Malgré le coût de calcul élevé d'*Expected Nodes*, nous avons implémenté un premier algorithme d'optimisation agglomératif. Cet algorithme n'est, pour l'instant, pas capable d'obtenir des partitions avec des scores élevés. Cependant il nous a permis de vérifier que les vérités de terrain générée par LFR sont des quasi-optimum locaux selon *Expected Nodes*. Bien qu'il soit possible d'améliorer légèrement les vérités de terrains, la structure de l'optimum local est très proche de la partition initiale.

5.5.1 Perspective

Les perspectives autour d'*Expected Nodes* sont nombreuses et portent sur deux points : d'une part le calcul et l'optimisation d'*Expected Nodes* et d'autre part les cas d'applications possibles.

Amélioration de l'optimisation d'*Expected Nodes*

Considérer plus de modifications Il semble que les mouvements envisagés pour modifier une solution courante ne soient pas suffisant pour échapper à des maximums locaux qui sont bien inférieurs à la qualité des vérités de terrain. Il faudrait donc envisager de nouveaux. Inclure la possibilité de diviser une communauté en deux ne semble pas envisageable car il existe de trop nombreuses coupes possibles et le coût de calcul associé est trop coûteux. Il semble, par contre, prometteur de considérer des mouvements associés à un nœud. En effet au lieu de changer un à un les liens d'une communauté, il devrait être possible d'intégrer en un seul mouvement tous les liens adjacents qui sont reliés à un nœud adjacent donné. Pour l'illustrer ce mouvement, reprenons l'exemple d'une clique incluse dans une autre clique plus grande. La solution finale devrait être la clique maximale. Si l'on ne considère que l'ajout de lien un à un³, alors il n'est pas possible d'atteindre la clique maximale. L'algorithme est bloqué car l'ajout d'un unique lien détériore la solution courante. Si on considère l'ajout de tous les liens adjacents à un nœud adjacent donné, alors, après l'ajout, le groupe considéré est toujours une clique et la qualité interne ne diminue pas mais en plus on peut espérer faire diminuer la pénalisation de la qualité externe. De cette manière, il devrait être possible d'éviter des maximums locaux.

3. Dans le cas où la fusion de communauté n'est pas une option viable.

Plus globalement, les mouvement faisant intervenir plusieurs liens d'un seul coup semblent intéressants.

Diminuer le coût de calcul du gain Ici, il ne s'agit pas d'améliorer la performance de l'algorithme en terme de qualité obtenue par la partition trouvée par l'algorithme mais de le rendre plus rapide. Dans sa version courante, l'algorithme est trop couteux pour traiter des graphes ayant plus de quelques dizaines de milliers de liens. La force de l'algorithme de Louvain réside dans l'existence d'une expression analytique du gain de fusion de deux communautés qui est rapide à évaluer. Dans notre algorithme, la lenteur est justement due au calcul du gain qui est très coûteux. En effet, nous ne calculons pas directement le gain d'un mouvement mais la différence entre la qualité courante et la qualité résultant du mouvement. C'est une différence importante car, dans notre cas, il faut calculer de zéro la qualité des communautés impactées. Une manière de résoudre ce problème serait de ne pas partir de zéro mais de calculer localement les changements ayant un impact. Prenons le cas de la fusion de communautés A et B . Cela est simple pour la qualité interne car elle ne dépend que du nombre de nœuds et liens internes. Le nouveau nombre de liens, après fusion, est simplement la somme des liens des communautés initiales. Le nouveau nombre de nœuds internes est la taille de l'union des nœuds internes. La situation est plus complexe pour la qualité externe. Le nouveau nombre de liens adjacents n'est pas simplement une somme. De même, l'identité et le degré des nouveaux nœuds adjacents dans $G \setminus (A \cup B)$ ne sont pas triviales à calculer. Il faudrait des parcours locaux sur $(A \cup B)$ et son voisinage pour arriver à mettre à jour ces données nécessaires au calcul de Q_{ext} . Si ce parcours est fait suffisamment soigneusement, il est sûrement possible de gagner en terme de vitesse d'exécution.

Changer la stratégie d'optimisation Une autre source potentielle de gain en vitesse de l'algorithme est la stratégie d'optimisation. Lors d'une itération, nous appliquons la meilleure fusion de communauté ou le meilleur changement de communauté d'un lien. Appliquer le meilleur mouvement peut améliorer la convergence de l'algorithme vers l'optimum local mais cela est coûteux car lors d'une itération tous les gains doivent être calculés. C'est problématique car c'est justement le calcul du gain qui est très coûteux. Une solution face à ce problème est d'utiliser une variante stochastique de l'algorithme. Dans cette variante, ce n'est pas le meilleur changement qui est appliqué mais le premier qui mène à une amélioration de la qualité. Pour ce faire, il est nécessaire de considérer un ordre aléatoire des mouvements possibles et d'appliquer le premier dont le gain est positif puis de recommencer. Avec ce changement, on teste beaucoup moins de mouvements avant d'en appliquer un. Cependant, il est possible que plus de mouvements soient nécessaires avant d'arriver à un optimum local. Par ailleurs avec ce changement, l'algorithme n'est plus déterministe car il dépend de l'ordre d'évaluation des mouvements. Il faudrait donc être très prudent vis-à-vis de ses aspects, lors du test de cette technique.

Cas d'utilisation d'*Expected Nodes*

Nous avons parlé de moyens possibles pour trouver une méthode d'optimisation efficace d'*Expected Nodes*. Il est également nécessaire de revenir sur l'utilisation d'*Expected Nodes*. Nous avons testé *Expected Nodes* sur un algorithme largement utilisé dans la littérature et les résultats sont encourageants. Il y a cependant une différence majeure dans la manière dont nous avons testé *Expected Nodes* vis-à-vis des autres fonctions de qualités. Il n'existe pas, pour l'instant, d'algorithme efficace pour trouver une partition maximisant *Expected Nodes*. C'est pourquoi nous testons des partitions trouvées empiriquement par d'autres moyens. Cependant, il existe peut-être une partition que nous n'avons pas testé qui aurait une bien meilleure qualité que les vérités de terrains. L'amélioration de l'algorithme d'optimisation et le test d'autres partitions accroiraient la pertinence des partitions testées et ainsi le bien fondé d'*Expected Nodes*. De manière

analogique, il serait intéressant d'évaluer *Expected Nodes* sur des jeux de données ayant une réelle vérité de terrain sur les liens, que ce soit des données réelles ou simulées. Plus globalement, il serait intéressant de comprendre ce qu'il est possible de capturer et représenter comme structure avec une partition de liens.

Nous avons parlé jusque maintenant de l'application d'*Expected Nodes* sur des graphes non pondérés et non dirigés. La question de la pondération est délicate à prendre en compte pour *Expected Nodes*. En effet, notre méthode repose sur des tirages aléatoires de demi-liens. La notion de tirage est intrinsèquement lié à des valeurs entières. En effet, il n'existe pas à ma connaissance de pendant continu à la loi hypergéométrique ; on ne tire pas aléatoirement 0.6 demi-liens parmi 14.3 demi liens possible. L'application d'*Expected Nodes* à un graphe pondéré semble donc compromise. Cependant, il devrait être trivialement possible de considérer des multigraphes. Dans les multigraphes, deux noeuds peuvent être relié par plus qu'un lien. Les notions de tirages aléatoires peuvent s'adapter à ce genre graphe assez facilement. Il y a toute fois une différence majeur lorsque l'on considère les multigraphes par rapport aux graphes classiques : les contraintes sur la partition résultat. Dans le cas des multigraphes, il serait sûrement intéressant d'ajouter la contrainte que chaque communauté ne doit contenir qu'une seule occurrence d'un lien donné. Ainsi s'il existe deux liens entre u et v , alors ces deux liens devront être dans deux communautés différentes.

Chapitre 6

Fonction de qualité

Sommaire

6.1 Définition	57
6.2 Générateur de flots de liens avec structure communautaire	57

Génération de : [GDA⁺15, KPV14, PGPSV12] snapshot generateur [SBPS13, VGB14]

6.1 Définition

6.2 Générateur de flots de liens avec structure communautaire

Chapitre 7

Conclusion

Bibliographie

- [ABFX08] Edoardo Airoldi, David Blei, Stephen Fienberg, and Eric Xing. Mixed Membership Stochastic Blockmodels. *J. Mach. Learn. Res.*, 9 :1981 – 2014, 2008.
- [ABL10] Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307) :761–764, aug 2010.
- [AG10] Thomas Aynaud and Jean-Loup Guillaume. Static community detection algorithms for evolving networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 508–514, 2010.
- [AGL14] Alice Albano, Jean Loup Guillaume, and Benedicte Le Grand. On the use of intrinsic time scale for dynamic community detection and visualization in social networks. In *Proceedings - International Conference on Research Challenges in Information Science*, 2014.
- [AM11] Rodrigo Aldecoa and Ignacio Marín. Deciphering Network Community Structure by Surprise. *PLoS ONE*, 6(9) :e24195, sep 2011.
- [aMGH11] a.F. McDaid, Derek Greene, and Neil Hurley. Normalized Mutual Information to evaluate overlapping community finding algorithms. *Arxiv preprint arXiv:1110.2515*, pages 1–3, 2011.
- [APU09] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs, 2009.
- [BBC⁺14] S. Boccaletti, G. Bianconi, R. Criado, C.I. del Genio, J. Gómez-Gardeñes, M. Romance, I. Sendiña-Nadal, Z. Wang, and M. Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 544(1) :1–122, 2014.
- [BBC⁺15] Oana Denisa Balalau, Francesco Bonchi, T-H. Hubert Chan, Francesco Gullo, and Mauro Sozio. Finding Subgraphs with Maximum Total Density and Limited Overlap. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining - WSDM '15*, pages 379–388, New York, New York, USA, feb 2015. ACM Press.
- [BC78] Edward A Bender and E.Rodney Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3) :296–307, may 1978.
- [BCS15] Sanghamitra Bandyopadhyay, Garisha Chowdhary, and Debarka Sengupta. FOCS : Fast Overlapped Community Search. *IEEE Transactions on Knowledge and Data Engineering*, PP(99) :1–1, 2015.
- [BDG⁺07] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On Finding Graph Clusterings with Maximum Modularity. In *Graph-Theoretic Concepts in Computer Science*, volume 4769, pages 121–132. 2007.
- [BGLL08] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics : Theory and Experiment*, 2008(10) :P10008, oct 2008.

- [BKN11] Brian Ball, Brian Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3) :036103, sep 2011.
- [BP15] Vladimir Batagelj and Selena Praprotnik. An algebraic approach to temporal network analysis based on temporal quantities. may 2015.
- [BPW⁺13] Danielle S. Bassett, Mason A. Porter, Nicholas F. Wymbs, Scott T. Grafton, Jean M. Carlson, and Peter J. Mucha. Robust Detection of Dynamic Community Structure in Networks. *Chaos : An Interdisciplinary Journal of Nonlinear Science*, 23(1) :013142, 2013.
- [BPW⁺16] Marya Bazzi, Mason a. Porter, Stacy Williams, Mark McDonald, Daniel J. Fenn, and Sam D. Howison. Community detection in temporal multilayer networks, and its application to correlation networks. *Multiscale Modeling & Simulation*, 14(1) :1–41, 2016.
- [Bre74] Breiger, Ronald. The Duality of Persons and Groups. *Social Forces*, 53(2) :181–190, 1974.
- [CA14] Rémy Cazabet and Frédéric Amblard. Dynamic Community Detection. In *Encyclopedia of Social Network Analysis and Mining*, pages 404–414. Springer New York, New York, NY, 2014.
- [CAH10] R. Cazabet, F. Amblard, and C. Hanachi. Detection of Overlapping Communities in Dynamical Social Networks. *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, 2010.
- [CBW13] RajmondaSulo Caceres and Tanya Berger-Wolf. Temporal Scale of Dynamic Networks. In Petter Holme and Jari Saramäki, editors, *Temporal Networks, Understanding Complex Systems*, pages 65–94. Springer Berlin Heidelberg, 2013.
- [CFQS11] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6811 LNCS, pages 346–359, 2011.
- [CGP11] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*, 4(5) :512–546, oct 2011.
- [CKT06] Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*, page 554, New York, New York, USA, aug 2006. ACM Press.
- [CKU13] Yudong Chen, Vikas Kawadia, and Rahul Urgaonkar. Detecting Overlapping Temporal Community Structure in Time-Evolving Networks. *arXiv preprint arXiv :1303.7226*, page 12, mar 2013.
- [CLR16] Marco Corneli, Pierre Latouche, and Fabrice Rossi. Exact ICL maximization in a non-stationary temporal extension of the stochastic block model for dynamic networks. *Neurocomputing*, mar 2016.
- [CSG16] Mário Cordeiro, Rui Portocarrero Sarmento, and João Gama. Dynamic community detection in evolving networks using locality modularity optimization. *Social Network Analysis and Mining*, 6(1) :15, mar 2016.
- [CVW⁺15] Eduardo Chinelate Costa, Alex Borges Vieira, Klaus Wehmuth, Artur Ziviani, and Ana Paula Couto da Silva. Time Centrality in Dynamic Complex Networks. *arXiv preprint*, apr 2015.

- [DDG⁺15] Armando Di Nardo, Michele Di Natale, Carlo Giudicianni, Dino Musmarra, Giovanni Francesco Santonastaso, and Antonietta Simone. Water Distribution System Clustering and Partitioning Based on Social Network Algorithms. *Procedia Engineering*, 119 :196–205, 2015.
- [DDGDA05] Leon Danon, Albert Díaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics : Theory and Experiment*, 2005(09) :P09008–P09008, sep 2005.
- [DGG12] Maximilien Danisch, Jean-Loup Guillaume, and Bénédicte Le Grand. Towards multi-ego-centred communities : a node similarity approach. *International Journal of Web Based Communities*, mar 2012.
- [DLAR14] Manlio De Domenico, Andrea Lancichinetti, Alex Arenas, and Martin Rosvall. Identifying modular flows on multilayer networks reveals highly overlapping organization in social systems. *arXiv preprint*, aug 2014.
- [DLCA07] Remi Dorat, Matthieu Latapy, Bernard Conein, and Nicolas Auray. Multi-level analysis of an interaction network between individuals in a mailing-list. *Ann Telecommun*, 62 :325–349, 2007.
- [DM04] Luca Donetti and Miguel A. Muñoz. Detecting network communities : a new systematic and efficient algorithm. *Journal of Statistical Mechanics : Theory and Experiment*, 10(2004) :8, 2004.
- [dRSKvdH14] Marcel A de Reus, Victor M Saenger, René S Kahn, and Martijn P van den Heuvel. An edge-centric perspective on the human connectome : link communities in the brain. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 369(1653) :20130527, oct 2014.
- [DSRC⁺13] Manlio De Domenico, Albert Solé-Ribalta, Emanuele Cozzo, Mikko Kivelä, Yamir Moreno, Mason A. Porter, Sergio Gómez, and Alex Arenas. Mathematical Formulation of Multilayer Networks. *Physical Review X*, 3(4) :041022, dec 2013.
- [DVR16] Simon De Ridder, Benjamin Vandermarkiere, and Jan Ryckebusch. Detection and localization of change points in temporal networks with the aid of stochastic block models. *arXiv preprint arXiv :1602.00661*, 2016.
- [DYB10] J-C Delvenne, S N Yaliraki, and M Barahona. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences of the United States of America*, 107(29) :12755–60, jul 2010.
- [EL09] T. S. Evans and Renaud Lambiotte. Line graphs, link partitions, and overlapping communities. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 80(1) :016105, jul 2009.
- [ER59] P Erdős and a Rényi. On random graphs. *Publicationes Mathematicae*, 6 :290–297, 1959.
- [ER11] Alcides Viamontes Esquivel and Martin Rosvall. Compression of Flow Can Reveal Overlapping-Module Organization in Networks. *Physical Review X*, 1 :1–11, 2011.
- [FB07] Santo Fortunato and Marc Barthélémy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences of the United States of America*, 104(1) :36–41, 2007.
- [FCF11] Adrien Friggeri, Guillaume Chelius, and Eric Fleury. Triangles to capture social cohesion. In *Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011*, pages 257–265, 2011.

- [For10] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3–5) :75–174, feb 2010.
- [FW15] Damien R Farine and Hal Whitehead. Constructing, conducting, and interpreting animal social network analysis. *The Journal of animal ecology*, jul 2015.
- [GB13] Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences of the United States of America*, 110(36) :14534–9, sep 2013.
- [GDA⁺15] Clara Granell, Richard K. Darst, Alex Arenas, Santo Fortunato, and Sergio Gómez. A benchmark model to assess community structure in evolving networks. page 11, jan 2015.
- [GdMC10] Benjamin H. Good, Yves-Alexandre de Montjoye, and Aaron Clauset. Performance of modularity maximization in practical contexts. *Physical Review E*, 81(4) :046106, apr 2010.
- [GPBC15] Laetitia Gauvin, André Panisson, Alain Barrat, and Ciro Cattuto. Revealing latent factors of temporal networks for mesoscale intervention in epidemic spread. jan 2015.
- [GPC14] Laetitia Gauvin, André Panisson, and Ciro Cattuto. Detecting the community structure and activity patterns of temporal networks : a non-negative tensor factorization approach. *PloS one*, 9(1) :e86028, jan 2014.
- [Gre10] Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12, 2010.
- [GVFS⁺16] Noé Gaumont, Tiphaine Viard, Raphaël Fournier-S’niehotta, Qinna Wang, and Matthieu Latapy. Analysis of the temporal and structural features of threads in a mailing-list. In *Complex Networks VII, Studies in Computational Intelligence*. 2016.
- [HA85] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1) :193–218, 1985.
- [HBG14] S Harenberg, G Bello, and L Gjeltema. Community detection in large-scale networks : a survey and empirical evaluation. *Wiley*, 2014.
- [HBP15] De-Shuang Huang, Vitoantonio Bevilacqua, and Prashan Premaratne, editors. *Intelligent Computing Theories and Methodologies*, volume 9225 of *Lecture Notes in Computer Science*. Springer International Publishing, Cham, 2015.
- [HDF14] Darko Hric, Richard K. Darst, and Santo Fortunato. Community detection in networks : Structural communities versus ground truth. *Physical Review E*, 90(6) :062805, 2014.
- [HK14] Dávid X Horváth and János Kertész. Spreading dynamics on networks : the role of burstiness, topology and non-stationarity. *New Journal of Physics*, 16(7) :073037, jul 2014.
- [HKKS04] John Hopcroft, Omar Khan, Brian Kulis, and Bart Selman. Tracking evolving communities in large linked networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101 Suppl :5249–5253, apr 2004.
- [HKW14] Tanja Hartmann, Andrea Kappes, and Dorothea Wagner. Clustering evolving networks. *arXiv preprint arXiv:1401.3516*, 2014.
- [HL14] Petter Holme and Fredrik Liljeros. Birth and death of links control disease spreading in empirical contact networks. *Scientific reports*, 4 :4999, jan 2014.
- [HLL83] Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels : First steps. *Social Networks*, 5(2) :109–137, 1983.

- [Hol15a] Petter Holme. Information content of contact-pattern representations and predictability of epidemic outbreaks. mar 2015.
- [Hol15b] Petter Holme. Modern temporal network theory : a colloquium. *Eur. Phys. J. B*, 88(9) :234, 2015.
- [HS13] Petter Holme and Jari Saramäki, editors. *Temporal Networks. Understanding Complex Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [HWW⁺13] Lan Huang, Guishen Wang, Yan Wang, Enrico Blanzieri, and Chao Su. Link Clustering with Extended Link Similarity and EQ Evaluation Division. *PLoS ONE*, 8(6) :e66005, jun 2013.
- [JB^P+15] Lucas G. S. Jeub, Prakash Balachandran, Mason A. Porter, Peter J. Mucha, and Michael W. Mahoney. Think locally, act locally : Detection of small, medium-sized, and large communities in large networks. *Physical Review E*, 91(1) :012821, jan 2015.
- [JPKK14] Hang-Hyun Jo, Juan I. Perotti, Kimmo Kaski, and János Kertész. Analytically Solvable Model of Spreading Dynamics with Non-Poissonian Processes. *Physical Review X*, 4(1) :011041, mar 2014.
- [KA12] Hyoungshick Kim and Ross Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2) :026107, feb 2012.
- [KAB⁺14] Mikko Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3) :203–271, jul 2014.
- [Kan14] Rushed Kanawati. Seed-centric approaches for community detection in complex networks. In Gabriele Meiselwitz, editor, *6th international conference on Social Computing and Social Media*, volume LNCS 8531, pages 197–208. Springer International Publishing, 2014.
- [KBV15] J. Kalavathi, S. Balamurali, and M. Venkatesulu. An Efficient Evolutionary Approach for Identifying Evolving Groups in Dynamic Social Networks Using Genetic Modeling. *Procedia Computer Science*, 57 :428–437, 2015.
- [Kim14] Sungmin Kim. Community Detection in Directed Networks and its Application to Analysis of Social Networks. 2014.
- [KJ11] Youngdo Kim and Hawoong Jeong. Map equation for link communities. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 84(2) :026110, aug 2011.
- [KKB⁺12] Gautier Krings, Márton Karsai, Sebastian Bernhardsson, Vincent D Blondel, and Jari Saramäki. Effects of time window size and placement on the structure of an aggregated communication network. *EPJ Data Science*, 1(1) :4, may 2012.
- [KKBK12] Márton Karsai, Kimmo Kaski, Albert-László Barabási, and János Kertész. Universal features of correlated bursty behaviour. *Scientific reports*, 2 :397, jan 2012.
- [KKK⁺11] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics : Theory and Experiment*, 2011(11) :P11005, nov 2011.
- [KKKS08] Jussi M Kumpula, Mikko Kivelä, Kimmo Kaski, and Jari Saramäki. Sequential algorithm for fast clique percolation. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 78 :026109, 2008.
- [KKKS13] Lauri Kovanen, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs reveal homophily, gender-specific patterns and group talk in mobile communication networks. page 8, feb 2013.

- [KKP⁺11] Márton Karsai, Mikko Kivelä, R. K. Pan, K. Kaski, J. Kertész, A.-L. Barabási, and Jari Saramäki. Small but slow world : How network topology and burstiness slow down spreading. *Physical Review E*, 83(2) :025102, feb 2011.
- [KN11] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1) :016107, jan 2011.
- [KPV14] Márton Karsai, Nicola Perra, and Alessandro Vespignani. Time varying networks and the weakness of strong ties. *Scientific reports*, 4 :4001, jan 2014.
- [KR15] Tatsuro Kawamoto and Martin Rosvall. Estimating the resolution limit of the map equation in community detection. *Physical Review E*, 91(1) :012809, jan 2015.
- [Lam78] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7) :558–565, jul 1978.
- [LB62] C. Lekkeikerker and J. Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51(1) :45–64, 1962.
- [LCZ⁺08] Yu-Ru Lin, Yun Chi, Shenghuo Zhu, Hari Sundaram, and Belle L. Tseng. Facetnet : A Framework for Analyzing Communities and Their Evolutions in Dynamic Networks. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 685, New York, New York, USA, apr 2008. ACM Press.
- [LF09a] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 80 :016118, 2009.
- [LF09b] Andrea Lancichinetti and Santo Fortunato. Community detection algorithms : a comparative analysis. Technical report, aug 2009.
- [LF11] Andrea Lancichinetti and Santo Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84(6) :066122, dec 2011.
- [LHB⁺12] Jingyong Li, Lan Huang, Tian Bai, Zhe Wang, and Hongsheng Chen. CDBIA : A dynamic community detection method based on incremental analysis. In *2012 International Conference on Systems and Informatics (ICSAI2012)*, pages 2224–2228. IEEE, may 2012.
- [LLDM08] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 695, New York, New York, USA, apr 2008. ACM Press.
- [LLJT14] Shenghong Li, Hao Lou, Wen Jiang, and Junhua Tang. Detecting Community Structure via Synchronous Label Propagation. *Neurocomputing*, nov 2014.
- [LRK⁺14] Sungsu Lim, Seungwoo Ryu, Sejeong Kwon, Kyomin Jung, and Jae-Gil Lee. LinkSCAN* : Overlapping community detection using the link-space transformation. In *2014 IEEE 30th International Conference on Data Engineering*, pages 292–303. IEEE, mar 2014.
- [LRRF11] Andrea Lancichinetti, Filippo Radicchi, José J Ramasco, and Santo Fortunato. Finding statistically significant communities in networks. *PloS one*, 6(4) :e18961, jan 2011.
- [LS99] D D Lee and H S Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755) :788–91, 1999.
- [LWP08] Feng Luo, James Wang, and Eric Promislow. Exploring Local Community Structures in Large Networks. *2006 IEEE/WIC/ACM International Conference on Web Intelligence WI 2006 Main Conference ProceedingsWI06*, 6(4) :387–400, 2008.

- [LZW⁺13] Zhenping Li, Xiang-Sun Zhang, Rui-Sheng Wang, Hongwei Liu, and Shihua Zhang. Discovering link communities in complex networks by an integer programming model and a genetic algorithm. *PloS one*, 8 :e83739, 2013.
- [MM15] Catherine Matias and Vincent Miele. Statistical clustering of temporal networks through a dynamic stochastic block model. jun 2015.
- [MRM⁺10] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community Structure in Time-Dependent, Multiscale, and Multiplex Networks. *Science*, 328(5980) :876–878, may 2010.
- [MSCA09] R. D. Malmgren, D. B. Stouffer, A. S. L. O. Campanharo, and L. A. N. Amaral. On Universality in Human Correspondence Activity. *Science*, 325(5948) :1696–1700, sep 2009.
- [MSMA08] R Dean Malmgren, Daniel B Stouffer, Adilson E Motter, and Luís A N Amaral. A Poissonian explanation for heavy tails in e-mail communication. *Proceedings of the National Academy of Sciences of the United States of America*, 105(47) :18153–8, nov 2008.
- [MSPS15] Antoine Moinet, Michele Starnini, and Romualdo Pastor-Satorras. Burstiness and Aging in Social Temporal Networks. *Physical Review Letters*, 114(10) :108701, mar 2015.
- [MT09] Marija Mitrović and Bosiljka Tadić. Spectral and dynamical properties in classes of sparse networks with mesoscopic inhomogeneities. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 80(2), 2009.
- [MT15] Clémence Magnien and Fabien Tarissan. Time Evolution of the Importance of Nodes in dynamic Networks. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015 - ASONAM '15*, pages 1200–1207, New York, New York, USA, 2015. ACM Press.
- [MV13] FD Malliaros and M Vazirgiannis. Clustering and community detection in directed networks : A survey. *Physics Reports*, 2013.
- [New09] M. E. J. Newman. Random Graphs with Clustering. *Physical Review Letters*, 103(5) :058701, jul 2009.
- [New16] MEJ Newman. Community detection in networks : Modularity optimization and maximum likelihood are equivalent. *arXiv preprint arXiv* :1606.02319, 2016.
- [NG04] M. E J Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 69, 2004.
- [NMCM09] V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics : Theory and Experiment*, 2009(03) :P03024, mar 2009.
- [NS01] Krzysztof Nowicki and Tom a. B Snijders. Estimation and Prediction for Stochastic Blockstructures. *Journal of the American Statistical Association*, 96(455) :1077–1087, 2001.
- [PB15] Selena Praprotnik and Vladimir Batagelj. Spectral centrality measures in temporal networks, jul 2015.
- [PBV07] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136) :664–667, 2007.
- [PC13] M Plantié and M Crampes. Survey on social community detection. *Social Media Retrieval*, 2013.

- [PC15] Leto Peel and Aaron Clauset. Detecting change points in the large-scale structure of evolving networks. In *29th AAAI Conference on Artificial Intelligence (AAAI)*, pages 1–11, 2015.
- [PDFV05] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043) :814–818, jun 2005.
- [Pei15] Tiago P. Peixoto. Inferring the mesoscale structure of layered, edge-valued, and time-varying networks. *Physical Review E*, 92(4) :042807, oct 2015.
- [PGPSV12] Nicola Perra, B Gonçalves, R Pastor-Satorras, and A Vespignani. Activity driven modeling of time varying networks. *Scientific reports*, 2 :469, jan 2012.
- [PHK11] Daniel Cosmin Porumbel, Jin Kao Hao, and Pascale Kuntz. An efficient algorithm for computing the distance between close partitions. *Discrete Applied Mathematics*, 159(1) :53–59, jan 2011.
- [PJHS14] Juan Ignacio Perotti, Hang-Hyun Jo, Petter Holme, and Jari Saramäki. Temporal network sparsity and the slowing down of spreading. nov 2014.
- [PL05] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *Computer and Information Sciences (ISCIS)*, 10 :284–293, 2005.
- [PSG⁺13] René Pfitzner, Ingo Scholtes, Antonios Garas, Claudio J. Tessone, and Frank Schweitzer. Betweenness Preference : Quantifying Correlations in the Topological Dynamics of Temporal Networks. *Physical Review Letters*, 110(19) :198701, may 2013.
- [RAK07] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 76, 2007.
- [Ran71] William M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336) :846–850, 1971.
- [RB06] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 74(1), 2006.
- [RB08] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*, 105(4) :1118–23, jan 2008.
- [RB10] Martin Rosvall and Carl T Bergstrom. Mapping change in large networks. *PLoS one*, 5(1) :e8694, jan 2010.
- [RB13] Luis Enrique Correa Rocha and Vincent D Blondel. Bursts of vertex activation and epidemics in evolving networks. *PLoS computational biology*, 9(3) :e1002974, jan 2013.
- [RPB13] Bruno Ribeiro, Nicola Perra, and Andrea Baronchelli. Quantifying the effect of temporal resolution on time-varying networks. *Scientific reports*, 3 :3006, jan 2013.
- [SBB10] Juliette Stehlé, Alain Barrat, and Ginestra Bianconi. Dynamical and bursty interactions in social networks. *Physical Review E*, 81(3) :035101, mar 2010.
- [SBBPS12] Michele Starnini, Andrea Baronchelli, Alain Barrat, and Romualdo Pastor-Satorras. Random walks on temporal networks. *Physical Review E*, 85(5) :056115, may 2012.

- [SBPS13] Michele Starnini, Andrea Baronchelli, and Romualdo Pastor-Satorras. Modeling Human Dynamics of Face-to-Face Interaction Networks. *Physical Review Letters*, 110(16) :168701, apr 2013.
- [SCCH09] Huawei Shen, Xueqi Cheng, Kai Cai, and Mao-Bin Hu. Detect overlapping and hierarchical community structure in networks. *Physica A : Statistical Mechanics and its Applications*, 388(8) :1706–1712, apr 2009.
- [SCF⁺13] Chuan Shi, Yanan Cai, Di Fu, Yuxiao Dong, and Bin Wu. A link clustering based overlapping community detection algorithm. In *Data and Knowledge Engineering*, volume 87, pages 394–404, 2013.
- [SFPY07] Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S. Yu. GraphScope. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*, page 687, New York, New York, USA, aug 2007. ACM Press.
- [SLX⁺14] J Shang, L Liu, F Xie, Z Chen, J Miao, and X Fang. A real-time detecting algorithm for tracking community structure of dynamic networks. *arXiv preprint arXiv* ;, 2014.
- [SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8) :888–905, 2000.
- [SSA06] Sulayman Sowe, Ioannis Stamelos, and Lefteris Angelis. Identifying knowledge brokers that yield software engineering knowledge in OSS projects. *Information and Software Technology*, 48(11) :1025–1033, 2006.
- [SSTM15] Natalie Stanley, Saray Shai, Dane Taylor, and Peter J. Mucha. Clustering Network Layers With the Strata Multilayer Stochastic Block Model. jul 2015.
- [SVB⁺11] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean François Pinton, Marco Quaggiotto, Wouter van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*, 6(8), 2011.
- [SWG15] Ingo Scholtes, Nicolas Wider, and Antonios Garas. Higher-Order Aggregate Networks in the Analysis of Temporal Networks : Path structures and centralities. page 27, aug 2015.
- [SWP⁺14] Ingo Scholtes, Nicolas Wider, René Pfitzner, Antonios Garas, Claudio J Tessone, and Frank Schweitzer. Causality-driven slow-down and speed-up of diffusion in non-Markovian temporal networks. *Nature communications*, 5 :5024, jan 2014.
- [TAD15] V. A. Traag, R. Aldecoa, and J. C. Delvenne. Detecting communities using asymptotical surprise. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 92(2), 2015.
- [Tra15] V. A. Traag. Faster unfolding of communities : speeding up the Louvain algorithm. mar 2015.
- [TYY16] Taro Takaguchi, Yosuke Yano, and Yuichi Yoshida. Coverage centralities for temporal networks. *The European Physical Journal B*, 89(2) :35, feb 2016.
- [VBB08] A Vespignani, M Barthelemy, and A Barrat. *Dynamical Processes on Complex Networks*. 2008.
- [VGB14] Christian L. Vestergaard, Mathieu Génois, and Alain Barrat. How memory generates heterogeneous dynamics in temporal networks. *Physical Review E*, 90(4) :042805, oct 2014.
- [VL14] Jordan Viard and Matthieu Latapy. Identifying roles in an IP network with temporal and structural density. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 801–806. IEEE, apr 2014.

- [WFAG12] Qinna Wang, Eric Fleury, Thomas Aynaud, and Jean-Loup Guillaume. Communities in evolving networks : definitions, detection and analysis techniques. *Dynamics of Time Varying Networks*, 2012.
- [WLWT10] Zhihao Wu, Youfang Lin, Huaiyu Wan, and Shengfeng Tian. A fast and reasonable method for community detection with adjustable extent of overlapping. In *Proceedings of 2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2010*, pages 376–379, 2010.
- [WWL⁺14] Yi Bo Wang, Wen Jun Wang, Dong Liu, Xiao Liu, and Peng Fei Jiao. Using Prior Knowledge for Community Detection by Label Propagation Algorithm. In *Advanced Materials Research*, volume 1049-1050, pages 1566–1571, nov 2014.
- [WZF14] Klaus Wehmuth, Artur Ziviani, and Eric Fleury. A Unifying Model for Representing Time-Varying Graphs. feb 2014.
- [XKS13] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks. *ACM Computing Surveys*, 45(4) :1–35, aug 2013.
- [XSL11] Jierui Xie, Boleslaw K. Szymanski, and Xiaoming Liu. SLPA : Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 344–349, 2011.
- [YL13] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale. In *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM '13*, page 587, New York, New York, USA, feb 2013. ACM Press.
- [YL15] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1) :181–213, jan 2015.
- [YWW13] L. Yu, B. Wu, and B. Wang. LBLP : link-clustering-based approach for overlapping community detection. *Tsinghua Science and Technology*, 18(4) :–, 2013.
- [Zha15] Pan Zhang. A revisit to evaluating accuracy of community detection using the normalized mutual information. jan 2015.