

THÈSE
présentée pour obtenir le grade de

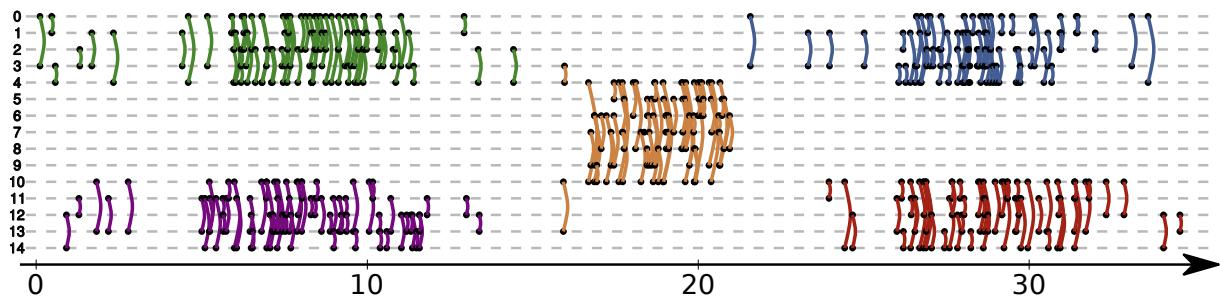
DOCTEUR DE L'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité INFORMATIQUE

École doctorale Informatique, Télécommunications et Électronique (Paris)

GROUPES ET COMMUNAUTÉS DANS LES FLOTS DE LIENS: DES DONNÉES AUX ALGORITHMES

Noé GAUMONT



Soutenue publiquement le 11 Octobre 2016 devant le jury composé de

<i>Rapporteurs :</i>	David CHAVALARIAS Jean-Loup GUILLAUME	Chargé de recherche, CNRS Professeur, Université de la Rochelle
<i>Examinateurs :</i>	Bertrand JOUVE Catherine MATIAS Gilles TREDAN Véronique SERFATY	Directeur de recherche, CNRS Directrice de recherche, CNRS Chargé de recherches, CNRS Responsable scientifique, DGA
<i>Directeurs :</i>	Clémence MAGNIEN Matthieu LATAPY	Directrice de recherche, CNRS Directeur de recherche, CNRS

Table des matières

Remerciements	ii
Introduction	1
1 État de l'art sur la détection de communautés et les réseaux dynamiques	5
1.1 Définition dans les graphes	6
1.1.1 Groupes, partitions et couvertures	7
1.1.2 Comparaison de partitions et couvertures	8
1.2 Communautés dans les graphes	9
1.2.1 Parititons de nœuds	10
Méthodes utilisant un modèle	10
Méthodes utilisant des marches aléatoires	12
1.2.2 Couverture de noeuds	13
Extension des méthodes de détection de partitions	14
Méthodes utilisant des communautés égocentrées	15
1.3 Extension temporelle des graphes	17
1.3.1 Extensions avec pertes d'informations temporelles	18
Séries de graphes	18
Tenseur 3D	20
Graphes multicouches	20
1.3.2 Extension sans perte d'information temporelle	21
Graphe temporel	21
Flot de liens	22
1.4 Bilan	24
2 Flots de liens : extension temporelle des graphes	27
2.1 Définition	27
2.2 Sous-flots	28
2.3 Degré et densité	29
2.4 Liste des notations pour les flots de liens	31
2.5 Manipulation concrète des flots de liens	31
3 Étude de la structure d'une archive de courriels en tant que flot de liens	35
3.1 Prétraitement sur le jeu de données	36
3.2 Caractéristiques élémentaires des discussions	37
3.3 Étude des discussions en tant que sous-flots	39
3.3.1 Application de la Δ -densité	39
3.3.2 Répartition temporelle et structurelle des discussions	42
3.3.3 Flot quotient	43
3.4 Détection de structures denses	45
3.4.1 Méthode de détection	45

3.4.2	Comparaison des partitions	46
3.4.3	Conclusion et perspectives	49
Conclusion	49	
Perspectives	50	
4	Détection de groupes pertinents dans les flots de liens	53
4.1	Travaux existants	54
4.2	Détection de groupes pertinents	55
4.2.1	Définition des voisinages	56
4.2.2	Définition de l'évaluation	58
4.2.3	Algorithme de calcul des scores	59
Voisinage aux nœuds	59	
Voisinage à la durée	60	
Voisinage au temps de début	60	
4.3	Jeux de données	61
4.4	Application	62
4.4.1	Évaluation des groupes candidats	63
Étude manuelle d'un groupe de Socio Pattern	64	
Étude manuelle d'un groupe de Rollernet	66	
4.4.2	Caractéristiques des groupes pertinents	68
Caractéristiques topologiques	68	
Caractéristiques temporelles	69	
4.5	Conclusion et perspectives	70
4.5.1	Conclusion	70
4.5.2	Perspectives	71
Amélioration de la détection des groupes pertinents	71	
Amélioration de la validation des résultats	72	
5	<i>Expected Nodes</i> : communautés de liens dans les graphes statiques	73
5.1	Travaux existants	74
5.2	Définition d' <i>Expected Nodes</i>	78
5.3	Comparaison	81
5.3.1	Cas du graphe complet	81
5.3.2	Cas de deux cliques chevauchantes	82
5.3.3	Graphe LFR	85
5.4	Calcul et optimisation	88
5.5	Conclusion	90
5.5.1	Perspectives	91
Amélioration de l'optimisation d' <i>Expected Nodes</i>	91	
Cas d'utilisation d' <i>Expected Nodes</i>	92	
6	Génération de flots de liens avec structure communautaire	95
6.1	Travaux existants	96
6.1.1	Séries de graphes	96
6.1.2	Flots de liens	97
6.2	Méthode de génération	98
6.2.1	Caractéristiques du générateur	100
6.3	Applications	101

6.3.1	Étude de méthodes statiques appliquées aux flots de liens	101
6.3.2	Vers des fonctions de qualité dans les flots de liens	103
	Jeux de paramètres pour la génération de flots de liens	103
	Proposition de fonctions de qualité	104
	Résultats	105
6.4	Conclusion	107
6.4.1	Perspectives	108
7	Bilan	111
7.1	Résumé et contributions apportées	111
7.2	Perspectives	113
	7.2.1 Extensions du formalisme de flot de liens	113
	7.2.2 Approfondissement des applications dans les flots de liens	114
A	Détection de groupes pertinents dans les flots de liens	117
A.1	Rollernet dataset	117
A.2	Baboon dataset	119
A.3	Reality mining dataset	121
Bibliographie		123

Introduction

Un réseau complexe est constitué d'un grand ensemble d'éléments interagissant entre eux. Avec cette définition, le nombre d'objets pouvant être considérés comme un réseau complexe est énorme. Il peut s'agir de réseaux sociaux constitués de personnes communiquant entre elles, de réseaux de routes entre des villes, de réseaux d'ordinateurs échangeant de l'information ou bien même de réseaux de neurones collaborant entre eux via leurs synapses. Les réseaux complexes sont donc un très vaste sujet d'étude et les graphes y ont une place prépondérante. En effet, les graphes sont constitués de nœuds et de liens où chaque lien relie deux nœuds. Ainsi, il est naturel de représenter un réseau de personnes par un graphe dans lequel une personne est représentée par un nœud et une interaction entre deux personnes par un lien entre deux nœuds.

Les questions posées sur les réseaux complexes donnent lieu à de nombreux axes de recherche liés aux graphes. La recherche de structures communautaires dans les graphes en est un exemple et il s'agit d'ailleurs d'un sujet activement étudié. On peut définir intuitivement une communauté comme étant un sous-ensemble d'éléments densément connectés entre eux mais peu densément connectés avec le reste du réseau. Dans le cas d'un réseau de personnes, une communauté est un groupe de personnes, par exemple un groupe d'amis communiquant beaucoup entre eux. Avec l'ensemble des communautés existantes dans un réseau, il est ainsi simple d'étudier un réseau dans sa globalité. La recherche de communautés est donc un moyen de construire une vue d'ensemble du réseau. Cette vue d'ensemble devient cruciale lorsque le réseau étudié est grand ; jusqu'à un milliard d'utilisateurs dans le cas de Facebook. Or, de plus en plus de jeux de données de tailles comparables sont disponibles depuis plusieurs années.

Il n'est cependant pas toujours possible de représenter un réseau par un graphe. C'est le cas lorsque le réseau évolue au fil du temps. Par exemple, deux personnes communiquent seulement durant les intervalles de temps où elles se téléphonent et les routes d'un réseau routier sont construites, voir détruites, à certains moments. L'information temporelle est alors très importante et on parle de réseau dynamique. Tous les réseaux dynamiques ne sont cependant pas équivalents dans leur évolution. Le réseau routier évolue lentement et il existe en permanence une structure en réseau. Le réseau des personnes communiquant par téléphone change très vite et les interactions existantes à un instant donné ne permettent pas de représenter le réseau dans sa globalité. Dans ce dernier cas, on parle alors de séquence d'interactions plutôt que de réseau dynamique car il n'existe pas de structure de réseau pertinente à un instant donné dans les séquences d'interactions.

Il est nécessaire de distinguer les instants pendant lesquels les liens existent pour étudier les réseaux dynamiques et les séquences d'interactions. Or, un graphe ne permet pas de tirer parti de cette information temporelle car tous les liens sont équivalents. Il faut donc transformer la notion même de graphe afin de tenir compte de la temporalité du réseau. Les graphes temporels sont une extension des graphes qui considèrent

l'ensemble des ajouts et retraits de liens via une fonction de présence. Ils sont particulièrement adaptés pour étudier les réseaux dynamiques car une structure de graphe existe à chaque instant mais ils le sont moins pour l'analyse de séquences d'interactions. L'approche que nous utilisons est la manipulation de flots de liens qui sont des ensembles de quadruplets (b, e, u, v) , où chaque quadruplet représente un lien entre les nœuds u et v existant durant l'intervalle $[b, e]$. Avec ce formalisme, on représente ainsi facilement que deux personnes ont interagi durant un intervalle de temps donné.

Avec toute l'information temporelle, on peut décrire une séquence d'interactions de manière plus fine qu'avec un graphe. Par exemple dans un réseau de personnes, une fête de famille ou une réunion de travail se matérialisent dans le flot de liens par des ensembles d'interactions reliant un ensemble de personnes durant un intervalle de temps. Ces ensembles d'interactions sont particuliers car ils concernent un nombre réduit de nœuds durant un court intervalle de temps. Avec un graphe, ces interactions ne sont plus distinguables de l'ensemble des interactions ayant eu lieu à d'autres moments. C'est pourquoi, le formalisme de flots de liens permet une description plus fine de ce type de structure. Pour arriver à ce résultat, il est nécessaire de pouvoir évaluer les caractéristiques d'un flot de liens. Nous construisons donc de nouveaux concepts et métriques adaptés à ce contexte en nous inspirant de la théorie des graphes.

Contrairement aux approches existantes manipulant des séquences d'interactions, nous nous basons sur un formalisme global de manière à ce que l'ensemble des notions définies soient cohérentes entre elles. Nous utilisons le formalisme de flot de liens et non celui de graphe temporel car les flots de liens sont plus à même de modéliser les séquences d'interactions. Dans cette thèse, nous nous attachons donc à la découverte des possibilités offertes par le formalisme de flots de liens lorsqu'il est appliqué à l'étude de la structure communautaire sous la forme d'ensembles d'interactions.

Plan du manuscrit

Dans le chapitre 1, nous présentons l'état de l'art en deux parties : la détection de communautés dans les graphes et les formalismes étendant la théorie des graphes pour prendre en compte le temps. La première partie est importante pour comprendre en détail les différentes définitions de communautés qui existent et les algorithmes de détections qui en découlent. La deuxième partie sur les extensions de graphe est primordiale pour comprendre les intuitions derrières chaque extension et ce qu'elles impliquent en terme de structures détectables. Nous présentons par la suite l'ensemble des définitions et notations que nous utilisons pour étudier un flot de liens dans le chapitre 2.

Nous appliquons ces notions à l'étude d'un réseau réel dans le chapitre 3. Il s'agit d'un jeu de données regroupant les courriels envoyés sur une liste de diffusion. Chaque courriel de la liste est associé à une discussion (*thread*) qui est connue. En utilisant la notion de densité définie pour les flots de liens, nous étudions la structure que forment les discussions. Ces travaux sont à mettre en parallèle avec les résultats décrivant les structures communautaires dans les graphes.

À partir de ces observations, nous développons, dans le chapitre 4, une méthode permettant de trouver automatiquement des ensembles d'interactions qui sont jugés pertinents. Un ensemble d'interactions est jugé pertinent s'il est plus dense que son

voisinage temporel et topologique. Ainsi, nous évaluons si les liens ayant eu lieu entre différents noeuds sur un intervalle donné sont surprenants. Les ensembles de liens pertinents forment ainsi une structure partielle du flot de liens car certains liens ne sont dans aucun groupe pertinent. Nous testons cette méthode sur quatre jeux de données réels. Ces travaux forment une part importante de cette thèse sur les structures communautaires dans les flots de liens.

Dans le chapitre 5, nous revenons sur les structures communautaires des graphes sous la forme de partitions de liens. En effet, il n'existe que peu de méthodes traitant de sujet. Nous proposons une nouvelle méthode pour évaluer la qualité d'une partition de liens qui s'inspire de ce qui est fait pour les partitions de noeuds avec la modularité. Nous testons notre fonction de qualité et les méthodes existantes sur des exemples synthétiques.

Fort des observations faites dans l'ensemble de ces travaux, nous esquissons de premiers travaux sur la génération de flots de liens et sur les fonctions de qualité dans les flots de liens dans le chapitre 6. Nous proposons dans ce chapitre un modèle de génération de flots de liens et nous testons différentes méthodes de détection et d'évaluation de la structure générée.

Enfin nous revenons dans le chapitre 7 sur les travaux réalisés au cours de cette thèse et des contributions apportées avant de présenter quelques pistes de recherches qu'ouvrent nos travaux.

Chapitre 1

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

Sommaire

1.1	Définition dans les graphes	6
1.1.1	Groupes, partitions et couvertures	7
1.1.2	Comparaison de partitions et couvertures	8
1.2	Communautés dans les graphes	9
1.2.1	Parititons de nœuds	10
1.2.2	Couverture de nœuds	13
1.3	Extension temporelle des graphes	17
1.3.1	Extensions avec pertes d'informations temporelles	18
1.3.2	Extension sans perte d'information temporelle	21
1.4	Bilan	24

Dans cette thèse, nous étudions deux axes de recherches liés aux graphes qui sont orthogonaux. Avant de présenter ces deux axes de recherches dans les sections 1.2 et 1.3, nous revenons dans la section 1.1 sur quelques définitions et notations utiles pour manipuler des graphes.

Le premier axe de recherche est la détection de structures dans les graphes et plus particulièrement de communautés. Une communauté est une partie du graphe telle qu'il existe beaucoup de liens à l'intérieur de la communauté et moins avec le reste du graphe. Cependant aucune définition ne fait l'unanimité. La notion de communauté dépend du contexte et de la méthode. Malgré cette définition floue, des structures communautaires ont été trouvées dans de nombreux graphes dans plusieurs domaines tels que le réseau constitué des régions du cerveau [dRSKvdH14], un 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.

Le second axe de recherche est la prise en compte du temps dans la théorie des graphes. La première approche fût de complètement ignorer l'information temporelle et de considérer que tous les liens apparaissent en même temps ; on parle alors de graphe agrégé. Cependant, la modélisation d'un réseau sous la forme d'un graphe agrégé manque de pertinence, lorsque le réseau change trop au cours du temps [Hol15b]. Si tel est le cas, certaines structures présentes dans le graphe peuvent n'être que des artefacts de l'agrégation temporelle. Imaginons par exemple un graphe représentant

les alliances politiques dans un pays sur une longue période. Si toutes les alliances politiques sont agrégées, alors les personnes changeant de parti politique seront faussement considérées comme influentes car connectées à plusieurs partis alors qu'elles peuvent ne plus avoir de contact dans leur ancien parti. Ainsi, il n'est plus possible d'observer la répartition des alliances politiques dans ce genre de réseau si l'agrégation temporelle est trop importante [MRM⁺10].

Si on prend en compte le temps, alors il est possible de détecter des structures plus fines que dans le cas statique. En effet, il devient possible de détecter des instants où l'organisation générale change [RB10], de comprendre quels sont les instants où un nœud est important [MT15, CVW⁺15, TYY16], ou bien de détecter des groupes temporels [CAH10]. Pour ce faire, plusieurs extensions de la théorie des graphes ont été proposées et elles sont présentées dans la section 1.3.

1.1 Définition dans les graphes

Un graphe G est défini par un couple (V, E) où V est un ensemble de nœuds et $E \subseteq V \times V$ est un ensemble de liens ; chaque lien étant une paire de nœuds. Sauf mention contraire, nous considérons des graphes non-orientés, c'est-à-dire pour toute paire 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 et aucun lien de la forme (u, u) . Ceci est en opposition avec les graphes multiples où il peut exister plusieurs liens entre deux nœuds.

Par convention, nous notons $n = |V|$ le nombre de nœuds et $m = |E|$ le nombre de liens. On suppose en général que $V = \{1, \dots, n\}$. Il est possible de représenter un graphe G par une matrice carrée A de taille n où l'élément $A_{i,j} \forall i, j \in V$ est égale à 1 si un lien relie les nœuds i et j et 0 sinon. Avec ces notations, nous définissons les notions suivantes :

Degré Le degré d'un nœud i , noté d_i , est égal au nombre de liens reliés à i : $d_i = \sum_{j \in V} A_{i,j}$;

Densité La densité, $\delta(G)$ d'un graphe est la probabilité que deux nœuds pris au hasard soient reliés par un lien : $\delta(G) = \frac{2m}{n(n-1)}$;

Degré moyen Le degré moyen $\tilde{\delta}(G)$ est égal à : $\tilde{\delta}(G) = \frac{2m}{n}$;

Chemin Un chemin dans un graphe est une suite de liens $((u_1, v_1), \dots, (u_k, v_k))$ de telle sorte que $v_i = u_{i+1} \forall i \in [1, k-1]$;

Graphe connexe Les nœuds d'un graphe sont dit connexes s'il existe un chemin entre toutes les paires de nœuds du graphe ;

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 connexe ;

Arbre Un arbre est le graphe connexe le moins dense et il est constitué de $n - 1$ liens pour n nœuds ;

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) = E \cap (X \times X) $	nombre de liens entre les nœuds de X
$l_{out}(X) = 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 liens 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 liens 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 .

Clique Une clique, aussi appelée graphe complet, est le graphe le plus dense et elle est composée de $\frac{n(n - 1)}{2}$ liens pour n nœuds ;

Sous-graphe Un graphe $G' = (V', E')$ est un sous-graphe de G si et seulement si $V' \subseteq V$ et $E' \subseteq E$;

Graphe induit Le graphe induit par un ensemble de nœuds $V' \subseteq V$ est défini par V' et $E' = E \cap (V' \times V')$.

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, notons $X \subseteq V$ un ensemble de nœuds et $Y \subseteq E$ un ensemble de liens. Nous listons les notations utilisées dans le tableau 1.1.

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

1. La partition recouvre l'ensemble des nœuds : $\bigcup_i V_i = V$.
2. Les ensembles de nœuds sont disjoints : $V_i \cap V_j = \emptyset \forall i, j \in [1, k], i \neq j$.

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. Ainsi, une couverture de nœuds, \mathcal{V} , est aussi un ensemble d'ensembles de nœuds. L'union des ensembles doit également être égale à V mais en revanche deux

ensembles de noeuds peuvent partager un noeud : $\exists i, j \in [1, k], i \neq j V_i \cap V_j \neq \emptyset$. Une couverture est parfois aussi appelée partition chevauchante.

Nous avons détaillé les notions de partitions et couvertures de noeuds mais les mêmes définitions valent pour les partitions et couvertures de liens. Enfin, il existe d'autres structures possibles comme les structures non-exhaustives où un élément peut n'appartenir à aucun ensemble.

1.1.2 Comparaison de partitions et couvertures

Il est souvent utile de pouvoir comparer deux partitions ou deux couvertures entre elles. Le but est de calculer une similarité entre deux structures de telle sorte que la similarité est égale à 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 des méthodes tirant parti de la structure d'ensembles et celles provenant de la théorie de l'information.

Approches ensemblistes Il est possible de comparer deux ensembles X et Y en utilisant l'indice de Jaccard, $\mathbb{J}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$, ou le *F1 score* $F_1(X, Y) = 2 \frac{|X \cap Y|}{|X| + |Y|}$. Avec l'indice de Jaccard ou le *F1 score*, 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 :

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

Il existe d'autres méthodes pour symétriser la similarité, *e.g.* la moyenne harmonique. Cette méthode peut s'appliquer indifféremment aux partitions et aux couvertures.

Il existe également le *Rand Index* [Ran71] qui lui ne s'applique qu'aux partitions. Il mesure le nombre de paires de noeuds qui sont classées de la même manière dans les deux partitions ; c'est à dire pour deux noeuds 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 paires de noeuds de telle sorte qu'ils soient dans le même ensemble dans les deux partitions, a_{00} le nombre de paires de noeuds de telle sorte qu'ils soient dans des ensembles différents dans les deux partitions et a_{10} (resp. a_{01}) le nombre de paires de noeuds 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 de la même manière une paire de noeuds, notamment à cause de a_{00} . C'est pourquoi une version ajustée du *Rand Index* (ARI) a été proposée [HA85]. Le *Rand Index* et sa version ajustée permettent de comparer des partitions. Porumbel *et al.* [PHK11] ont proposé l'*Omega Index* qui est une extension de l'ARI pour les couvertures.

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 $P(X) = n_X/|V|$. De manière similaire, la probabilité jointe est $P(X, Y) = |X \cap Y|/|V|$. 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 mutuelle $I(\mathcal{X}, \mathcal{Y})$. Cette dernière est définie par $I(\mathcal{X}, \mathcal{Y}) = H(\mathcal{X}) - H(\mathcal{X}|\mathcal{Y})$. L'entropie, $H(\mathcal{X})$, et l'entropie conditionnelle, $H(\mathcal{X}|\mathcal{Y})$, sont définies dans le sens de Shanon par :

$$H(\mathcal{X}) = - \sum_{X \in \mathcal{X}} P(X) \log(P(X)), \quad H(\mathcal{X}|\mathcal{Y}) = - \sum_{X \in \mathcal{X}, Y \in \mathcal{Y}} P(X, Y) \log \frac{P(X, Y)}{P(Y)}. \quad (1.4)$$

Afin de normaliser l'information mutuelle, Danon *et al.* [DDDGA05] ont défini l'information mutuelle normalisée (NMI_{shanon}) :

$$NMI_{shanon}(\mathcal{X}, \mathcal{Y}) = \frac{2I(\mathcal{X}, \mathcal{Y})}{H(\mathcal{X}) + H(\mathcal{Y})}. \quad (1.5)$$

Lancichinetti *et al.* l'ont par la suite étendue pour prendre en compte les 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 la comparaison de structures est assez restreinte comparé à celle sur la détection de communautés. En effet, il semble qu'uniquement 3 similarités soient couramment utilisées : la similarité se basant sur Jaccard, l'Omega index et la NMI. Tout 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 de la majeure partie 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 Communautés dans les graphes

Ce champ de recherche est très vaste et il est illusoire de vouloir énumérer les méthodes existantes dans ce domaine car elles sont extrêmement nombreuses et les caractéristiques voulues d'une communauté peuvent varier selon le contexte [LLDM08, CGP11, YL15, JBP⁺15]. Les méthodes se séparent tout de même en plusieurs catégories selon qu'elles capturent une partition ou une couverture de noeuds. 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 communautés sous la forme de partitions de liens que nous traitons dans le chapitre 5.

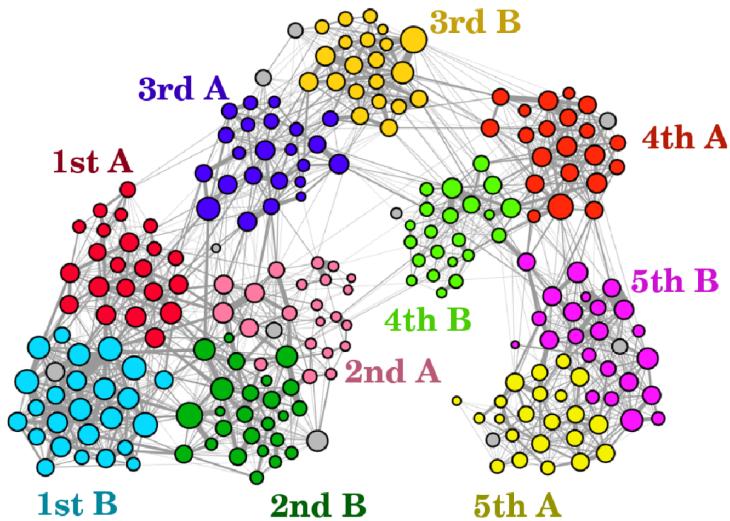


FIGURE 1.1 – Graphe de contact des enfants d'une école primaire. L'épaisseur du lien représente la durée de communication entre deux élèves. La couleur représente la classe de chaque élève et la taille représente le degré du nœud. Les professeurs sont en gris.¹

1.2.1 Parititons de nœuds

Afin de mieux comprendre ce que peut capturer une partition de nœuds, 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 1 mètre 50. Ce dispositif permet de mesurer les interactions entre élèves et de construire le graphe des relations à l'école. Un lien existe entre deux élèves s'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 interagissent beaucoup entre eux mais peu avec les élèves des autres classes. Cela se remarque particulièrement bien pour la classe 3A. Il existe beaucoup de liens entre les élèves de la classe 3A mais aucun avec les élèves de la classe 5A par exemple.

Afin de capturer des partitions de nœuds, beaucoup de méthodes existent. Il y a d'ailleurs régulièrement des états 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

Comme une communauté est souvent définie comme devant être très densément connectée, le problème est de trouver une fonction capable d'évaluer la qualité d'une communauté. Pour ce faire, il faut définir une métrique qui mesure la densité, *e.g.* le

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

nombre de liens dans un groupe. Puis, il est nécessaire de définir comment évaluer cette métrique.

Il est possible d'utiliser une métrique en la normalisant par ses bornes minimum et maximum. Avec une métrique normalisée entre 0 et 1, un groupe est alors considéré comme très connecté si son évaluation est supérieure à un certain seuil. Par exemple, le nombre de liens entre n nœuds peut être normalisé par le nombre de liens dans un arbre et dans une clique de taille n . Cette approche n'est cependant pas adaptée pour la recherche de communautés car elle ne tient pas compte de la structure du graphe². Prenons l'exemple du graphe constitué d'une unique clique. Dans ce graphe, tout groupe de nœuds est également une clique et par conséquent tout groupe de nœuds a une 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 ses valeurs minimum et maximum, il est intéressant de considérer l'écart à une valeur attendue. L'idée est la suivante : quelle serait la valeur attendue de la métrique considérée si le graphe n'avait pas de structure communautaire. Le problème est alors de "retirer" la structure communautaire du graphe ou bien de définir un ensemble de graphes similaires au graphe initial mais n'ayant pas de structure communautaire. Ce processus définit alors ce qu'on appelle un *modèle nul*. Détecter des communautés se traduit ainsi en trouver des groupes qui s'éloignent du modèle nul 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 comparés au graphe initial, des contraintes sont généralement ajoutées et donnent lieu à différents modèles nuls. 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 de configuration est utilisé car il a été observé que les graphes provenant de données réelles ont une distribution des degrés très éloignée d'une distribution uniforme. C'est pourquoi l'ajout de la contrainte sur les degrés permet de considérer des graphes plus proche du graphe initial. Il existe 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 entre -1 et 1 . Plus la valeur de modularité d'une partition est élevée, plus la partition est censée capturer une bonne structure communautaire. La modularité est définie de la manière suivante pour une partition \mathcal{C} :

$$Q(\mathcal{C}) = \frac{1}{2m} \sum_{i,j \in V} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta_{\mathcal{C}(i)=\mathcal{C}(j)}, \quad (1.6)$$

où $\delta_{\mathcal{C}(i)=\mathcal{C}(j)}$ est égale à 1 si i et j sont dans la même communauté et 0 sinon. Il s'agit pour deux nœuds d'une même communauté de comparer la présence ou l'absence

2. Cette approche est plus appropriée dans la recherche des groupes les plus denses [BBC⁺15].

d'un lien, A_{ij} , à la probabilité que ces noeuds soient reliés dans le modèle de configuration, $\frac{d_i d_j}{2m}$. L'idée sous-jacente est que les noeuds 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 NP-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 existants, 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 différentes variantes de la modularité [RB06, DYC10].

Stochastic Block Model Nous avons défini 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 noeuds soient reliés dépend uniquement de leur groupe respectif. Si le graphe a une structure communautaire³, alors deux noeuds d'une même communauté devraient avoir une forte chance d'être connecté. À l'inverse, deux noeuds de deux communautés différentes devraient avoir une probabilité assez faible d'être connecté. Le SBM est défini par de nombreux paramètres : le nombre de groupes, l'assignation d'un groupe à chaque noeud et les probabilités d'interactions entre les groupes. Avec un jeu de paramètres donné, il est possible de calculer la vraisemblance que ce jeu de paramètres soit à l'origine du graphe. Trouver une partition de noeuds 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 noeuds 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 noeuds a été proposée : le Degree-corrected Stochastic Block Model (DSBM) [KN11]. Enfin d'après des récents travaux [New16], il semblerait que le DSBM et l'optimisation de la modularité soient liés.

Méthodes utilisant des marches aléatoires

Il existe d'autres approches que celles utilisant un modèle nul ou un modèle génératif. Notamment, il y a les méthodes utilisant les marches aléatoires [PL05, RB08]. Ces méthodes tirent parti du fait que si une communauté est densément connectée alors

3. Il est également possible de représenter d'autres type de structures.

un marcheur aléatoire devrait y rester assez longtemps. En particulier, la méthode Infomap [RB08] repose sur une idée très élégante qui est la suivante. Une partition de noeuds 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 disjointes et la majorité des routes se trouvent à l'intérieur des départements. Dans une carte, il est courant qu'un nom de ville soit unique dans un département mais qu'un même nom puisse être utilisé dans plusieurs départements. De plus, un voyageur se déplaçant aléatoirement sur les routes a peu de chance de sortir d'un département. Pour décrire, *a posteriori*, l'ensemble des villes traversées par ce voyageur, il suffit alors de donner le département initial puis la liste des villes visitées. Il n'est pas nécessaire de répéter le département pour chaque ville si le voyageur n'en est pas sorti. En ce sens, la découpe d'un pays en département permet de réduire la complexité de la description 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 noeuds du graphe et les communautés comme zones du graphe. Si les communautés sont bien formées, alors les marcheurs aléatoires restent bloqués à l'intérieur des communautés et la description de leur marche est courte. La longueur de cette description devient alors la signature de la partition et plus la signature est courte, meilleure la partition est.

Le phénomène de résolution limite a également été étudié dans le cadre de Infomap [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 noeuds. Il y a les méthodes spectrales [DM04, MT09] qui se basent sur les vecteurs propres de la représentation d'un graphe sous la forme d'une matrice. De manière moins formelle, il existe également les méthodes de propagation de labels (LPA) [RAK07, LLJT14]. Dans ces méthodes, chaque noeud a initialement un label puis à chaque itération chaque noeud prend comme label un des labels de ses voisins. En général, un noeud prend comme label celui qui est le plus présent parmi ses voisins. Au bout d'un certain nombre d'itérations ou une fois à l'équilibre, il ne reste que quelques labels dans le graphe et ils représentent les communautés.

Résumé

Il existe de très nombreuses méthodes pour la détection de communautés en tant que partition de noeuds. Il semble 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 noeuds

Jusqu'à maintenant, nous avons considéré les communautés comme des partitions de noeuds. Or, les partitions sont très restrictives et ne peuvent pas capturer toutes les situations possibles. Prenons l'exemple d'un graphe reflétant des interactions entre personnes. Il existe des communautés qui sont disjointes comme le travail et la famille mais bien souvent des personnes appartiennent à plusieurs groupes, voir l'exemple

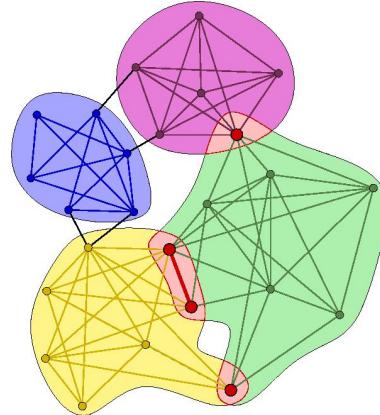


FIGURE 1.2 – Exemple graphique avec une structure communautaire chevauchante représentée par les couleurs⁴.

dans la figure 1.2. Ainsi le groupe des personnes faisant du sport et le groupe des personnes travaillant ensemble peuvent ne pas être disjoints. Si tel est le cas, alors il n'est plus possible de représenter ces communautés avec une partition. Il est nécessaire de manipuler une couverture de noeuds. Ainsi dans l'exemple dans la figure 1.2, les noeuds rouges appartiennent à 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. Il existe de nombreux état de l'art dans le domaine [Kan14, XKS13, BCS15, HDF14].

Une des premières méthodes de détection de couverture de noeuds 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 un noeud k qui serait relié à i aurait une forte chance d'être également connecté à j . 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 les communautés doivent être composées de plusieurs cliques. C'est pourquoi CPM cherche l'ensemble des cliques d'une taille k donnée⁵ puis fusionne toutes les cliques qui partagent suffisamment de noeuds, en général $k - 1$. Comme cette méthode est relativement coûteuse, Kumpula *et al.* [KKKS08] ont repris le même mécanisme en optimisant le mode de calcul.

Extension des méthodes de détection de partitions

La majorité des méthodes existantes pour les partitions ont été adaptées pour manipuler les couvertures de noeuds. Il existe notamment plusieurs extensions de la modularité [SCCH09, NMCM09]. Cependant ces extensions ne reposent plus sur un modèle nul car elles introduisent des termes de normalisation. 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 noeuds. Une extension du SBM est le Mix-Membership Stochastic Block Model (MMSBM)[ABFX08] qui permet à un noeud d'avoir plusieurs groupes. Dans [GB13],

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

5. En général, k est compris entre 3 et 5 pour des raisons de coût de calcul.

les auteurs utilisent la même méthode mais en échantillonnant le graphe initial afin de réduire le coût de calcul. Il existe d'autres méthodes à base de modèle génératif [BKN11, YL13]. Ces méthodes se basent sur des graphes d'affiliations [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 Ball *et al.* [BKN11], ce graphe d'affiliation est pondéré et la pondération représente la propension d'un nœud à créer des liens dans un groupe donné tandis que pour Yang *et al.* il s'agit du facteur d'appartenance du nœud 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 en matrices non-négatives [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 telles que leur combinaison permette de retrouver la matrice initiale.

Jusque récemment ce genre de technique généralisant le SBM ne pouvait s'appliquer qu'à des graphes relativement petits. Cependant les méthodes récentes [GB13, YL13] arrivent à traiter des graphes ayant énormément de liens, de l'ordre 10^9 à 10^{12} liens.

Propagation de label Des méthodes ont étendu 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 labels. Ainsi à la fin de l'algorithme, il suffit de choisir les labels les plus fréquents. Cependant ce mécanisme de propagation change radicalement 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 avec le SBM de la méthode étant la moins "déformée" par l'adaptation aux couvertures de nœuds. 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 dans ce contexte. Ce travail d'extension a été fait par Esquivel *et al.* [ER11].

Méthodes utilisant des communautés égocentriques

On a vu avec la modularité qu'il est assez difficile de définir une fonction de qualité évaluant une couverture de nœuds en fonction des caractéristiques des groupes. C'est pourquoi certaines méthodes s'affranchissent complètement de fonction de qualité globale et se concentrent sur des propriétés locales. Dans cette philosophie, il y a les méthodes se basant sur des fonctions de proximité et celle utilisant des fonctions de qualité locales.

Dans le premier cas, le but est de mesurer la proximité entre tous les nœuds et un nœud appelé égo. Puis une fois les nœuds ordonnés selon leur similarité, il suffit de définir une coupe pour séparer les nœuds appartenant à la même communauté que l'égo et les autres. Le choix de la coupe se fait, en général, en fonction de la présence de forte décroissance de la similarité. Pour obtenir une couverture de nœuds,

il est nécessaire d'appliquer ce processus égocentré sur plusieurs égos. Danisch *et al.* [DGG12] utilise une méthode de diffusion pour mesurer la similarité, alors que Whang *et al.* [WGD13] utilise le *Personalized Page Rank*. Ces méthodes ne sont cependant pas à même d'évaluer les communautés qu'elles trouvent.

D'autres méthodes prennent le contre-pied des méthodes de proximité en utilisant des fonctions de qualité locales. 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 pas étendre un groupe indéfiniment, un ajout n'est fait que s'il améliore la fonction de qualité locale. Ainsi dans ce type d'algorithme, il y a deux critères importants : le choix des communautés initiales et le critère d'évaluation. Dans la majorité des cas, chaque nœud constitue initialement 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 applicables sont très nombreuses et nous n'en mentionnerons 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 l_{in} de liens internes à un groupe de nœuds et le nombre de liens total $l_{in} + l_{out}$: $\frac{l_{in}}{l_{in} + l_{out}}$. En effet, plus le degré relatif est élevé, plus le groupe est dense comparé à son voisinage et plus il a de fortes chances d'être une bonne communauté. Cette formulation est assez proche de la conductance ou coupe normalisée [SM00] :

$$\phi = \frac{l_{out}}{\min(2(l_{in} + l_{out}), m - 2(l_{in} - l_{out}))} \quad (1.7)$$

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 dans la communauté. C'est ce que mesure la cohesion [FCF11] pour un groupe de taille k :

$$cohesion = \frac{\Delta_3}{\binom{k}{3}} \times \frac{\Delta_3}{\Delta_3 + \Delta_2}, \quad (1.8)$$

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

D'autres 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 grands graphes tout comme les algorithmes de propagations de label mais elles profitent des fonctions de qualité locales qui permettent d'une certaine manière d'évaluer le résultat obtenu.

Résumé

La littérature sur la détection de couverture de nœuds 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 capables 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. Les modèles génératifs ne semblent en effet plus limités à de petits graphes. Les méthodes utilisant des fonctions de qualité locales sont quant à elle très rapides et le choix de la fonction de qualité permet de s'adapter au contexte.

1.3 Extension temporelle des graphes

Les graphes sont utilisés pour représenter une situation ou résoudre un problème réel. Dans la section précédente, 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 mais elle ne l'est plus lorsque l'objet d'étude évolue beaucoup.

Afin d'illustrer ce propos, nous 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 s'ils ont interagi au moins une fois au cours de la capture. 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 voire plusieurs années. Dans ce cas de figure, il ne sera plus possible de dégager un comportement général des élèves car il aura 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 d'une 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 est un très bon exemple de l'importance du temps. Un modèle épidémique représente 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 interactions entre personnes. En plus du graphe, il est nécessaire d'ajouter l'état de chaque nœud, *e.g.* sain ou infecté. Ainsi, un nœud sain ne peut devenir infecté que s'il est relié à un nœud infecté dans le graphe. Ce genre de modèles met donc en avant un chemin de diffusion, *e.g.* une suite de personnes transmettant la maladie à l'autre.

Afin d'illustrer ce phénomène, les premiers travaux [VBB08] s'appuient sur un graphe d'interactions 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 pas 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 en tenir compte. 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 quels 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

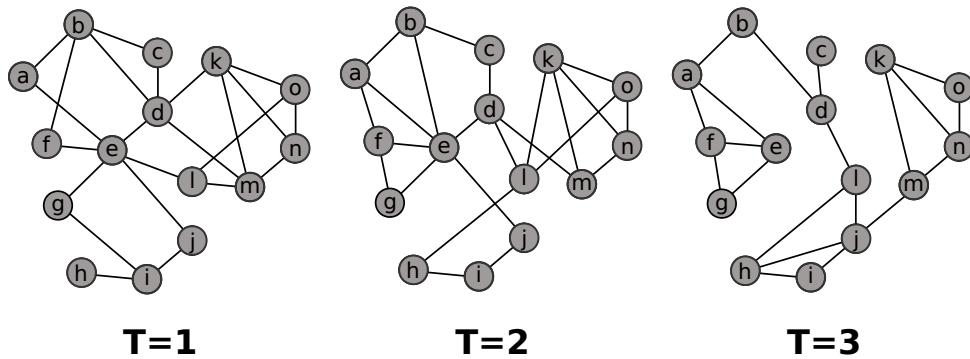


FIGURE 1.3 – Exemple de série de graphes sur trois intervalles de temps.

La première solution qui a été apportée ne prend le temps en compte que partiellement. Il s'agit de manipuler une série de graphes où chaque graphe représente le réseau durant un intervalle de temps donné. Ainsi, il est possible d'appliquer les outils de la théorie des graphes sur chaque intervalle. Cependant chaque intervalle de temps est représenté par un graphe agrégé. Il y a donc toujours une perte d'information. Plus formellement, une série de graphe est définie 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 la journée. Un graphe agrégeant 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 d'utiliser une autre mesure que la seconde mais plutôt le nombre de changements 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 changements 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 graphes, 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 communautés consiste à comprendre comment une communauté donnée évolue, étant donné une série de graphes et une série de partitions. Palla *et al.* [PBV07] ont été parmi les premiers à décrire les évolutions possibles d'une communauté. Muni d'indicateurs de similarité tels que l'indice de Jaccard, voir la section 1.1.2, il est possible de trouver les communautés les plus proches aux instants précédent et suivant. À partir de ces informations, six types d'évolutions d'une communauté sont définis :

Naissance Une nouvelle communauté apparaît.

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.

Retrecissement La communauté continue d'exister mais perd quelques nœuds.

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

Ce type de méthodes souffre souvent de l'instabilité des méthodes de détection [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].

Des extensions des Stochastic Block Models ont également été proposées par différents auteurs dans le cadre des séries de graphes. Yang *et al.* [YCZ⁺11] sont parmi les premiers à considérer ce cas de figure. Ils considèrent que la probabilité d'un lien entre deux communautés est fixe et que ce qui change est l'affiliation des nœuds au cours du temps. Ce processus de changement de communauté suit alors une chaîne de markov cachée. À l'inverse, Corneli *et al.* [CLR16] considèrent une partition de nœuds fixe tout au long du temps et c'est l'activité entre deux communautés qui change selon l'intervalle de temps. Xu *et al.* [XH14] permettent à l'affiliation et à l'activité entre deux communautés de changer selon l'intervalle de temps. Cependant, il semblerait que cette relaxation se fasse au dépens de l'identifiabilité des paramètres d'après Matias *et al.* [MM15]. Ils ont donc proposé une nouvelle méthode afin de résoudre ce problème. De plus, leur méthode permet de traiter des séries de graphes pondérés.

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. Cependant ce genre de décomposition semble moins expressive que les SBM.

Graphes multicouches

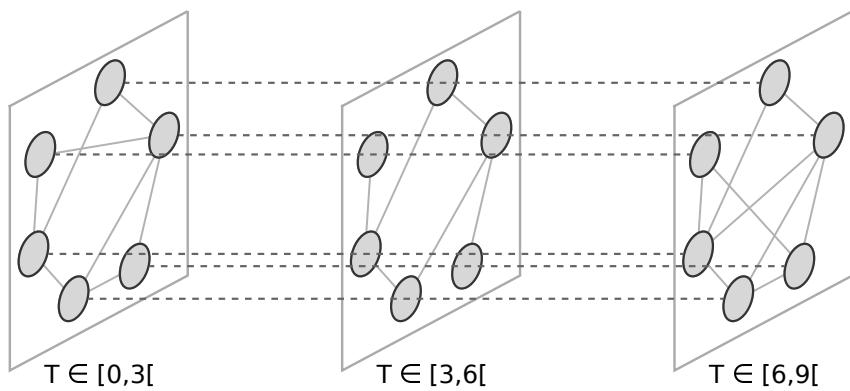


FIGURE 1.4 – Graphe multicouches avec trois couches représentant le temps. Les liens pleins (resp. pointillés) sont les liens intra-couches (resp. inter-couches).

La construction de graphes multicouches (*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 durant l'intervalle de temps. Cependant, le graphe multicouches ajoute des liens entre les nœuds de deux intervalles, voir l'illustration 1.4. Par conséquent, un graphe multicouches est un graphe où il existe deux types de liens, les liens intra-couches et les inter-couches et il existe T répliques d'un même nœud, une par intervalle de temps. Les liens intra-couches représentent un connexion entre deux noeuds 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 général limités à relier deux couches consécutives.

Les graphes multicouches représentent les données évoluant dans le temps mais ils modélisent aussi très bien d'autres situations. Par exemple, ils permettent de représenter facilement les différents 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 multicouches, 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 multicouches est un graphe, il est possible d'adapter les méthodes existantes pour tenir compte des

différents types de liens. C'est le cas d'infomap [DLAR14], de la modularité [MRM⁺10, BPW⁺13, BPW⁺16] et du SBM [SSTM15, Pei15].

Résumé

Les séries de graphes, les tenseurs et les graphes multicouches permettent de prendre en compte le temps tout en autorisant l'utilisation de méthodes conçues sur un graphe statique. Cette liberté d'utilisation a un coût. Ces approches reposent sur une découpe du temps en sous-intervalles durant lesquelles le temps n'est plus pris en compte afin d'obtenir un graphe statique. Or, il peut être délicat de définir ces intervalles de temps. De plus, 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 perte d'information temporelle

Graphe temporel

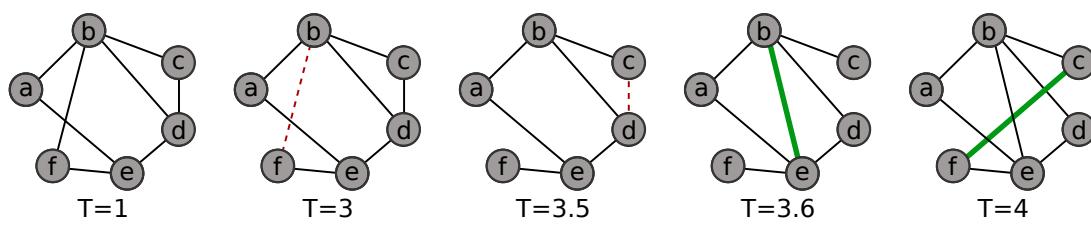


FIGURE 1.5 – Graphe temporel avec des ajouts de lien représentés en trait épais vert et des retraits de lien représentés par des liens pointillés rouge.

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 présence en fonction de temps qui vaut 1 à un instant t si le lien existe à cet instant et 0 sinon. Ainsi, il est possible de connaître la structure de graphe à chaque instant. Ce formalisme est présenté dans différents travaux [CFQS11, WZF15] et illustré dans la figure 1.5. Dans cette figure, on voit apparaître l'ordre de modification du graphe. Tout d'abord, les liens (b, f) et (c, d) disparaissent puis les liens (b, e) et (f, c) apparaissent chacun leur tour.

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 impactés par ce changement. En fonction de l'évolution de cette métrique, il est alors décidé d'ajouter ou retirer un nœud voire de fusionner deux communautés. Li *et al.* [LHB⁺12] se base sur le nombre de liens que partage un nœud avec les communautés environnantes. Ainsi, un nœud est toujours dans la communauté avec laquelle il partage le plus de liens. Shanget *al.* [SLX⁺14], Cordeiro *et*

al. [CSG16] et *Sunet al.* [SHZ⁺14] se basent sur l'évolution de la modularité. Cependant, ces approches ne permettent pas l'ensemble des évolutions de communauté possibles, 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] afin d'améliorer l'évolution de la partition. Ils utilisent une métrique locale basée 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. Ainsi, ils assurent une meilleure qualité de la partition au cours de l'évolution.

Flot de liens

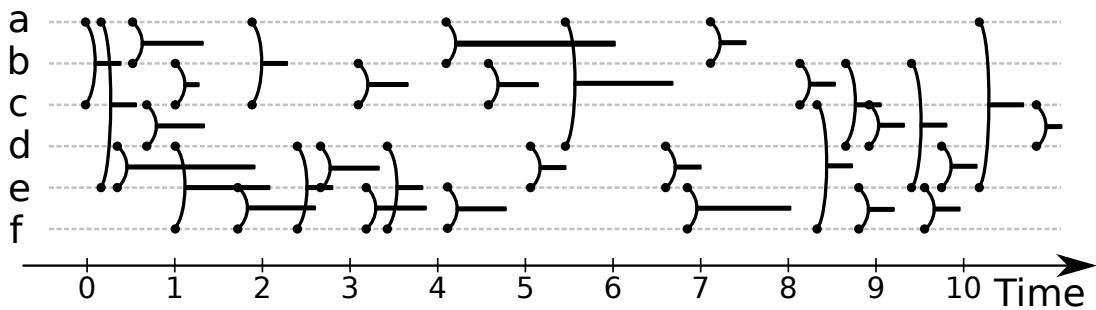


FIGURE 1.6 – Flot de liens entre 6 nœuds, représentés sur l'axe des ordonnées, au cours du temps représenté sur l'axe des abscisses. Dans l'exemple, il existe un lien entre *a* et *b* durant l'intervalle [4, 6].

Dans les graphes temporels, toute l'information temporelle est gardée. Cependant, l'intuition derrière cette méthode est qu'il existe une structure de graphe à chaque instant. Cette hypothèse n'est pas toujours vérifiée, en particulier lorsque les liens apparaissent et disparaissent très rapidement. C'est le cas des appels téléphoniques qui durent rarement plus d'une heure ou bien de manière plus frappante avec les SMS et les courriels qui n'ont même pas de durée. Dans ces contextes, il n'est pas possible de supposer qu'à chaque instant une structure de graphe existe. Il faut donc un formalisme et des mesures qui s'adaptent à ce contexte. C'est pour répondre à ce besoin que le formalisme de flot de liens a été pensé. Le but est de construire un objet ne présupposant aucune structure et qui stocke toute l'information disponible. Même si le formalisme ne présuppose aucune contrainte structurel, il se peut en revanche que le réseau représenté en ait. Par exemple dans les télécommunications, une personne ne peut appeler qu'une ou deux personnes en même temps. Plusieurs travaux [HS13, Hol15b, Hol15a] font un tour d'horizon des méthodes existantes pour étudier les flots de liens qui sont souvent appelés *temporal networks* mais il n'existe, à notre connaissance, qu'un seul travail donnant un fondement théorique solide au flot de liens. Il s'agit de *Batagel et al.* [BP16] qui se basent sur l'algèbre. Malheureusement avec une formulation purement algébrique, il semble difficile de transcrire l'ensemble des notions de graphes pour l'instant.

Prenons l'exemple des appels téléphoniques, c'est-à-dire qui parle avec qui à quelle heure et pendant combien de temps. Un appel peut donc être représenté par un quadruplet (b, e, u, v) où b (res. e) est le début (resp. la fin) de l'appel et u et v représentent des personnes. Il est donc possible de modéliser des appels téléphoniques par un ensemble de quadruplet. En faisant cela, aucune information n'est perdue et, en ce sens, flots de liens et graphes temporels sont équivalents. Cependant, ce changement de perspective induit une réflexion différente selon le formalisme considéré.

Les différences dans les méthodes de représentation des graphes temporels de la figure 1.5 et celle des flots de liens de la figure 1.6 illustrent bien ce changement de perspective, bien que les deux figures ne représentent pas le même réseau. Dans l'exemple de la figure 1.6, les nœuds sont représentés sur l'axe des ordonnées et le temps sur l'axe des abscisses. Un lien dans cette visualisation est représenté par : un arc vertical reliant deux axes de nœuds et un trait horizontal représentant la durée du lien. Ainsi dans l'exemple, il existe un lien entre les nœuds a et b dans l'intervalle [4, 6].

Dans un graphe temporel, on abordera plus souvent les questions d'évolution de communautés de nœuds ou de l'importance d'un nœud. Dans un flot de liens, on s'intéressera plus souvent au temps nécessaire pour que deux nœuds soient de nouveau en contact ou à l'importance d'un lien. De manière très manichéenne et inexacte, le formalisme de graphe temporel pousse à étudier les relations et leur évolution alors que celui de flot de liens mets plus l'accent sur les interactions et leur dynamique.

Avec ce formalisme, la majorité des travaux sont encore descriptifs car ce type d'objet n'a jamais été étudié sous cet angle. Il existe de nombreux travaux étudiant le temps séparant l'apparition de deux liens pour un nœud [MSMA08, MSCA09]. Il semblerait que ces temps inter-contacts soient très hétérogènes et que de nombreuses connexions apparaissent dans un faible intervalle de temps suivie de longues durées sans activité. On parle alors de temps inter-contact *bursty*. Les effets des temps inter-contacts sur les phénomènes de diffusions et de marches aléatoires sont très étudiés [KKP⁺11, KKBK12, SBBPS12, RB13]. Il semble cependant ne pas y avoir de conclusion définitive sur le sujet car la diffusion peut être accélérée ou ralentie par les temps inter-contacts selon la structure sous-jacente.

Il existe également quelques méthodes qui s'intéressent plus à la structure des flots de liens. Des études [KKK⁺11, KKKS13] s'intéressent à la présence de motifs. Un motif dans un graphe est un petit sous-graphe comme le triangle qui est l'un des motifs les plus étudiés. Dans les flots de liens, le temps est également pris en compte dans les motifs. Il y a donc plusieurs variantes temporelles d'un même motif dans un graphe. Prenons l'exemple d'un chemin entre quatre nœuds A, B, C et D qui est représenté dans le graphe par $\{(A, B), (B, C), (C, D)\}$. Dans un flot de liens, il existe deux variantes à ce motif soit $\{(t_1, A, B), (t_2, B, C), (t_3, C, D)\}$ soit $\{(t_1, A, B), (t_2, C, D), (t_3, B, C)\}$ avec $t_1 < t_2 < t_3$. Dans un cas, une information peut être propagée au fur et à mesure de A vers D tandis que dans l'autre ce n'est pas possible. L'étude de la fréquence d'apparition de ces motifs dans le cas temporel permet d'observer si le flot de liens a une structure particulière.

Détection de structures Il existe peu de méthodes détectant des structures décrivant l'ensemble d'un flot de liens. Rozenshtein *et al.* [RTG14] se penchent sur la détection de la zone la plus dense dans les flots de liens. Ils permettent de capturer un ensemble de

nœuds et plusieurs intervalles de temps disjoints tels que ces nœuds sur ces intervalles aient le degré moyen le plus élevé dans le graphe agrégé sur ces intervalles. Bien que la mesure utilisée ne tienne pas directement compte du temps, leur méthode permet ainsi de mettre en évidence une partie du flot de liens.

Il existe également des travaux conduits dans l'équipe [VLM16, VLM15] sur la notion de clique dans les flots de liens sans durée, c'est à dire un ensemble de (b, b, u, v) . Ils généralisent la notion de cliques à ces flots de liens : pour un delta donné, une delta-clique est un ensemble de nœuds et un intervalle de temps, tels que toutes les paires de nœuds dans cet ensemble interagissent au moins tous les delta sur cet intervalle. Dans ce cadre, ils proposent un premier algorithme permettant d'énumérer les delta-cliques. Ils ont de plus appliqué leur algorithme avec succès sur différents jeux de données.

Une méthode de type Stochastic Block Model a été proposée par Matias *et al.* [MRV15]. Elle est très proche de celle proposé par Corneli *et al.* [CLR16]. En effet, l'affiliation des nœuds est fixe et c'est l'activité d'une communauté qui change au cours du temps. La grosse différence ici est que l'activité varie de manière continue dans le temps. Ainsi l'apparition d'un lien dépend de la réalisation d'un processus de Poisson non homogène qui change selon les communautés. Cependant, leur méthode ne permet pas de considérer les changements de communauté.

Résumé

Les graphes temporels et les flots de liens ne souffrent pas d'agrégation temporelle. Ils ont donc un pouvoir expressif plus important que les formalismes présentés précédemment. Formellement, flots de liens et graphes temporels sont équivalents dans le sens où un graphe temporel peut être représenté en un flot de liens et *vice versa*. En revanche, ils diffèrent dans le point de vue considéré. Dans un graphe temporel, il existe une structure de graphe représentant le réseau à chaque instant et les métriques de graphes sont pertinentes. Dans un flot de liens, il n'existe pas de structure pertinente à un instant donné et par conséquent les métriques de graphes ne sont pas pertinentes dans ce contexte. Cette différence implique d'utiliser des mesures différentes et, en particulier, de créer de nouvelles métriques pour les flots de liens. Cela explique notamment pourquoi il n'existe pour l'instant qu'assez peu de travaux traitant de la structure des flots de liens. Cette différence de perspective entraîne également un déplacement du centre d'intérêt. Dans les graphes temporels, on aura plutôt tendance à étudier les relations et leur évolution. À l'inverse, on s'intéresse plutôt aux interactions et leur répartition dans les flots de liens. Ainsi, les deux formalismes coexistent et répondent à des besoins différents.

1.4 Bilan

Dans un graphe statique, il existe de très nombreuses méthodes capturant une structure des nœuds soit via une partition soit via une couverture. L'abondance de méthodes existantes s'explique par la diversité des définitions de communautés existantes. Les structures capturées permettent de mieux comprendre l'organisation générale du graphe mais aussi de mieux comprendre le type d'un nœud.

Avec l'émergence de nouvelles données incorporant l'information temporelle, il est pertinent d'adapter la théorie des graphes pour prendre en compte le temps. L'extension temporelle des graphes est un champ de recherche très récent et il n'y a pas à

douter que de nombreuses méthodes de détection de communautés dans ce contexte vont voir le jour. Cependant, tout les formalismes existants ne sont pas équivalents et ils limitent parfois les solutions possibles.

Il y a d'une part les formalismes se rapprochant de la théorie des graphes : séries de graphes, tenseurs 3d et graphes multicouches. Ces formalismes sont proches des graphes statiques et il est même possible d'y appliquer des méthodes statiques. En revanche, la prise en compte du temps n'est que partielle. Il y a toujours une forme agrégation temporelle et il n'est pas possible d'avoir une vision très fine de l'objet d'étude. De plus, les solutions existantes reposent sur le suivi de communautés qui ne semble pas être un problème résolu.

D'autre part, les formalismes de graphes temporels et de flots de liens capturent toute l'information temporelle. Ils ne souffrent donc pas de perte d'information. Ces deux formalismes, bien qu'équivalents, ne présupposent pas la même structure sur les données sous-jacentes. Dans un graphe temporel, les liens durent assez longtemps et par conséquent il existe une structure de graphe pertinente à chaque instant. Dans les flots de liens, les liens sont plus courts et il n'existe aucune structure de graphe à un instant donné. Par conséquent, ces deux formalismes répondent à des situations différentes. Par exemple, les études des temps inter-contacts sont très majoritairement conduites en utilisant le formalisme de flot de liens. Il semble donc plus aisés d'utiliser un flot de liens qu'un graphe temporel lorsque l'on souhaite étudier la structure des interactions.

Au cours de cette thèse, nous nous intéressons à la structure communautaire que peuvent former les liens dans le temps. Nous ne pouvons pas utiliser les formalismes utilisant une agrégation temporelle car l'identité du lien est perdue et le formalisme de graphe temporel considère des situations assez différentes de celles que nous étudions. C'est pourquoi le formalisme de flot de liens semble être le plus adapté pour étudier la structure des liens. Il n'existe que peu de travaux définissant formellement les flots de liens et traitant de leur structure. C'est pourquoi dans le chapitre 2, nous définissons plus formellement ce qu'est un flot de liens ainsi que les métriques utilisées tout au long de cette thèse avant de présenter nos travaux sur la structure des liens dans les chapitres suivants.

Chapitre 2

Flots de liens : extension temporelle des graphes

Sommaire

2.1 Définition	27
2.2 Sous-flots	28
2.3 Degré et densité	29
2.4 Liste des notations pour les flots de liens	31
2.5 Manipulation concrète des flots de liens	31

2.1 Définition

Nous avons décrit rapidement le formalisme de flot de liens dans le chapitre précédent. Nous nous attachons maintenant à définir plus formellement les flots de liens et quelques notions utilisées dans le reste de cette thèse. Un flot de liens est défini par un triplet $L = (T, V, E)$ où $T = [\alpha, \omega]$ est un intervalle de temps, V un ensemble de n noeuds 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 le lien (u, v) existe sur l'intervalle $[b, e] \subseteq [\alpha, \omega]$. Nous dénotons le nombre de liens dans le flot par $|L| = |E| = m$ et sa durée par $\bar{L} = \omega - \alpha$. De manière analogue, la durée d'un lien $l = (b, e, u, v) \in E$ est notée $\bar{l} = e - b$. On note $\beta(E) = \min_{(b,e,u,v) \in E}(b)$ et $\psi(E) = \max_{(b,e,u,v) \in E}(e)$ l'apparition du premier lien et la disparition du dernier lien dans le flot de liens.

Nous considérons les flots de liens non orientés et sans boucles, *i.e.* $(b, e, u, v) = (b, e, v, u)$ et $u \neq v$. Enfin de manière analogue aux graphes et aux multigraphes, nous définissons les flots de liens simples. Un flot de liens est simple si pour tout $l_1 = (b, e, u, v) \in E$ et $l_2 = (b', e', u, v) \in E$, $[b, e] \cap [b', e'] = \emptyset$ si $l_1 \neq l_2$. Dans les graphes, il est possible de transformer un multigraphe en graphe simple. Nous définissons également cette opération que nous nommons simplification : $\sigma(L)$. Afin de définir la simplification, nous nous aidons de la fonction de présence $\zeta_L(u, v, t)$ d'un flot de liens qui est égale à 1 si au moins un lien existe dans L entre u et v à l'instant t et 0 sinon. $L' = (T, V, E') = \sigma(L)$ est la simplification de $L = (T, V, E)$ si et seulement si L' est simple et si $\forall u, v \in V, \forall t \in T, \zeta_L(u, v, t) = \zeta_{L'}(u, v, t)$.

Il est parfois nécessaire d'augmenter la durée des liens. C'est notamment utile lorsque les liens sont de la forme (t, t, u, v) , ce qui est le cas lorsqu'on étudie des envoies de courriels par exemple. Nous notons $\xi(L, \Delta) = L'$ le fait d'augmenter de Δ la durée

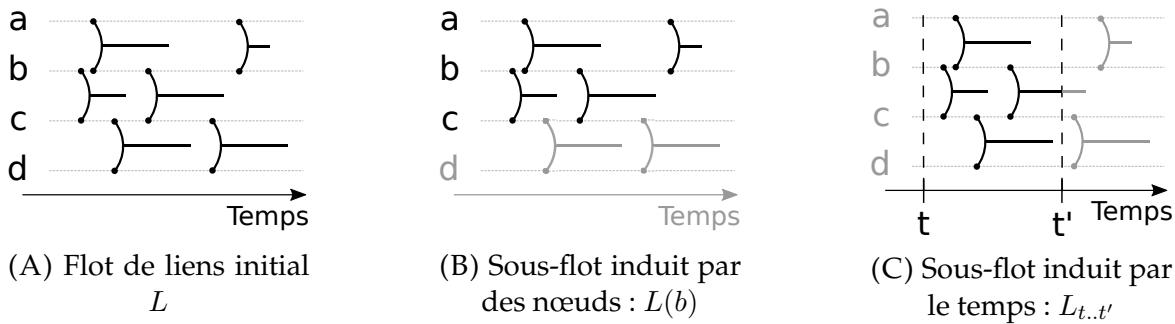


FIGURE 2.1 – Exemple de différents sous-flots, (B) et (C), du flot initial en (A). Les liens en noirs sont les liens sélectionnés dans le sous-flot.

de chaque lien. Il y a plusieurs moyens d’augmenter de Δ un intervalle $[b, e]$. Nous considérons l’ajout symétrique c’est à dire qu’un intervalle $[b, e]$ est transformé en l’intervalle $[b - \Delta/2, e + \Delta/2]$.

Enfin, il est parfois intéressant d’agréger l’information temporelle pour créer un graphe statique. Un graphe $G = (V, E') = G(L)$ est le graphe agrégé de L si et seulement si $\forall(u, v) \in E', \exists b, e \in T$ tel que $(b, e, u, v) \in E$.

2.2 Sous-flots

Dans les graphes, il existe la notion de sous graphes, voir la section 1.1, que nous étendons également aux flots de liens. Un flot de liens $L' = (T', V', E')$ est un sous-flot de L , ce que l’on note $L' \subseteq L$, si et seulement si $T' \subseteq T$, $V' \subseteq V$ et $\forall u, v \in V, \forall t \in T, \zeta_{L'}(u, v, t) \leq \zeta_L(u, v, t)$. Cette notion est en particulier utile pour définir des sous-flots induits par différents ensembles d’éléments. Nous illustrons ces notions dans les figures 2.1B et 2.1C avec le flot initial dans la figure 2.1A.

Nous définissons $L(E')$, le sous-flot de L induit par un ensemble de liens $E' \subset E$: $L(E') = ([\beta(E'), \psi(E')], V(E'), E')$ où $V(E') = \{u, \exists(b, e, u, v) \in E\}$ est l’ensemble des nœuds induits par E' .

Nous définissons $L(S)$, le sous-flot de L induit par un ensemble de paires de nœuds $S \in V^2$: $L(S) = ([\beta(E'), \psi(E')], V', E')$ avec $E' = \{(b, e, u, v) \in E, (u, v) \in S\}$ et $V' = \{u, \exists(u, v) \in S\}$. Par convention, on note $L(v) = L(\{v\} \times V)$, le sous-flot induit par un nœud. Un exemple de sous-flot induit par un nœud est dans la figure 2.1B.

Enfin, nous définissons $L_{t..t'} = ([t, t'], V, E')$, le sous-flot de L induit par l’intervalle de temps $[t, t'] \subset [\alpha, \omega]$ où $E' = \{(b', e', u, v), \exists(b, e, u, v) \in E, b' = \max(b, t), e' = \min(e, t'), [b, e] \cap [t, t'] \neq \emptyset\}$. Il est également possible de définir $L_{t..t'}$ via la fonction de présence : $\forall u, v \in V, \forall x \in [t, t'] \zeta_{L_{t..t'}}(u, v, x) = \zeta_L(u, v, x)$. Tous les liens qui sont présents durant au moins un instant de l’intervalle $[t, t']$ sont dans le sous-flot. Un exemple de sous-flot induit par un intervalle de temps est présenté dans la figure 2.1C. Il est intéressant de noter que le sous-flot $L_{t..t}$, que l’on note L_t , est équivalent au graphe statique à l’instant t , que nous notons $G(L_t)$. Ce graphe statique à un instant correspond au formalisme de graphe temporel présenté précédemment.

Enfin, il est aussi possible de combiner ces notions. Par exemple avec $V' \subset V$, $L_{t..t'}(V'^2)$ est le sous-flot correspondant aux liens entre les nœuds de V' sur l’intervalle $[t, t']$.

Un sous-flot induit par un ensemble de paires de nœuds peut être construit de manière quasiment linéaire en nombre de liens dans le sous-flot. Pour un ensemble de paires de nœuds $S = V_1 \times V_2$, il suffit d'itérer sur l'ensemble des liens qui sont reliés aux nœuds de V_1 et de vérifier que l'autre nœud du lien appartient à V_2 . ce qui se fait en $O(\log(|V_2|))$ pour chaque lien. Le parcours des liens reliés à V_1 peut être fait en $O(|L(V_1 \times V)|)$ si chaque nœud a la connaissance des liens auxquels il est relié. Si tel est le cas alors il est possible de construire le sous-flot en $O(|L(V_1 \times V)| \log(|V_2|))$. Il faut tout de fois distinguer le cas où $S = V_1 \times V$ car il n'est alors pas nécessaire de faire la vérification et la complexité est alors $O(|L(V_1 \times V)|)$.

Pour l'intervalle de temps, la situation est plus compliquée car il n'est pas facile de connaître l'ensemble des liens existant à un instant donné. Il est possible de les trier par ordre d'apparition ou de disparition mais il n'y a pas d'ordre total sur la présence des liens. Pour qu'un lien, $l = (b, e, u, v)$ appartienne au sous-flot temporel sur $[t, t']$, il doit remplir les deux conditions suivantes :

- $b \leq t'$ le lien commence avant la fin de l'intervalle $[t, t']$.
- $e \geq t$ le lien finit après le début de l'intervalle $[t, t']$.

Il n'est donc pas nécessaire de considérer les liens ayant un temps de fin inférieur à t . De manière analogue, il n'est pas nécessaire de considérer les liens ayant un temps de début supérieur t' . Cependant ces deux conditions ne peuvent pas être combinées pour restreindre le nombre liens considérés. Une manière de construire le sous-flot temporel est d'itérer sur tous les liens dans l'ordre temporel d'apparition et d'arrêter le parcours dès qu'un lien a un temps de début supérieur à t' . Il faut cependant tester l'appartenance de chaque lien parcouru. Il n'est pas possible de restreindre d'avantage le parcours car l'ensemble des liens débutant entre α et t peuvent être dans le sous-flot en respectant la dernière condition. De manière analogue, il est possible d'itérer sur les liens dans l'ordre inverse de disparition des liens et de s'arrêter dès qu'un lien a un temps de fin inférieur à t .

Dans le pire des cas, ces méthodes sont donc linéaires sur le nombre total de liens dans le flot et non dans le sous-flot comme précédemment. Il n'y a alors que deux optimisations possibles. Il est possible de choisir le sens du parcours si les deux ordres sont disponibles. Si la plus longue durée de liens, \bar{l}_{max} , est connue, alors il est possible de commencer la recherche à partir du premier lien respectant $b + \bar{l}_{max} \geq t$.

2.3 Degré et densité

Nous avons défini des outils pour manipuler et extraire des sous-flots d'un flot de liens. Nous nous intéressons maintenant à étendre quelques notions existantes dans les graphes, en particulier le degré et la densité.

Il est assez trivial de définir le degré temporel d'un nœud u de la manière suivante :

$$d(t, u) = |L_t(u)| = \sum_{v \in V} \zeta_L(u, v, t). \quad (2.1)$$

Le degré temporel n'est alors pas une valeur comme dans les graphes mais une fonction qui dépend du temps. Comme les liens apparaissent et disparaissent de manière instantanée, la fonction de degré est une fonction constante par morceaux. Un exemple de degré temporel est présenté dans la figure 2.2.

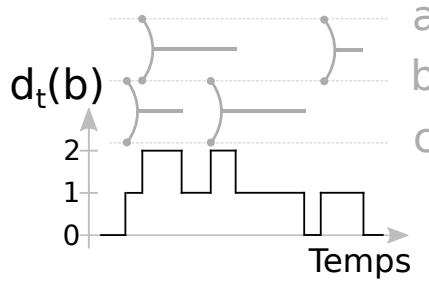


FIGURE 2.2 – Degré temporel du nœud b dans le flot de liens dans la figure 2.1A qui est également rappelé en grisé dans cette figure.

De manière analogue aux graphes, il est également possible de définir le degré temporel d'un ensemble de nœuds, $d_t(V')$, ou d'un ensemble de liens, $d_t(E')$:

$$d(t, V') = \sum_{v \in V'} d(t, v) = 2|L_t(V' \times V')| + |L_t(V' \times V \setminus V')|, \quad (2.2)$$

$$d(t, E') = 2|L_t(E')|. \quad (2.3)$$

Avec ces définitions, il est aussi possible définir les notions de degré temporel interne, $d_{in}(t, V') = 2|L_t(V' \times V')|$, et externe $d_{out}(t, V') = |L_t(V' \times V \setminus V')|$. Les degrés temporels d'un nœud, d'un ensemble de nœuds ou de liens peuvent se calculer rapidement. Il faut pour ce faire considérer les liens appartenant au sous-flot induit respectivement par un nœud, ensemble de liens ou un ensemble de paires de nœuds. Il faut d'abord transformer les liens (b, e, u, v) du sous-flot en une suite de modifications de la forme $(b, +1)$ et $(e, -1)$ ce qui a une complexité en $O(m)$. Une fois cette liste créée, il suffit d'ordonner ces modifications dans l'ordre temporel, ce qui peut être fait en temps $O(2m \log(2m))$, puis d'itérer sur l'ensemble des modifications en sommant au fur et à mesure les apparitions et disparitions de lien, ce qui peut être fait en temps $O(2m)$. Ainsi, le degré temporel à un instant t est égal à la valeur de la somme des modifications à cet instant.

Les degrés sont des fonctions du temps mais il est souvent intéressant de regarder la somme pondérée du degré, que l'on note $D_{t..t'}(u)$, et le degré moyen, que l'on note $d_{t..t'}(u)$, sur un intervalle donné :

$$D_{t..t'}(u) = \int_t^{t'} d(t, u) dt = \sum_{l \in L_{t..t'}(u)} \bar{l} \quad d_{t..t'}(u) = \frac{D_{t..t'}(u)}{t' - t}. \quad (2.4)$$

Lorsque cela n'est pas ambigu, nous notons $d_{\alpha..\omega}(u) = d(u)$. Il est intéressant de noter dans cette formulation que si tous les liens durent tout au long du flot de liens alors le degré dans le graphe agrégé et le degré moyen dans le flot de liens sont égaux, c'est-à-dire $d_{\alpha..\omega}(u) = d_{G(L)}(u), \forall u \in V$.

À partir du degré, il est possible de définir beaucoup de notions différentes et notamment la densité. Pour rappel, la densité dans un graphe est définie par $\delta(G) = 2m/(n(n - 1)) = d(V)/n(n - 1)$ et est égale à la probabilité qu'il existe un lien entre 2 nœuds. Si l'on transpose l'idée au formalisme de flot de liens, la densité dans un flot de liens est la probabilité qu'il existe un lien entre 2 nœuds à un instant donné aléatoire.

Cela se traduit par la formule suivante :

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

Il se trouve que cette formulation est complètement équivalente à la densité moyenne des graphes $G(L_t)$:

$$\begin{aligned} \frac{1}{\omega - \alpha} \int_{\alpha}^{\omega} \delta(G(L_t)) dt &= \frac{1}{\omega - \alpha} \int_{\alpha}^{\omega} \frac{d(t, V)}{n(n-1)} dt = \frac{1}{\omega - \alpha} \int_{\alpha}^{\omega} \frac{\sum_{u \in V} d(t, u)}{n(n-1)} dt \\ &= \frac{\sum_{u \in V} \int_{\alpha}^{\omega} d(t, u) dt}{n(n-1)(\omega - \alpha)} = \frac{\sum_{u \in V} \sum_{l \in L(u)} \bar{l}}{n(n-1)(\omega - \alpha)} = \frac{2 \sum_{l \in E} \bar{l}}{n(n-1)(\omega - \alpha)}. \end{aligned}$$

Pour arriver à ce résultat, nous utilisons la relation entre degré temporel moyen et somme des durées de l'équation 2.4.

La notion de densité que nous avons définie est donc cohérente avec notre notion de degré et traduit le même concept que dans les graphes. Ainsi, la densité dans les flots de liens est aussi comprise entre à 0 et 1. Enfin, cette formulation de densité est une généralisation de la densité proposée par Viard *et al.* [VL14] qui ne considérait que les liens sans durée. Les auteurs considèrent l'ajout de temps arbitraire sur chaque lien. Leur formulation est équivalente à la densité suivante : $\delta(\sigma(\xi(L, \Delta)))$ si l'augmentation de la durée ne se fait pas de manière symétrique. Un lien est (t, t, u, v) est transformé en un lien $(t - \Delta, t, u, v)$.

Le calcul de la densité d'un flot est linéaire en le nombre de liens car il est dépendant du calcul de $d_{\alpha \dots \omega}(V)$ qui est fait de manière linéaire.

2.4 Liste des notations pour les flots de liens

2.5 Manipulation concrète des flots de liens

Nous avons défini formellement quelques notions pour les flots de liens. Afin de manipuler ces notions simplement, nous avons mis en place une librairie capable de les calculer. Le but est de fournir une implémentation générale qui soit simple d'accès. Ainsi, il sera possible à n'importe qui de calculer ces notions. La démarche, bien que beaucoup plus modeste, est similaire à ce qui est fait pour les graphes avec *networkx*¹.

L'implémentation est en C++ pour la rapidité avec un export en *python* pour faciliter l'utilisation. Le code de cette implémentation est en ligne² et la documentation également³. Il est, par exemple, possible avec la librairie de lire un flot de liens et de calculer la densité moyenne d'un sous-ensemble de nœuds sur un intervalle arbitraire. Comme le but de cette librairie est d'être généraliste, l'implémentation n'est pas optimisée pour un calcul spécifique. Ainsi, il est aisément d'étendre la librairie afin de permettre le calcul de nouvelles notions.

1. <https://networkx.github.io/>

2. <https://bitbucket.org/nGaumont/liblinkstream/>

3. <https://linkstream.ngaumont.fr/>

Symbole	description
L	Flot de liens
T	intervalle de temps
V	ensemble de nœuds
E	ensemble de liens : (b, e, u, v)
$ L , E $	nombre de liens dans le flot
$\beta(E)$	temps d'apparition du premier lien
$\psi(E)$	temps de disparition du dernier lien
$\xi(L, \Delta)$	augmentation de la durée de chaque lien de Δ
$\sigma(L)$	Simplification du flot de liens L
$G(L)$	graphe agrégé de L
$L(V_1 \times V_2)$	sous-flot induit par l'ensemble de paires de nœuds $V_1 \times V_2$
$L_{t..t'}$	sous-flot induit par l'intervalle $[t, t']$
L_t	sous-flot induit par un instant t
$d_t(v)$	degré de v à l'instant t
$d_t(V)$	somme des degrés des noeuds dans V à l'instant t
$d_{in}(t, V')$	somme des degrés internes des nœuds dans V' à l'instant t
$d_{t..t'}(v)$	degré moyen de v sur $[t, t']$
$D_{t..t'}(u)$	somme pondérée du degré de v sur $[t, t']$
$d(v)$	degré moyen v sur T
$\delta(L)$	densité du flot

TABLE 2.1 – Liste des notations pour les flots de liens

Un intérêt de cette implémentation est de permettre, en python, la génération de visualisation⁴, voir le dessin dans la figure 2.3. Parmi les possibilités offertes par la visualisation, il est possible de n'afficher qu'une partie des nœuds ou de ne garder qu'un sous intervalle de temps. De plus, il est possible de choisir la couleur des liens de manière manuelle ou en fonction d'une partition de liens.

La lisibilité de ce genre de visualisation est très dépendante de l'ordre attribué aux nœuds sur l'axes des ordonnées. C'est pourquoi avec notre outil, il est également possible de fixer un ordre arbitraire. Comme il peut être fastidieux d'écrire un ordre, nous avons implémenté un algorithme rudimentaire pour améliorer l'ordonnancement

4. Pour l'instant, uniquement un export en svg est possible.

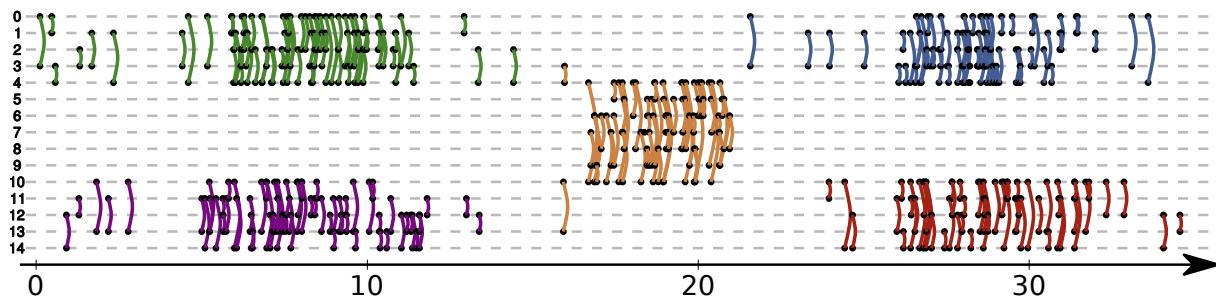


FIGURE 2.3 – Exemple d'une visualisation d'un flot de liens sans durées entre 15 nœuds au cours du temps qui a été généré par notre librairie. La couleur des liens a été fixée par l'utilisateur de la librairie.

des nœuds dans la visualisation. Afin de pouvoir améliorer une visualisation, il est nécessaire de pouvoir quantifier la complexité de la visualisation actuelle. Empiriquement, on se rend vite compte que ce sont les longs traits verticaux qui rendent la visualisation complexe. Il faut donc les limiter.

C'est pourquoi nous décidons d'évaluer un ordonnancement en fonction de la somme des longueurs des traits représentant les liens. Avec la fonction d'ordre $Ordre : V \mapsto \mathbb{N}$, trouver le meilleur ordonnancement se résume à résoudre :

$$Ordre^* = \arg \min_{Ordre} \sum_{(b,e,u,v) \in E} |Ordre(u) - Ordre(v)| \quad (2.6)$$

Malheureusement, trouver l'optimum ne semble pas trivial et il n'est pas envisageable de tester l'ensemble des ordres. C'est pourquoi, nous utilisons un algorithme probabiliste qui teste des permutations aléatoires. Une permutation est appliquée si elle améliore l'évaluation de la visualisation. Il s'agit bien sûr d'une première approche qu'il est possible d'améliorer.

Une piste possible est la construction d'un ordre potentiellement proche de l'optimum. Par exemple, il serait intéressant de trier chaque paire de nœuds (u, v) en fonction du nombre de liens existants entre u et v puis de construire itérativement un ordre des nœuds en fonction de l'ordre des paires de nœuds.

Chapitre 3

Étude de la structure d'une archive de courriels en tant que flot de liens

Sommaire

3.1	Prétraitement sur le jeu de données	36
3.2	Caractéristiques élémentaires des discussions	37
3.3	Étude des discussions en tant que sous-flots	39
3.3.1	Application de la Δ -densité	39
3.3.2	Répartition temporelle et structurelle des discussions	42
3.3.3	Flot quotient	43
3.4	Détection de structures denses	45
3.4.1	Méthode de détection	45
3.4.2	Comparaison des partitions	46
3.4.3	Conclusion et perspectives	49

Nous nous intéressons ici à une archive de courriels publiquement disponibles¹. Cette archive contient l'ensemble des courriels échangés par différent 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 noeud 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 tout 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 liens représentant les courriels envoyés sur la liste.

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

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

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 724 985 courriels que contenait l'archive en janvier 2015. Tout d'abords, 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 flots de liens, un script d'extraction a été développé². Lors de l'extraction, 2 269 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 posté³, 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 simple auxquelles 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 dépassent 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 discussion trop récente.

2. <https://bitbucket.org/nGaumont/mailarchiver>

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

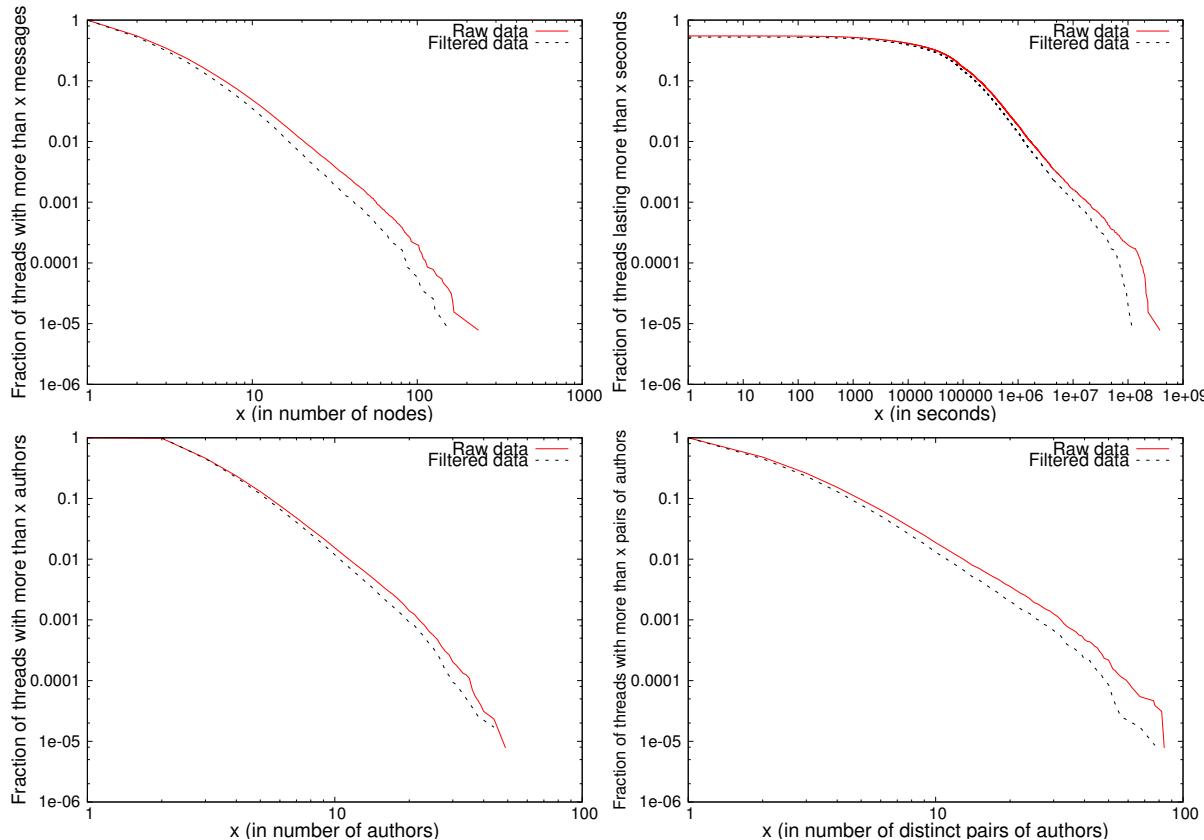


FIGURE 3.1 – Distribution cumulative inverse de différentes caractéristiques pour les données brutes (ligne pleine) et filtrées (ligne pointillée). En haut à gauche : nombre de courriels dans une discussion ; en haut à droite : durée d'une discussion ; en bas à gauche : nombre de personnes dans une discussion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

3.2 Caractéristiques élémentaires des discussions

Les caractéristiques les plus élémentaires des discussions sont le nombre de courriels, le nombre 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

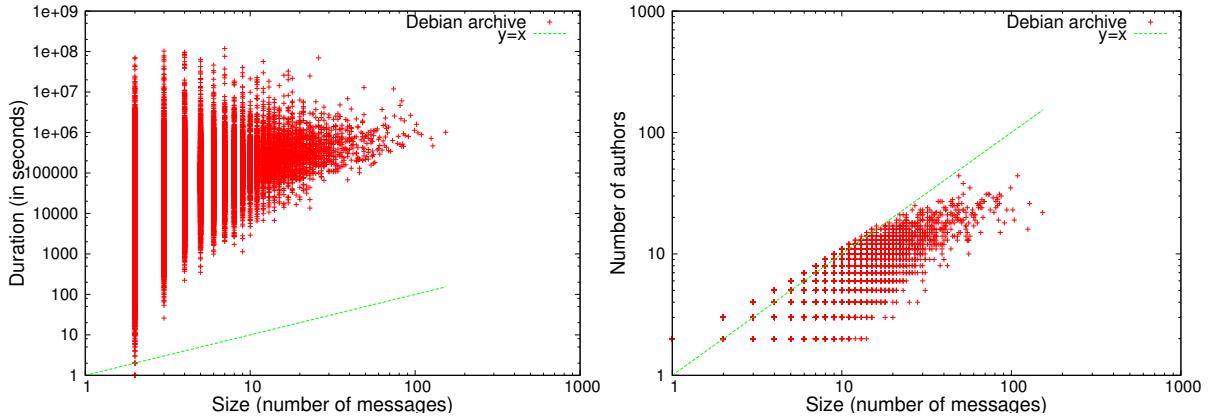


FIGURE 3.2 – Gauche : Corrélations entre le nombre de courriels et la durée d'une discussion. Droite : Corrélation entre le nombre de courriels et le nombre d'auteurs dans une discussion.

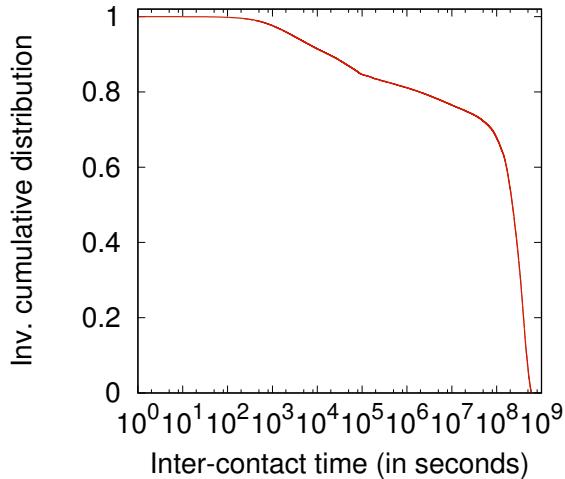


FIGURE 3.3 – Distribution des temps inter-contacts dans le fil de discussions.

la corrélation entre le nombre de courriels et d'auteurs, on observe un autre fait attendu [DLCA07] qui est qu'une discussion est 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.

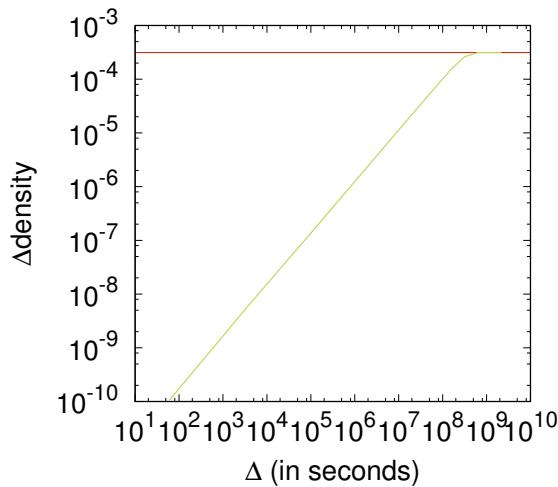


FIGURE 3.4 – Évolution de la Δ -densité (en vert) du flot de liens pour Δ de 60 seconde à 20 ans. En rouge, la densité dans le graphe agrégé.

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 densité non pas dans le flot de liens initial mais dans $\sigma(\xi(L, \Delta))$ pour différentes valeurs de Δ entre 1 seconde et 20 ans. La notation $\delta_{\alpha.. \omega}(\sigma(\xi(L, \Delta)))$ est cependant très lourde et nous la simplifions par $\delta_\Delta(L)$ et nous parlons donc de Δ -densité. 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, c'est-à-dire $\delta(\sigma(\xi(\Delta, D_i)))$. 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 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 10⁵ 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.

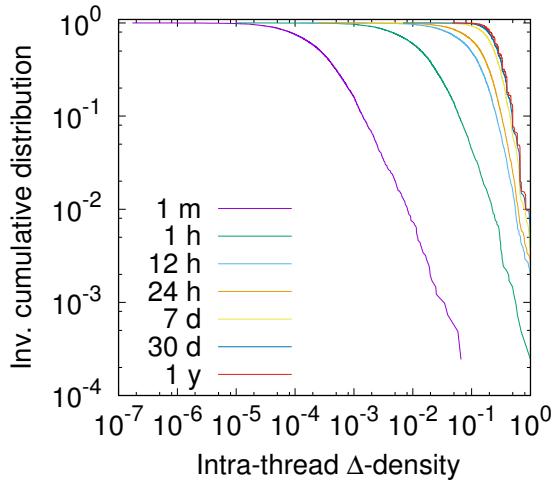


FIGURE 3.5 – Distribution cumulative inverse de la Δ -densité des discussions pour différentes valeurs de Δ .

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 soit, 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 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, $\delta^{inter}(C_i)$:

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

Il s'agit 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 $\Delta_{ij} = \xi(\Delta, D_i \cup D_j)$ et $L_{inter}(D_i, D_j) = (T', V', E')$ avec $V' = V(\Delta_{ij})$, $T' = [\beta(\Delta_{ij}), \psi(\Delta_{ij})]$, et $E' = \xi(\Delta, E) \setminus \Delta_{ij}$. Avec ces notation, $\delta_\Delta^{inter}(D_i, D_j)$ 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. Il est important de noter que le nombre de liens pris en compte dans le flot inter-discussions varie selon le Δ utilisé.

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 :

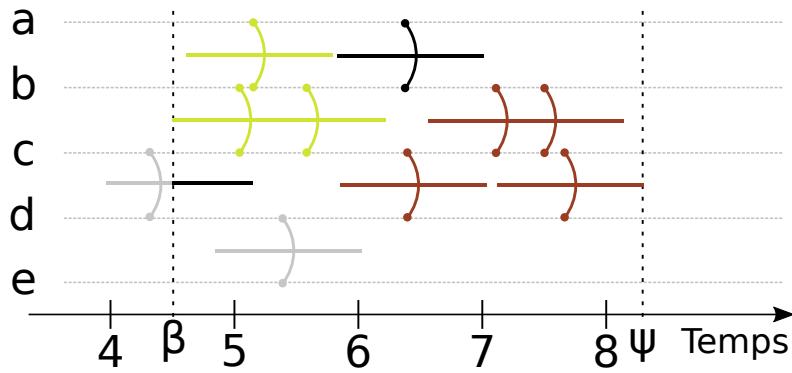


FIGURE 3.6 – Le flot entre les discussions verte et rouge est constitué des liens ou partie de liens en noir. Les liens en gris ne sont pas pris en compte.

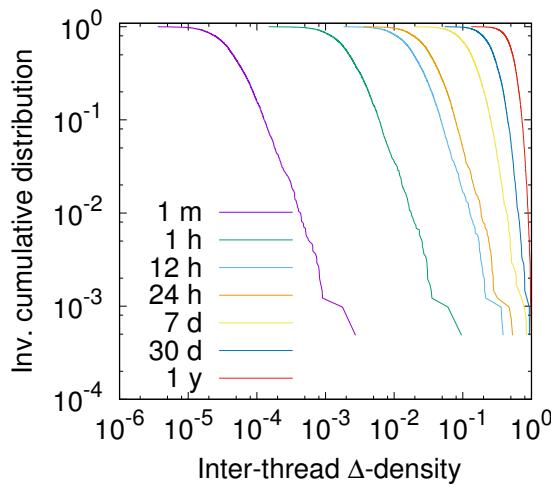


FIGURE 3.7 – Distribution cumulative inverse de la Δ -densité inter discussions pour différentes valeurs de Δ s.

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

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 peu 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 peu 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 durée 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

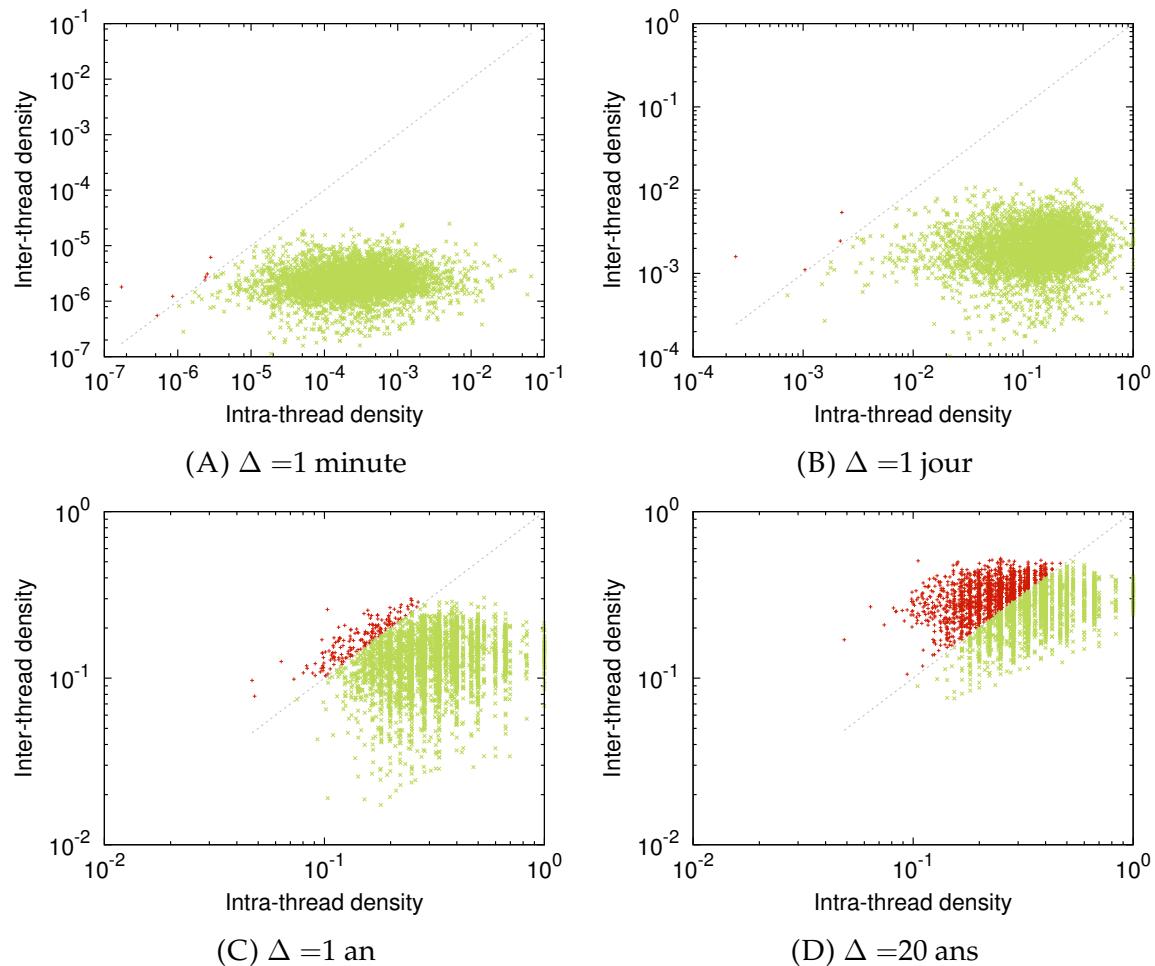


FIGURE 3.8 – Corrélations entre Δ -densité et Δ -densité inter discussions pour différentes valeurs de Δ . Une discussion est en vert (resp. rouge) si elle a une Δ -densité plus (resp. moins) élevée que sa Δ -densité inter discussions.

Δ . On remarque que les discussions sont effectivement plus denses intérieurement qu'avec les autres 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'ancre 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,

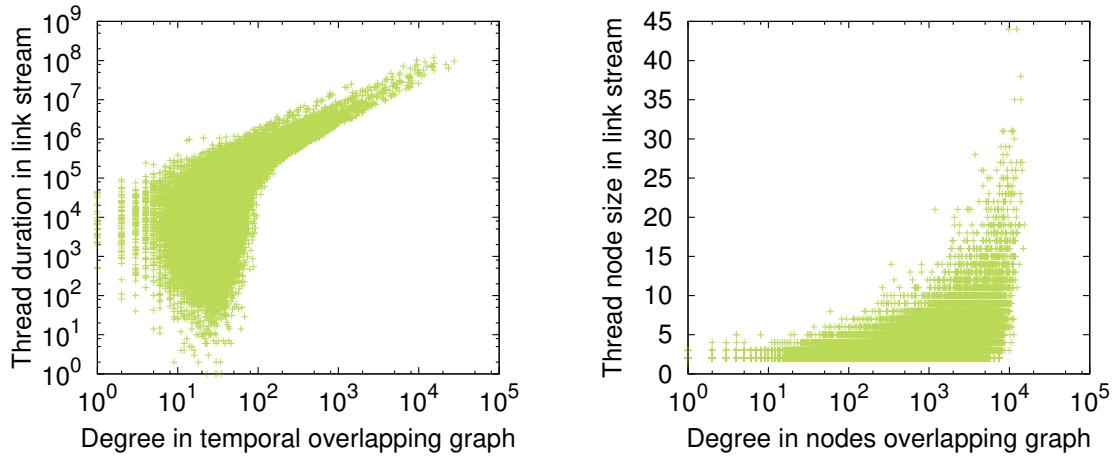


FIGURE 3.9 – Gauche : Corrélation entre le degré des discussions dans le graphe de chevauchement temporel et leur durée. Droite : Corrélation entre le degré des discussions dans le graphe de chevauchement topologique et leur nombre de participants.

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 supérieure 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 discussion peut partager des nœuds avec énormément d'autres discussions.

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 s'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} égal au nombre de liens

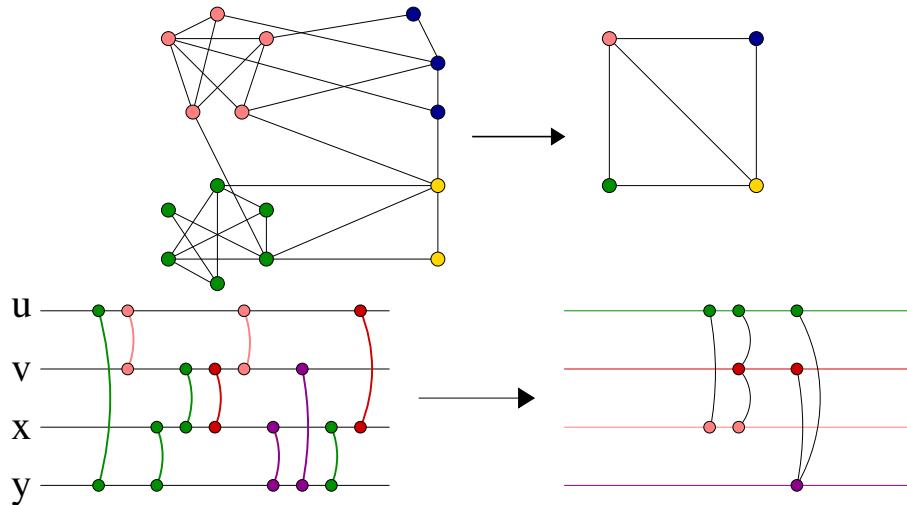


FIGURE 3.10 – Haut : Exemple de graphe ayant une structure communautaire et son graphe quotient associé. Bas : Exemple d'un flot de lien avec une structure ainsi que son flot quotient associé.

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

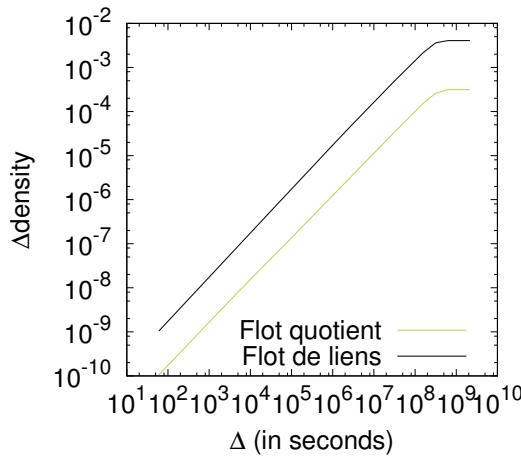


FIGURE 3.11 – Δ -densité du flot de liens et du flot de liens quotient en fonction de Δ pour $\Delta = 1mn, 1h, 12h, 1j, 7j, 30j, 1\text{ an}$ et 20 ans .

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, \mathfrak{L} , dont les liens ont une durée. À partir de \mathfrak{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 $(b', e', u', v') \in \mathfrak{L}$ 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 la 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 \mathfrak{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

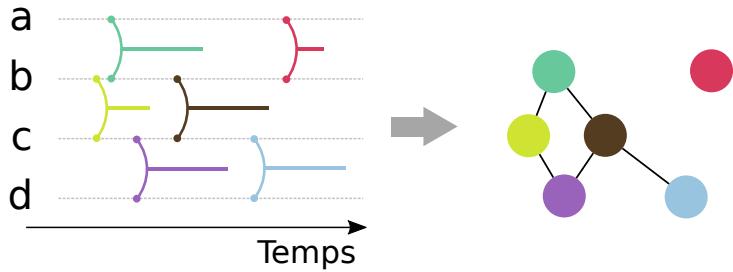


FIGURE 3.12 – Transformation d'un flot de liens avec 4 noeuds (a-d) et 6 liens à gauche en un graphe à droite à 6 noeuds. La couleur d'un noeud dans le graphe indique le lien du flot qu'il représente.

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 :

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

Par construction, une discussion est donc représentée dans \mathcal{G} par un ensemble connexe de noeuds. 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 noeuds 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 noeuds 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 noeuds 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 noeuds et de liens alors qu'ils ont des durées similaires.

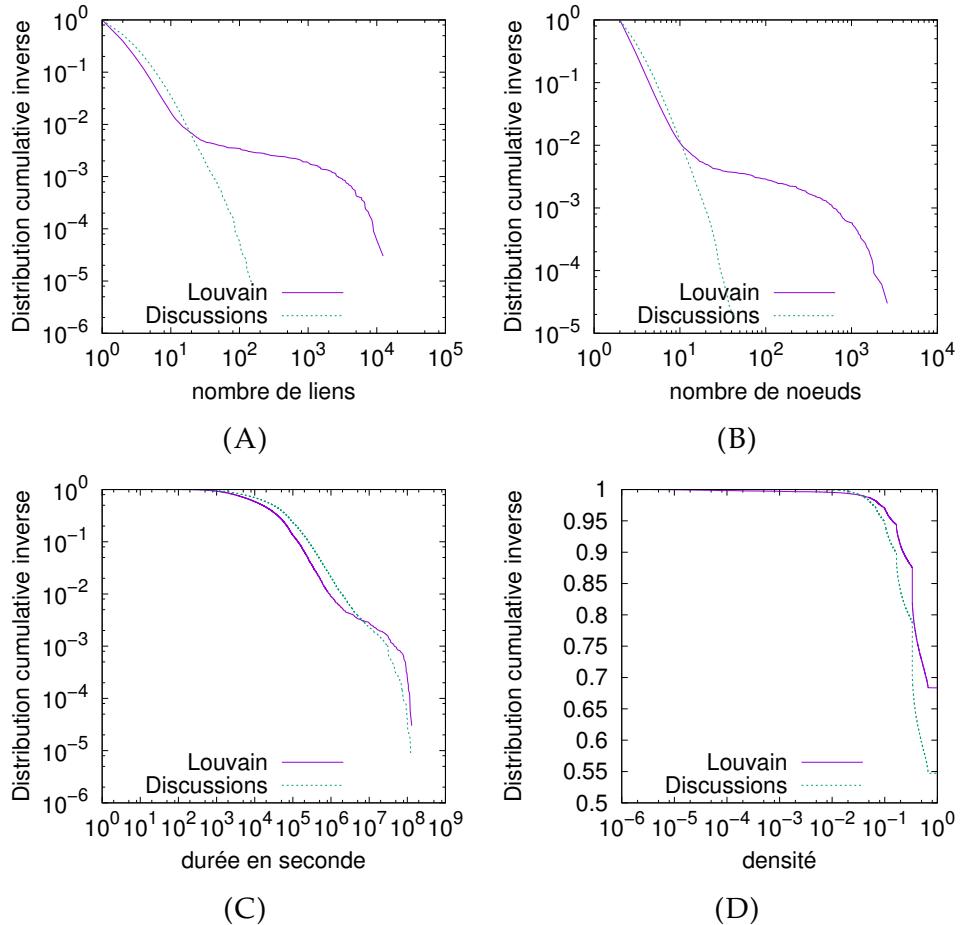


FIGURE 3.13 – Distribution cumulative inverses du nombre de liens (a), du nombre de nœuds (b), de la durée (c) et de la densité (d) pour les groupes trouvés par Louvain et les discussions.

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 égal (entre 2 et 160), on remarque que les groupes trouvés par Louvain sont plus denses en moyenne ,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 :

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

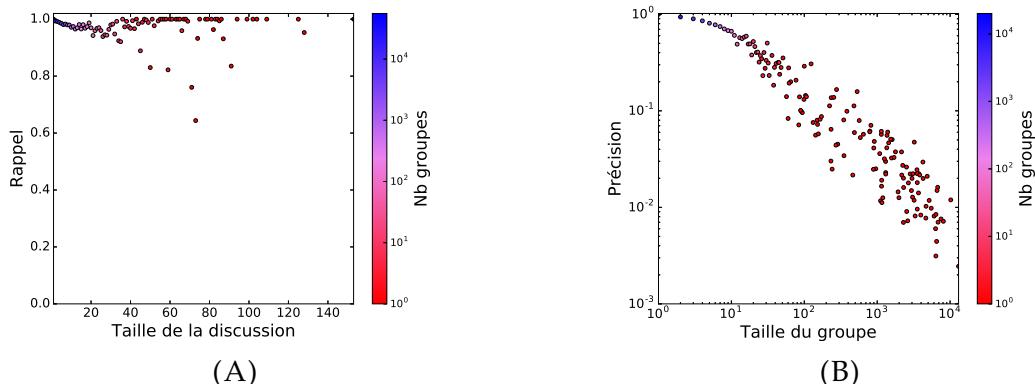


FIGURE 3.14 – (A) Rappel des discussions vis-à-vis des groupes trouvés par Louvain. (B) Précision des groupes trouvés vis-à-vis des discussions. Chaque point représente la moyenne du rappel (resp. précision) pour les discussions (resp. groupes) ayant la même taille. La couleur du point indique le nombre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

Dans la figure 3.14, sont présentées la *précision* des groupes et le *rappel* des discussions en fonctions de leur taille. Chaque point représente la moyenne de la *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 général 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 chaque groupe \mathfrak{D}_j . Le niveau 0 est la première partition trouvée par l'algorithme de Louvain dans \mathfrak{G} , c'est-à-dire $\mathfrak{D}_{0,j} = \mathfrak{D}_j$. Soit $\mathfrak{D}_{h,j'}$ un groupe trouvé au niveau h , par construction, il est inclus dans un groupe trouvé au niveau $h - 1$, c'est à dire $\mathfrak{D}_{h,j'} \subseteq \mathfrak{D}_{h-1,j}$.

Soit $D_i \in D$, notons $\mathfrak{D}_{h,\tilde{i}}$ 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 $|\mathfrak{D}_{h,\tilde{i}} \cap D_i| = \max_j |\mathfrak{D}_{h,j} \cap D_i|$. Avec ces définitions, on observe la relation suivante : $\mathfrak{D}_{h,\tilde{i}} \cap D_i \subseteq \mathfrak{D}_{h-1,\tilde{i}} \cap D_i$. 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 :

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

Ainsi, le *rappel* au niveau h prend 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

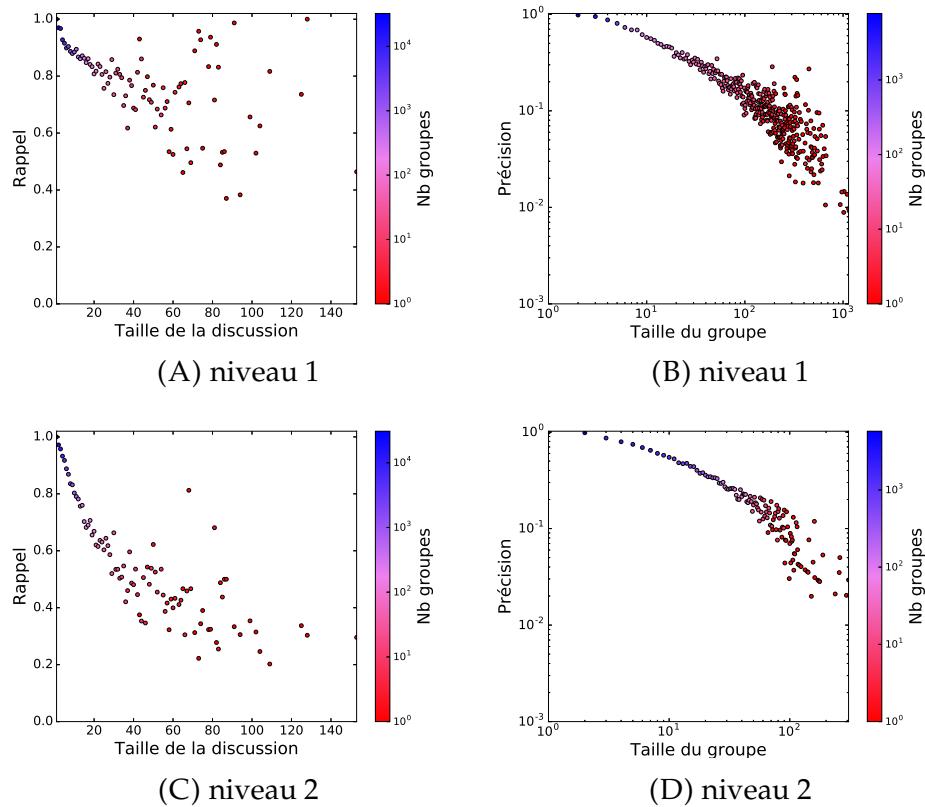


FIGURE 3.15 – (A,B,C) Rappel des discussions vis-à-vis des groupes trouvés par Louvain à différent niveaux récursifs. (B,D,F) Précision des groupes trouvés vis-à-vis des discussions. Chaque point représente la moyenne du rappel (resp. précision) pour les discussions (resp. groupes) ayant la même taille.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

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 et perspectives

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 nœuds, rendant ainsi l'utilisation de partitions statiques de nœuds difficilement envisageable pour décrire les discussions.

Par la suite, nous avons mis en place un première méthode de détection de partition de liens. Cette méthode se base sur la projection du flot de liens en un graphe statique sur lequel nous appliquons un algorithme de détection de communautés. 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. Il semble donc qu'il existe une autre structure que celle des discussions.

Cependant, ces observations ne remettent pas en cause les conclusions faites sur les discussions 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 mais cela change l'objet d'étude. 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 ait du sens au vu des valeurs de densité des groupes.

Perspectives

Ce jeu de données de courriels est intéressant. D'une part, il couvre une très longue période de temps et, d'autre part, il existe de nombreuses informations associées. Nous avons utilisé ici la connaissance de la discussion associée à chaque courriel pour étudier la notion de communautés. D'autres notions pourraient être développées en s'appuyant sur ce jeu de donnée, notamment les notions de centralité et de rôle des nœuds. Grâce à la durée du jeu de données, il serait intéressant de mesurer l'évolution de l'implication et du rôle d'une personne. Il serait ainsi possible de mieux comprendre la participation de personnes à un projet cet ampleur.

Il y a également le texte brut du mail qui est accessible et dont nous n'avons pas tiré parti. Il serait sûrement intéressant pour des linguistes de quantifier l'évolution du vocabulaire. De même, nous nous sommes focalisé sur la liste de discussion anglaise, alors qu'il est possible avec le script d'extraction de considérer les listes de discussions dans d'autres langues. Il pourrait être intéressant de comparer les structures de ces deux sources de données.

Dans le cadre de la manipulation de flot de liens, une perspective importante est la compréhension plus en détail de la structure de liens que nous avons capturée. C'est cette perspective que nous poursuivons dans le chapitre suivant.

Chapitre 4

Détection de groupes pertinents dans les flots de liens

Sommaire

4.1 Travaux existants	54
4.2 Détection de groupes pertinents	55
4.2.1 Définition des voisinsages	56
4.2.2 Définition de l'évaluation	58
4.2.3 Algorithme de calcul des scores	59
4.3 Jeux de données	61
4.4 Application	62
4.4.1 Évaluation des groupes candidats	63
4.4.2 Caractéristiques des groupes pertinents	68
4.5 Conclusion et perspectives	70
4.5.1 Conclusion	70
4.5.2 Perspectives	71

Nous avons vu précédemment qu'il est possible de trouver des groupes de liens dans un flot de liens via l'intermédiaire d'une projection du flot de liens en un graphe statique. Pour ce faire, nous avons appliqué l'algorithme de Louvain sur la projection pour obtenir une partition des liens en communautés. Parmi l'ensemble des communautés, il se peut que certaines capturent des structures plus importantes et plus pertinentes que d'autres. Le but ici est de réussir à extraire parmi ces groupes ceux qui sont pertinents. En ce sens, nous nous posons la question de la décomposition partielle d'un flot de liens en sous-flots pertinents. Il s'agit donc d'un problème légèrement différent de la détection de partition car d'une part, certains liens peuvent n'appartenir à aucun groupe et, d'autre part, les groupes doivent être pertinents.

Toute la difficulté pour réussir à trouver les groupes pertinents est de trouver une définition de la pertinence. La notion de densité, que nous avons utilisée précédemment, est une première approche mais ce n'est pas suffisant. En effet, il peut exister de grands groupes pertinents mais peu denses et de petits groupes très denses mais peu pertinents. Il n'est pas possible de faire la différence entre ces deux catégories en utilisant juste la densité. C'est pourquoi, nous utilisons une notion de voisinage. Un groupe est alors pertinent s'il est plus dense que son voisinage. Ainsi, un groupe peu dense peut être pertinent s'il est plus dense que son voisinage. Comme le but est de pouvoir décrire le flot de liens, nous nous limitons aux groupes ayant une taille supérieure à un minimum donné. Un exemple de flot avec une décomposition en groupes pertinents est présenté dans la figure 4.1.

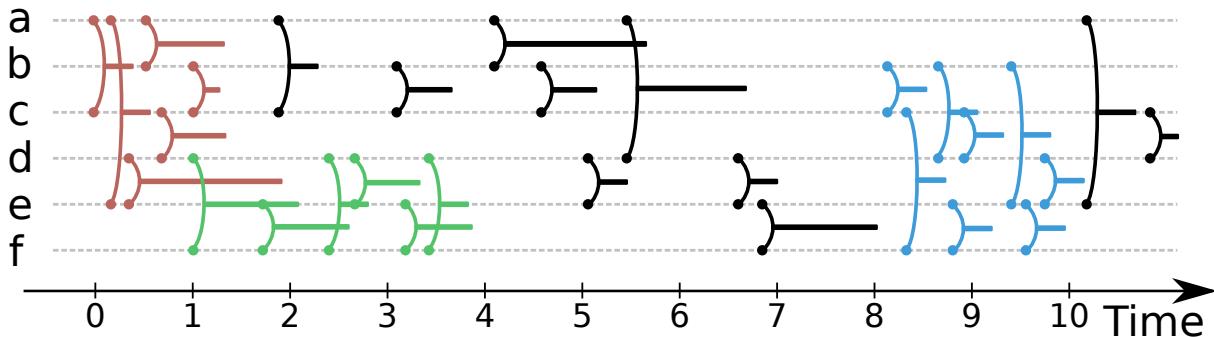


FIGURE 4.1 – Exemple de flots de liens avec 3 groupes denses représenté par la couleur des liens (rouge, vert et bleu).

Afin de trouver des groupes pertinents, nous procédons de la manière suivante :

- nous construisons la projection du flot de liens en un graphe statique ;
- nous appliquons sur cette projection un algorithme de détection de communautés afin d'obtenir une partition des liens du flot de liens ;
- nous ne gardons dans la partition que les groupes qui sont considérés comme pertinents, c'est-à-dire ceux qui sont assez gros et qui sont plus denses que leur voisinage.

Afin de montrer la pertinence de cette approche, nous l'appliquons sur différents jeux de données et faisons une analyse manuelle de certains groupes trouvés. Par ailleurs, nous montrons que les groupes que nous détectons ne le sont pas par une méthode statique.

Le chapitre est organisé de la manière suivante. Dans la section 4.1, nous revenons sur les travaux existants qui traitent de sujets similaires. Puis dans la section 4.2, nous présentons notre méthode de détection et de validation de groupes pertinents. Enfin, les jeux de données que nous utilisons sont présentés dans la section 4.4 et les résultats associés dans la section 4.3.

4.1 Travaux existants

Comme nous l'avons vu dans le chapitre 1, il existe assez peu de méthodes cherchant des structures dans un formalisme sans perte d'information.

Mitra *et al.* [MTR12] et Speidel *et al.* [STM15] ont développé une méthode de détection de communautés mais dans les réseaux diachroniques. Dans un tel réseau, ce sont les noeuds qui ont un *timestamp* et non les liens. Ce cas de figure s'applique parfaitement aux réseaux de citations car une citation relie bien deux publications apparues à deux dates spécifiques. Cependant ce genre de situation est particulier et ne peut pas être trivialement représenté par le formalisme de flot de liens. C'est pourquoi nous ne pouvons utiliser la même méthode.

Il existe également des méthodes capturant la partie la plus dense d'un réseau. C'est le cas de Bogdanov *et al.* [BMS11] qui considèrent un graphe dont la topologie n'évolue pas mais dont les poids des liens changent entre -1 et 1 en fonction du temps. Ce formalisme est encore une fois très différent de celui de flot de liens. De plus, la notion de densité utilisée ne tient pas vraiment compte du temps car un groupe de noeuds sur un intervalle est évalué par la somme des liens entre ces noeuds sur tout l'intervalle.

Epasto *et. al* [ELS15] capturent le sous-graphe le plus dense dans un graphe temporel. Ainsi, le temps est bien pris en compte dans le formalisme. Cependant, la méthode capture le sous-graphe à un instant t tel qu'il ait le degré moyen le plus élevé parmi l'ensemble des sous-graphes sur tous les instants. La notion de densité est donc statique car elle ne considère que la structure de graphe à un instant.

Enfin les travaux de Rozenshtein *et. al* [RTG14] sont ceux qui se rapprochent le plus de notre méthode. Leur méthode utilise le formalisme de flots de liens et ne suppose pas de structure de graphe à un instant donné. Elle capture un groupe de nœuds et plusieurs intervalles disjoints de telle sorte que ce groupe de nœuds dans le graphe agrégé sur ces intervalles ait le degré moyen le plus élevé. Plus formellement, un groupe de nœuds $V' \subset V$ et un ensemble d'intervalles $\{I_1, \dots, I_k\} = \mathcal{I} \subset T$ doivent maximiser $\tilde{d}_{\mathcal{G}}(V')$ où $\mathcal{G} = \bigcup G(L_t), \forall t \in \mathcal{I}$. Ainsi même si la méthode permet de capturer un groupe de nœuds sur des intervalles de temps, elle évalue un candidat sur un graphe agrégé pour appliquer une métrique de graphe. Le temps n'a donc une influence qu'indirecte sur l'évaluation. Par ailleurs, leur méthode repose sur plusieurs paramètres qui doivent être fixés *a priori* : le nombre maximum d'intervalles disjoints et la durée cumulée maximum de ces intervalles. Le premier paramètre permet d'éviter d'utiliser une infinité d'intervalles qui seraient chacun centré sur un lien. Le second paramètre permet d'éviter de considérer tout le graphe agrégé et ainsi d'imposer la prise en compte du temps. Ainsi, leur méthode dépend de manière non triviale du choix des paramètres.

Pour résumer, il existe des méthodes cherchant la structure la plus dense dans un contexte dynamique ; soit en considérant un graphe temporel soit en considérant un flot de liens. Cependant, ces méthodes utilisent des métriques de graphes pour évaluer un groupe de nœuds. Le temps n'est donc pas complètement pris en compte dans l'évaluation. De manière plus importante, ces méthodes évaluent de manière absolue un groupe sans prendre en compte son voisinage. Enfin, nous capturons plusieurs groupes de liens alors que les autres méthodes capturent un groupe nœuds. Certaines de ces méthodes peuvent être étendues pour capturer les k groupes de nœuds les plus denses mais cela se fait au prix de l'ajout d'un paramètre pour contrôler le chevauchement entre deux groupes.

4.2 Détection de groupes pertinents

Nous avons défini dans le chapitre précédent 3.4.1 une projection du flot de liens en un graphe statique. Dans cette projection, chaque lien du flot est représenté par un nœud. Deux liens (b, e, u, v) et $(b', e', u', v') \in L$ sont reliés 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 la figure 3.12. Sur cette projection, nous appliquons l'algorithme de Louvain pour la détection de communautés. Les communautés trouvées par l'algorithme sont des groupes potentiellement pertinents et nous les appelons donc groupes candidats.

Tous les groupes candidats ne sont pas forcément pertinents. Tout d'abord, les groupes d'une partition peuvent avoir des tailles très variées. En particulier, il peut y avoir beaucoup de très petits groupes de 1 ou 2 liens. Ces groupes ne sont pas pris en compte car nous fixons une taille minimum des groupes fixée *a priori*.

Par ailleurs, l'algorithme de Louvain optimise de manière gloutonne la modularité dans la projection du flot de liens en un graphe statique et non une métrique de flot de liens directement. C'est pourquoi, certains candidats peuvent être une bonne communauté dans le graphe mais un groupe non pertinent dans le flot de liens. Cet phénomène est d'autant plus important que la modularité ne prend pas en compte le voisinage d'un groupe. Par exemple dans la figure 4.1, les liens dans l'intervalle [4, 8] forment une composante connexe dans la projection et ils sont considérés comme une communauté par l'algorithme de Louvain. Or, ces liens ne sont pas un groupe pertinent car ils sont beaucoup moins denses que les liens durant l'intervalle [0, 4] ou l'intervalle [7, 11]. C'est pourquoi il est nécessaire de filtrer les groupes candidats pour ne garder que les groupes pertinents.

4.2.1 Définition des voisinages

Dans le chapitre 2.3, nous avons défini la densité d'un flot de liens. Pour un groupe de liens, il faut donc calculer la densité du sous-flot induit par ses liens. Seulement, la densité en elle même n'est pas un bon critère pour évaluer la pertinence d'un candidat. En effet, un candidat constitué d'un unique lien a une densité maximale de 1. C'est pourquoi il est également nécessaire de tenir compte du voisinage. Ainsi, plus le candidat a une densité plus importante que son voisinage, plus le candidat est un groupe pertinent.

Il est donc nécessaire de définir une notion de voisinage qui puisse tenir compte à la fois de la structure et du temps. Pour ce faire, il faut utiliser une autre formulation de la densité pour comprendre quels sont les paramètres qui influent sur la densité. Jusqu'à maintenant pour un candidat C_i , nous considérons la densité du sous-flot induit par ses liens, $\delta(L(C_i))$. Pour faire apparaître l'influence de la structure et du temps, nous utilisons $\delta(L_{\beta(C_i)..\psi(C_i)}(V(C_i)^2))$ que nous notons, par abus de notation, $\delta(V(C_i), \beta(C_i), \bar{C}_i)$ où $\bar{C}_i = \psi(C_i) - \beta(C_i)$. Il s'agit de la densité des nœuds induits par C_i sur la durée du groupe. Ces deux quantités ne sont pas complètement équivalentes car les deux sous-flots sont différents, en particulier $L(C_i) \subseteq L_{\beta(C_i)..\psi(C_i)}(V(C_i)^2)$. De cette inclusion découle le fait $\delta(L(C_i)) \leq \delta(V(C_i), \beta(C_i), \bar{C}_i)$. C'est le cas dans l'exemple de la figure 4.1 si l'on calcule la densité des liens noirs. Dans ce cas, tous les nœuds sont induits par ces liens et l'intervalle de temps est égal à [2, 11]. Le sous-flot induit par V sur [2, 11] comprend l'ensemble des liens noirs mais aussi des liens **verts** et **bleus**.

Cependant avec cette formulation, on fait apparaître 3 dimensions : l'ensemble de nœuds, le temps de début et la durée. Il devient alors aisément de considérer comme voisins les sous-flots qui diffèrent sur une seule dimension : soit le temps de début, soit la durée soit l'ensemble de nœuds. Cette différence entre les voisinages est illustrée dans la figure 4.2.

Pour le voisinage aux nœuds, il est impossible de considérer l'ensemble de tous les ensembles de nœuds car il est beaucoup trop grand. De plus, il est probable que ces groupes de nœuds ne soient même pas connexes dans le graphe agrégé si le flot de liens est peu dense. C'est pourquoi, si $V(C_i)$ est constitué de k nœuds, nous limitons à l'ensemble des groupes de k nœuds qui partagent $k - 1$ nœuds avec $V(C_i)$. De cette manière, uniquement des groupes de nœuds similaires au candidat sont considérés. Cette comparaison nous semble plus stricte mais plus équitable que celles utilisant l'ensemble des groupes de nœuds ou des groupes de nœuds pris aléatoirement. Il serait

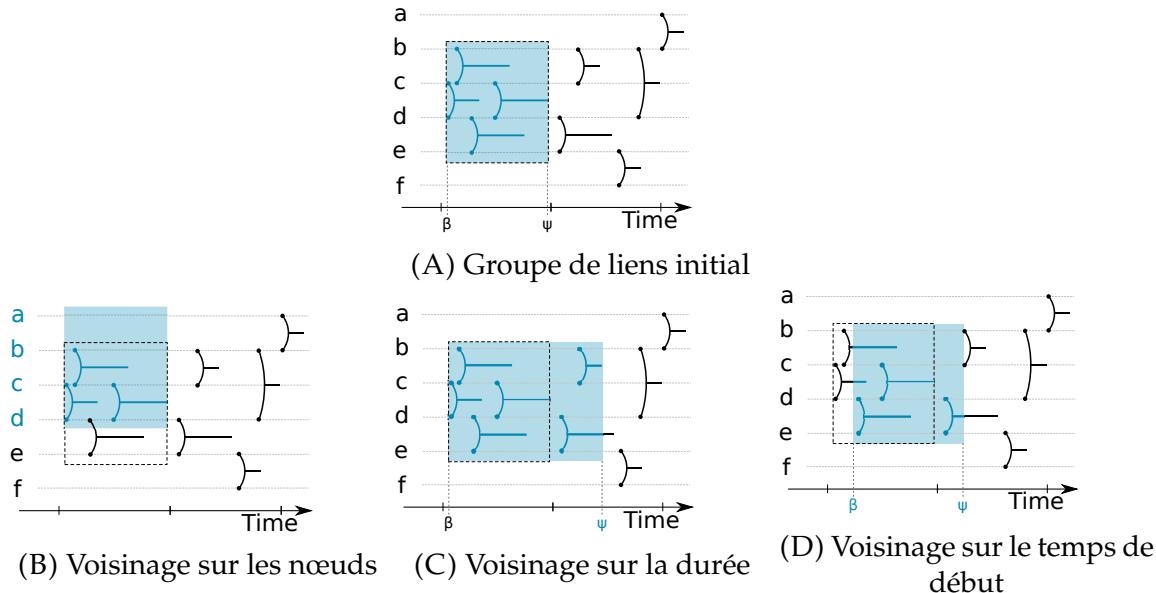


FIGURE 4.2 – Exemple d'un groupe de liens en (A) et de ses différents voisinages en (B), (C) et (D). La zone en bleu est la zone prise en compte lors du calcul de la densité. La zone en pointillé représente la zone considérée lors du calcul de la densité du groupe initial.

possible de considérer un nombre moins important de nœuds partagés mais nous ne l'avons pas fait à cause de la complexité de calcul. En effet, il y a déjà $k(|V| - k)$ sous-flots partageant $k - 1$ nœuds avec un candidat.

Pour le voisinage au temps de début (resp. à la durée), nous considérons toutes les valeurs possibles de temps de début (resp. durée) dans un intervalle donné. Chacune de ces valeurs donne lieu à un sous-fLOT induit ayant les mêmes nœuds et la même durée (resp. temps de début) que le sous-fLOT induit par les liens du groupe candidat. Ensuite, nous calculons la densité de chacun de ces sous-fLOTS voisins. Puis formellement, l'ensemble des densités des sous-fLOTS voisins se définit de la manière suivante. Soit $I_{\text{début}}$ et $I_{\text{durée}}$ deux intervalles représentant respectivement les valeurs de temps de début et de durée que nous allons considérer et C_i un groupe de liens. Les densités des sous-fLOTS dans le voisinage au temps de début sont alors définies par la fonction $\delta(V(C_i), x, \bar{C}_i)$, $\forall x \in I_{\text{début}}$. Les densités des sous-fLOTS dans le voisinage de la durée sont alors définies par la fonction $\delta(V(C_i), \beta(C_i), y)$, $\forall y \in I_{\text{durée}}$.

L'intervalle utilisé pour le voisinage au temps de début est $I_{\text{début}} = [\alpha, \omega - \bar{C}_i]$. Pour le voisinage à la durée, nous utilisons l'intervalle $I_{\text{durée}} = [0.8\Delta_{\min}, 1.2\Delta_{\max}]$, où Δ_{\min} (resp. Δ_{\max}) est la plus petite (resp. grande) durée de tous les candidats de plus de 10 liens. Nous utilisons cet intervalle pour deux raisons. Tout d'abord, il contient l'ensemble des durées acceptables quand nous l'appliquons à nos jeux de données. Mais surtout, nous avons également testé l'intervalle $[1, \omega - \alpha]$ qui est beaucoup plus grand. Cela change les résultat de manière quantitative mais pas de manière qualitative. En effet lorsque la durée considérée est proche de $\omega - \alpha$ alors la densité est très proche de 0. Comme ces valeurs sont peu informatives, nous préférions nous limiter à un intervalle de durée plus restreint.

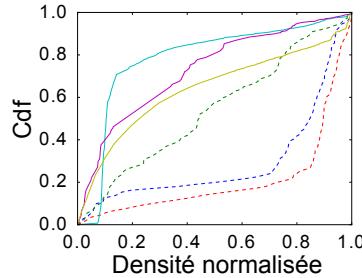


FIGURE 4.3 – Distributions cumulatives (*cdf*) des valeurs de densité renormalisées des sous-flots voisins dans le voisinages à la durée. En trait plein (resp. pointillé), trois candidats du jeu de données Rollernet (resp. Socio Pattern).

Au final, nous capturons pour chaque voisinage les valeurs de densités des sous-flots voisins. Pour le voisinage aux nœuds, il y a $k(|V|-k)$ valeurs de densité différentes. Pour le voisinage au temps de début (resp. à la durée), nous avons une fonction de la densité qui dépend du temps de début (resp. de la durée).

4.2.2 Définition de l'évaluation

Nous évaluons chaque groupe candidat en comparant sa densité à celle des sous-flots dans chaque voisinage. Si un candidat est vraiment pertinent alors il devrait avoir une densité significativement plus élevée que la plupart des sous-flots voisins. Si les densités des sous-flots voisins suivaient une distribution connue, par exemple une loi normale, alors cela reviendrait à évaluer si la densité du candidat est éloignée de la moyenne en observant le *z-score*. De par nos observations, les distributions de densité des sous-flots voisins ne suivent pas la même distribution pour différents candidats. Par exemple, la figure 4.3 présente des distributions cumulatives de la densité pour le voisinage à la durée pour trois groupes candidats des jeux de donnée Socio Pattern et Rollernet¹. Ces distributions sont très différentes et il n'est pas donc pas justifié d'assumer une unique distribution sous-jacente. C'est pourquoi nous utilisons les *percentiles* qui représentent la proportion des valeurs de densités qui sont inférieures à la densité du candidat.

Pour le voisinage aux nœuds qui est représenté par un ensemble, S , de valeurs de densité, le score d'un candidat C_i est :

$$p_{noeud}(C_i) = \frac{\sum_{\delta \in S} \mathbf{1}_{\delta \leq \delta^*}}{|S|}, \quad (4.1)$$

où $\mathbf{1}$ est la fonction indicatrice et $\delta^* = \delta(V(C_i), \beta(C_i), \bar{C}_i)$. Pour les voisinage au temps de début et à la durée qui sont représentés par des fonctions, les scores sont respectivement :

$$p_{début}(C_i) = \frac{1}{\omega - \bar{C}_i - \alpha} \int_{\alpha}^{\omega - \bar{C}_i} \mathbf{1}_{\delta(V(C_i), z, \bar{C}_i) < \delta^*} dz, \quad (4.2)$$

1. Ces jeux de données sont présentés dans la section suivante 4.3.

$$p_{durée}(C_i) = \frac{1}{1.2\Delta_{max} - 0.8\Delta_{min}} \int_{0.8\Delta_{min}}^{1.2\Delta_{max}} \mathbf{1}_{\delta(V(C_i), \beta(C_i), z) < \delta^*} dz. \quad (4.3)$$

Un score de 15% pour un voisinage signifie que la densité du candidat est plus faible que 85% des sous-flots dans ce voisinage et que par conséquent le candidat ne devrait pas être considéré comme pertinent.

Pour résumer, un candidat est évalué par un triplet constitué des scores obtenus pour chaque voisinage. Un candidat est considéré comme pertinent si tous ces scores sont plus élevés que des seuils définis manuellement. La définition de ces seuils est difficile et dépend des caractéristiques du flot de liens. C'est pourquoi nous ne les fixons pas *a priori* mais *a posteriori* après avoir examiné la distribution des scores. Le choix des seuils est décrit plus amplement dans la section 4.4.

4.2.3 Algorithme de calcul des scores

Nous avons défini formellement les trois voisinages mais il faut encore définir un algorithme efficace pour les calculer. En effet même s'il n'est pas nécessaire de construire explicitement les sous-flots des voisinages, il est nécessaire de considérer l'ensemble des sous-flots. Cette nécessité est simple à remplir dans le cas du voisinage aux noeuds car il existe un nombre fini de sous-flots voisins. Pour les voisinages au temps de début et à la durée, il existe une infinité de sous-flots voisins car le temps est continu.

Voisinage aux noeuds

Pour ce voisinage, il est nécessaire d'analyser des sous-flots durant l'intervalle, $[\beta(C_i), \psi(C_i)]$, mais pour différents ensembles noeuds, V' . Il existe plusieurs stratégies possibles pour calculer la densité de ces sous-flots.

La première méthode consiste à manipuler chaque sous-flot de manière indépendante et de calculer leur densité. Pour extraire le sous-flot induit par un ensemble de noeuds sur un intervalle de temps, il est judicieux de d'abord considérer le sous-flot sur les noeuds puis d'intégrer le temps car la construction du sous-flot induit par un ensemble de noeuds est plus rapide. Le calcul de la densité d'un sous-flot induit par un ensemble de noeuds V' , sur intervalle de temps $[\beta(C_i), \psi(C_i)]$ est alors dépendant du nombre de liens existant entre les noeuds de V' avant $\psi(C_i)$ et de la taille de V' : $\tilde{C}_{V'} = O(|L_{\alpha.. \psi(C_i)}(V'^2)| \log(|V'|))$. $|L_{\alpha.. \psi(C_i)}(V'^2)|$ correspond au parcourt de l'ensemble des liens ayant un de leur noeuds dans V' et dont il faut vérifier qu'ils intersectent l'intervalle $[\beta(C_i), \psi(C_i)]$. Pour chacun de ces liens, il faut également tester que l'autre noeud du lien appartienne également à V' ce qui se fait en $\log(|V'|)$. Pour calculer l'ensemble du voisinage pour un groupe de noeuds de taille k , il est alors nécessaire de répéter ce processus pour chaque ensemble de noeuds ce qui donne une complexité de $O(k(|V| - k)\tilde{C}_{V'})$. C'est cette méthode que nous avons implémentée. Elle est d'ailleurs facilement parallélisable.

La deuxième méthode consisterait à tirer parti de la ressemblance des ensembles de noeuds dans le voisinage. La première étape est alors de calculer $d_{t..t'}(V'^2)$, le degré moyen du groupe de noeuds initial. Puis lors de l'évaluation d'un groupe de noeuds voisin V'' , nous savons que V' et V'' ne diffèrent que sur un noeud, v' qui est remplacé par v'' . Pour calculer la densité de V'' sur $[\beta(C_i), \psi(C_i)]$, il suffit de mettre à jour

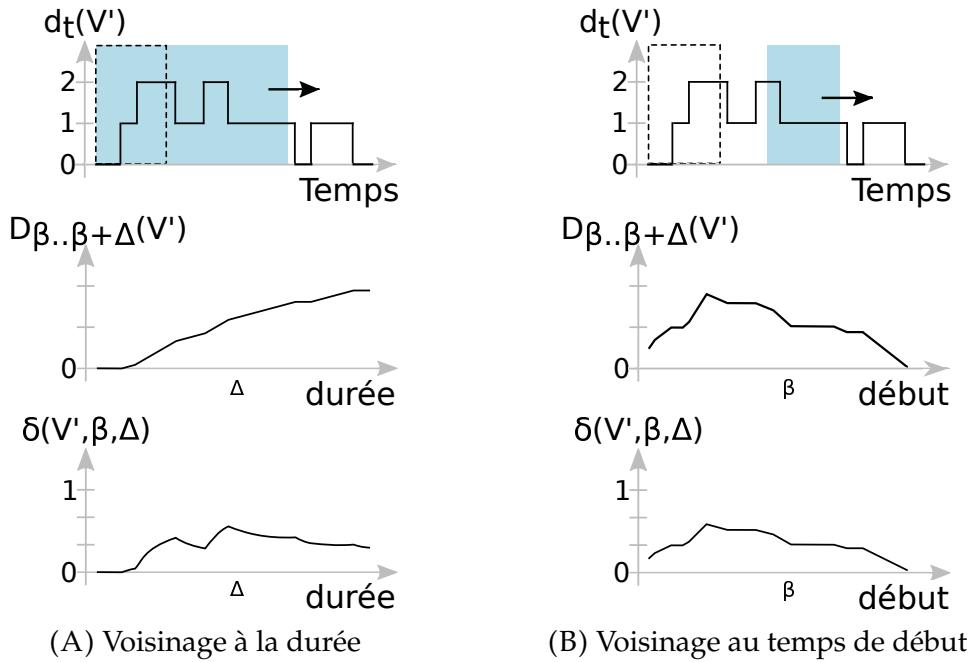


FIGURE 4.4 – Calcul du voisinage à la durée en (A) et au temps de début en (B) pour un groupe ayant un degré temporel donné. Le groupe de liens considéré est le même que celui dans la figure 2.2.

$d_{\beta(C_i).. \psi(C_i)}(V'^2)$ en fonction des liens qui ne sont plus pris en compte et des nouveaux liens pris en compte. C'est à dire qu'il faut considérer $|L_{\alpha.. \psi(C_i)}(v' \cap V')|$ suppressions de liens et $|L_{\alpha.. \psi(C_i)}(v'' \cap V'')|$ ajouts de liens. Pour évaluer l'ensemble des voisinages, il suffit de répéter ce processus pour chaque groupe de noeuds. Cette méthode peut également être parallélisée.

Voisinage à la durée

Pour le voisinage à la durée, l'ensemble de noeuds est fixe et il faut de manière continue augmenter la durée du sous-fLOT voisin. Pour réussir à faire cela, il est nécessaire de se baser encore une fois sur le degré temporel de l'ensemble de noeuds en question.

La densité dépend d'une part de la somme des degrés du groupe candidat, $D_{\beta(C_i).. \psi(C_i)}(V')$, et d'autre part de la la durée de l'intervalle. Comme les degrés temporels sont des fonctions constantes par morceaux, la somme des degrés, qui est l'intégrale des degrés, est une fonction linéaire par morceaux dont les points de changement sont les instants où un lien apparaît ou disparaît. C'est pourquoi, il est possible de définir la densité en fonction de la durée par une fonction par morceaux, voir l'exemple dans la figure 4.4A. L'idée est de mettre à jour la somme des degrés en fonction de l'apparition et disparition des liens tout en tenant compte de la durée du sous-fLOT qui augmente linéairement. La complexité est donc $O(|L_{\beta(C_i).. \beta(C_i) + \Delta_{max}}(V'^2)|)$ si le degré de temporel du groupe est connu.

Voisinage au temps de début

Pour le voisinage au temps début, l'approche est similaire. Il faut mettre à jour la somme des degrés. La principale différence est que les deux bornes de l'intervalle

Datasets	$ V $	$ E $	$\omega - \alpha$
Socio Pattern	180	19774	9 jours
Rollernet	62	15803	3 heures
Reality Mining	94	44975	9 mois
Babouins	28	95616	14 jours

TABLE 4.1 – nombre de nœuds $|V|$, nombre de liens $|E|$ et durée de chaque jeu de données.

changent, alors que dans le voisinage à la durée seule la borne maximum est modifiée. Le parcours est donc un peu plus complexe pour calculer $D_{t..t'}(V')$. Cependant la durée de l'intervalle est fixe, c'est pourquoi il existe une unique bijection entre la somme des degrés et la densité qui est $\delta(V', \beta, \Delta) = \frac{D_{\beta..\beta+\Delta}(V')}{|V'|(|V| - 1)}$. Ainsi, la densité en fonction du temps de début est aussi une fonction linéaire par morceaux.

Voir l'exemple dans la figure 4.4B. La complexité est donc $O(|L_{\alpha..\omega-\Delta}(V'^2)|)$ où Δ est la durée du groupe.

4.3 Jeux de données

Nous appliquons notre méthode sur quatre jeux de données différents. La table 4.1 liste le nombre de nœuds, le nombre de liens et la durée de chaque jeu de données.

Socio Pattern [FB14]² contient le réseau dynamique d'étudiants d'une école préparatoire à Marseille en 2012. Durant l'expérience, 180 étudiants de 5 classes ont porté pendant 9 jours des capteurs de proximité. Ainsi, il est possible de savoir quand deux étudiants interagissent. Dans ce jeu de données, la classe de chaque étudiant est connue.

Rollernet [TLB⁺09] a été collecté durant une randonnée roller ayant eu lieu en 2006 à Paris. Il s'agit d'un événement ayant regroupé 2500 participants. Parmi eux, 62 étaient équipés d'appareils *bluetooth* qui enregistrent l'ensemble des appareils *bluetooth* à portée de communication. Nous nous limitons uniquement aux contacts ayant eu lieu entre les 62 personnes portant un capteur. Des métadonnées sont également associées à ces données et notamment le rôle de chaque personne, *e.g.* membre de l'organisation à l'avant du peloton ou membre d'une association de roller.

Reality Mining [EPL09] représente le réseau d'interactions entre 94 personnes du *MIT Media Laboratory* entre septembre 2004 et juin 2005. Parmi les 94 participants, 68 évoluent dans le même bâtiment (90% étudiants, 10% employés) alors que le reste sont des étudiants de l'école de commerce de l'université. Ce jeu de données a été récolté grâce à des téléphones *bluetooth* prêtés aux étudiants.

2. <http://www.sociopatterns.org>

Jeux de données	$ \mathcal{C} $	$\langle V \rangle$	$\langle L \rangle$
Socio Pattern	12532 (155)	2 (9)	1 (15)
Rollernet	559 (75)	2 (31)	1 (194)
Reality Mining	5737 (474)	2 (12)	1 (36)
Babouin	37671 (1249)	2 (7)	1 (16)

TABLE 4.2 – $|\mathcal{C}|$: nombre de groupes dans la partition, $\langle |V| \rangle$ médiane du nombre de nœuds dans un groupe, $\langle |L| \rangle$ médiane du nombre de liens dans un groupe. Les valeurs en parenthèses correspondent à la valeur quand uniquement les groupes d'au moins 10 liens sont pris en compte.

Babouins [CKW15, SPFCC15] contient la position GPS de 28 babouins (*Papio anubis*) dans le centre de recherche Mpala au Kenya. Ces 28 babouins représentent 80% d'une troupe. Chaque babouin est équipé d'un collier muni d'un capteur GPS qui enregistre la position du babouin toutes les secondes. Nous transformons ces données en un flot de liens en créant un lien entre deux babouins s'ils sont à moins de 1 mètre 50 durant les dernières 10 secondes pour lisser les potentielles imprécisions du GPS. Le code est disponible en ligne³ et est écrit en *rust*.

Même si ces jeux de données sont des réseaux d'interactions face-à-face, ils sont différents dans leur dynamique. Par exemple, le jeu de données Rollernet a environ le même nombre de liens que celui de Socio Pattern alors qu'il ne dure que trois heures contre 9 jours pour Socio Pattern. Enfin, nous n'utilisons pas le jeu de données de courriels utilisé précédemment car notre méthode nécessite un flot de liens avec durées. Nous pourrions utiliser le flot de liens avec des durées ajoutées artificiellement mais elles ne correspondent à aucune réalité.

4.4 Application

Pour chaque jeu de données, nous appliquons notre méthode pour capturer les groupes d'interactions pertinents. Comme présenté dans la section 4.2, la première étape est de trouver une partition, \mathcal{C} , des liens via la méthode de Louvain sur une projection du flot de liens en un graphe statique. Quelques statistiques obtenues pour chaque partition sont listées dans la table 4.2, notamment les nombres de nœuds et de liens médians. La première chose qui est frappante est qu'il existe énormément de très petits groupes. On remarque également que seul le jeu de données Rollernet se distingue des autres en ayant des groupes plus gros. Cette différence est probablement due au flot de liens qui contient beaucoup de liens pour une durée de seulement trois heures.

Afin d'avoir une vision plus précise, nous présentons dans la figure 4.5 les distributions cumulatives inverses du nombre de nœuds, du nombre de liens, de la durée et de la densité des groupes candidats dans le jeu de données Socio Pattern. Les distributions sont toutes hétérogènes sauf pour la densité. Les irrégularités dans la distribution de la densité sont dues aux petits candidats ; typiquement un groupe de 2 liens entre 3

3. <https://bitbucket.org/nGaumont/baboonstreamextractor>

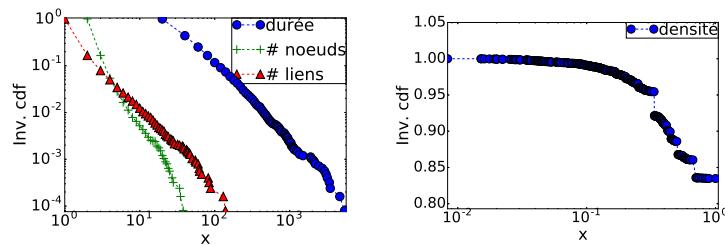


FIGURE 4.5 – Distribution cumulative inverse du nombre de liens, de nœuds et de la durée en (A) et de la densité en (B) pour les candidats trouvés par la méthode de Louvain dans le jeu de données Socio Pattern.

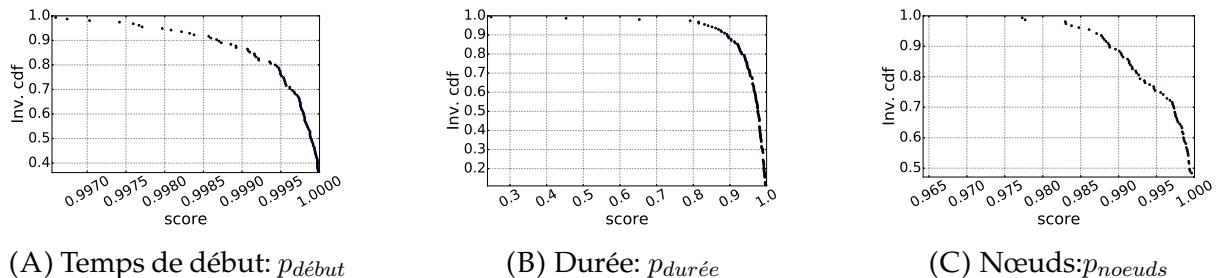


FIGURE 4.6 – Distribution cumulative inverse des scores pour chaque voisinage dans le jeu de données Socio Pattern.

nœuds a une densité de 0.33 et un groupe avec un seul lien⁴ a une densité de 1. Les distributions pour les autres jeux de données sont en appendice pages 117 à 121 dans les figures A.1,A.5 et A.9 Pour ces jeux de données, les distributions du nombre de liens, du nombre de nœuds et de la durée sont également hétérogènes mais les distributions de densité sont légèrement moins irrégulières.

Comme les très petits groupes de liens ont un intérêt limité pour décrire un flot de liens, nous commençons par écarter les groupes ayant moins de 10 liens. Même si cela filtre énormément de groupes, il reste tout de même beaucoup de groupes dont il faut encore évaluer la pertinence.

4.4.1 Évaluation des groupes candidats

Pour séparer les groupes qui sont pertinents des autres, nous utilisons des seuils. Un candidat est ensuite considéré comme pertinent si ses scores sont au dessus des seuils choisis. Baisser les valeurs des seuils entraîne la capture de plus de groupes mais cela ne change ni les groupes candidats ni leurs scores. Ainsi, si un groupe est considéré comme pertinent pour un seuil donné alors il le sera également pour tout autre seuil inférieur. C'est pourquoi, nous fixons les seuils après avoir étudié les distributions cumulatives inverses de score pour chaque voisinage. Ces distributions sont représentées pour le jeu de données Socio Pattern dans la figure 4.6. Pour ce jeu de données, les scores sont très élevés, c'est-à-dire proches de 1, quel que soit le voisinage considéré. On remarque également que les distributions cumulatives inverses chutent toutes à partir d'une certaine valeur de score, que nous utilisons comme seuil.

4. Ils représentent 83% de tous les candidats.

Jeux de données	p_{noeuds}	$p_{début}$	$p_{durée}$
Socio Pattern	0.98	0.998	0.85
Rollernet	0.9	0.7	0.6
Reality Mining	0.97	0.98	0.8
Babouin	0.95	0.99	0.85

TABLE 4.3 – p_{noeuds} , $p_{début}$ et $p_{durée}$: seuils utilisés pour chaque voisinage.

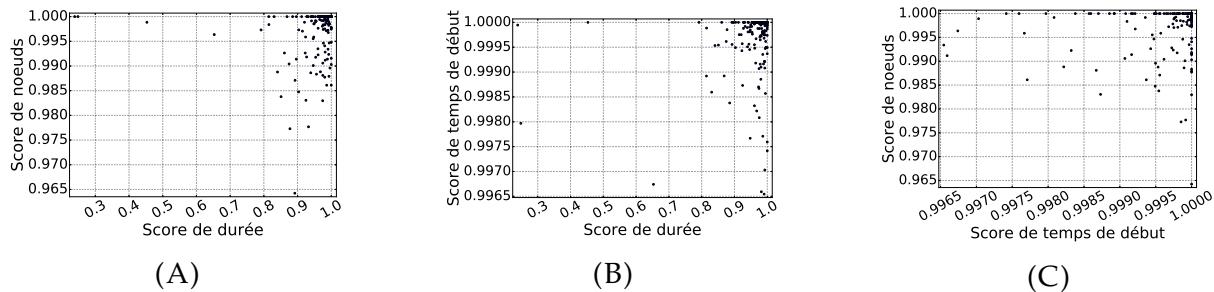


FIGURE 4.7 – Corrélations des scores selon le voisinage dans le jeu de données Socio Pattern.

Nous utilisons pour le voisinage au temps de début, à la durée et aux noeuds les valeurs de seuils suivantes : 0.998, 0.85 et 0.98. Pour les autres jeux de données, les distributions de scores sont présentées en appendice page 117 à 121 dans les figures A.2,A.6 et A.10. Elles sont similaires à celles obtenues pour Socio Pattern mis à part pour le jeu de données Rollernet dont les scores sont plus faibles. Cette différence est sûrement due à la structure plus dense du flot de liens dans le cas de Rollernet. Enfin, la table 4.3 liste les seuils utilisés pour chaque jeu de données.

Un candidat est écarté si un de ses scores est en dessous du seuil correspondant. C'est pourquoi il est important d'analyser par quels voisinages sont filtrés les groupes. Les corrélations entre chaque voisinage sont présentées pour le jeu de données Socio Pattern dans la figure 4.7. Les trois voisinages filtrent des candidats différents. Ils ne sont donc pas redondants. Même si un score élevé dans un voisinage est souvent corrélé avec un score élevé dans les autres, il arrive qu'un candidat ait un score parfait dans un voisinage et un score très faible dans les autres. Cette situation se matérialise dans la figure 4.7 par des groupes situés en haut à gauche ou en bas à droite. Les corrélations entre chaque voisinage pour les autres jeux de données sont présentées en appendice pages 118 à 121 dans les figures A.3,A.7 et A.11.

Afin d'illustrer les scores et les groupes pertinents, nous présentons maintenant une analyse manuelle pour deux candidats extraits des jeux de données Socio Pattern et Rollernet. Nous avons choisi ces jeux de données pour une analyse manuelle car des méta-données sont également associées, ce qui aide pour la validation des résultats.

Étude manuelle d'un groupe de Socio Pattern

Dans les données Socio Pattern, la classe de chaque participant est connue et nous savons également quand commencent les premiers cours et quand ont lieu les pauses. Le groupe que nous considérons contient 50 liens entre 17 nœuds et dure environ 15

minutes, ce qui en fait un des plus gros groupes trouvés. Comme ce groupe commence à 7h44 le deuxième lundi de la capture, il précède le premier cours de la journée qui a lieu à 8h. De plus parmi les élèves participants à ce groupe, 16 font partie de la même classe. C'est pourquoi il est fort probable que ce groupe corresponde à un regroupement d'élèves avant le premier cours.

Cette analyse est une première étape mais elle ne suffit pas. En effet, il est possible que ce regroupement ait en réalité concerné d'autres nœuds, duré plus longtemps ou même commencé à un autre instant. C'est pourquoi nous avons développé les notions de voisinages et nous avons calculé les scores associés à ce groupe.

Voisinage aux nœuds

Pour ce voisinage, le groupe a un score de 0.987. Cela implique que ce groupe a une densité plus importante que 98.7% des sous-flots ayant des nœuds similaires et exactement le même intervalle de temps. Dans le cas de ce groupe à 17 nœuds, il y a $2771 = 17(180 - 17)$ sous-flots dans le voisinage aux nœuds. Ce score élevé nous permet d'être raisonnablement sûr que cet ensemble de nœuds est véritablement important dans cet intervalle de temps.

Voisinage au temps de début

Pour ce voisinage, la densité en fonction du temps de début est présentée dans la figure 4.8A. Les cycles circadiens sont clairement visibles ainsi que le week-end. Par ailleurs, on remarque qu'avec une densité de 0.04 le groupe est plus dense que tous les sous-flots ayant la même durée et les mêmes nœuds mais un temps de début différent. C'est pourquoi le groupe obtient un score de 1 pour ce voisinage. Il est donc également pertinent vis-à-vis de ce voisinage.

Voisinage à la durée

Pour finir, le groupe a un score de 0.95 pour le voisinage à la durée et donc le groupe a capturé une durée pertinente. La densité en fonction de la durée des sous-flots voisins est présentée dans la figure 4.8B. La densité du sous-fLOT est légèrement plus importante si une durée plus courte est utilisée. Ce n'est pas complètement surprenant car la durée du sous-fLOT impacte la densité.

Par ailleurs, il est important de noter deux choses lorsque l'on observe cette courbe. Tout d'abord, on se rend compte que considérer des durées plus longues n'est pas vraiment utile. En effet pour ces jeux de données, la fonction de densité tend vers 0 lorsque la durée est très longue. C'est pourquoi, augmenter l'intervalle de durée pris en compte ne fait qu'augmenter les scores sans apporter d'information. De plus, il est possible qu'avec une durée arbitraire pour définir un sous-fLOT du voisinage, il n'existe aucun groupe de liens exhibant exactement cette durée. En effet, un sous-fLOT induit par un ensemble de nœuds, V' , sur un intervalle donné, I , peut mener à considérer des liens sur seulement une partie de leur durée. C'est le cas notamment si un lien, $(b, e, u, v) \in E$, relie deux nœuds de V' et qu'il commence pendant l'intervalle I mais finit après. C'est pourquoi trouver un groupe de liens avec un score parfait pour la durée est plus surprenant.

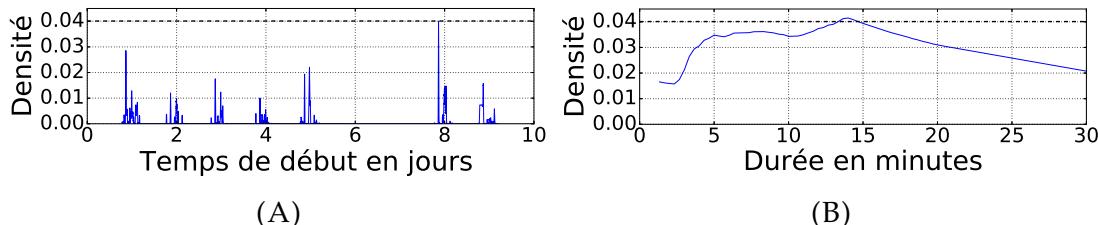


FIGURE 4.8 – Densités du voisinage d'un groupe dans Socio Pattern. En trait plein : la fonction $d(V(C_i), z, \bar{C}_i)$ en (A) et la fonction $d(V(C_i), \beta(C_i), y)$ en (B). En pointillé, la densité du groupe est rappelée.

Enfin, nous avons en sus fait un parcours extensif des densité $d(V(C_i), y, z)$ pour $y \in [\alpha, \omega - \bar{C}_i]$ et $z \in [0.8\Delta_{min}, 1.2\Delta_{max}]$ pour ce groupe en particulier. Pour ce faire, nous avons dû nous baser sur une discréétisation du temps et n'évaluons $d(V(C_i), y, z)$ que toutes 20 secondes. Dans ce contexte, nous obtenons que ce groupe est effectivement très souvent le plus dense.

Étude manuelle d'un groupe de Rollernet

Dans les données Rollernet, nous connaissons le rôle des participants. Certains des participants ne sont connus que comme membres d'une association de roller mais d'autres ont des rôles plus précis, *e.g.* membre organisateur à l'arrière gauche du peloton. Ce genre d'information aide lors de l'étude d'un groupe. Le groupe que nous considérons contient 38 liens, 9 noeuds et dure pendant environ 5 minutes. Ce groupe de liens commence juste au début de la randonnée et 8 des membres du groupe font partie des membres organisateurs à l'arrière de la randonnée. Le dernier fait partie des membres organisateurs mais à l'avant. Ce groupe pourrait indiquer une discussion rapide avant le début du tour. Comme pour le groupe de Socio Pattern, nous analysons les scores obtenus par ce groupe.

Voisinage aux nœuds

Pour ce voisinage, le groupe a un score de 0.99 ce qui indique une fois de plus que cet ensemble de nœuds est important lorsque l'on considère le même intervalle de temps. Il y a par ailleurs $477 = 9(62 - 9)$ sous-flots considérés pour ce candidat dans le voisinage aux nœuds.

Voisinage au temps de début

Pour le voisinage au temps de début, le groupe a un score de 0.85, voir la figure 4.9A. Comme le jeu de données Rollernet est plus court et plus dense que Socio Pattern, la densité est beaucoup plus stable en fonction du temps de début. Cela se reflète d'ailleurs sur le score obtenu qui est plus faible. Cependant comme le groupe de liens en question apparaît 5 minutes après le début de la randonnée, il correspond tout de même à un maximum local de la densité.

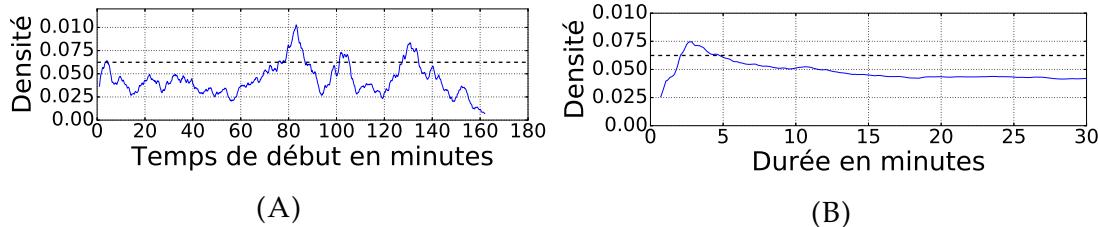


FIGURE 4.9 – Densités du voisinage d'un groupe dans Rollernet. En trait plein : la fonction $d(V(C_i), z, \bar{C}_i)$ en (A) et la fonction $d(V(C_i), \beta(C_i), y)$ en (B). En pointillé, la densité du groupe est rappelée.

	N_c	temps d'exécution
Socio Pattern	136	2 min
Rollernet	37	4 min
Reality Mining	394	1h
Babouin	1023	52 min

TABLE 4.4 – N_C nombre de groupes capturés et temps d'exécution pour chaque jeu de données

Voisinage à la durée

Pour le voisinage à la durée, le groupe a un score de 0.86, voir la figure 4.9B. Encore une fois, le profil de la densité est plus lisse que dans le cas de Socio Pattern et une durée un peu plus courte donnerait lieu à une densité plus élevée. Enfin on remarque que, même si Rollernet est beaucoup plus dense que Socio Pattern, la densité décroît tout de même en fonction de la durée considérée.

Même si ces scores ne sont pas optimum et sont plus faibles que ceux obtenus dans l'exemple précédent, il faut les considérer dans le contexte du flot de liens de Rollernet. Comme Rollernet est globalement plus dense, il est moins surprenant d'observer des sous-flots qui soient denses.

Pour résumer, il y a 12 532 groupes candidats initialement dans le cas de Socio Pattern. Parmi eux, 155 ont plus de 10 liens et 136 sont considérés comme pertinents. La table 4.4 résume le nombre final de groupes pertinents capturés pour chaque jeu de données.

Tous ces groupes ont des scores plus élevés que les seuils que nous avons fixés mais pour la plupart ils n'ont pas des scores parfaits. Cela implique qu'il existe des sous-flots voisins ayant une densité plus importante. C'est pourquoi nous avons également comparé la densité obtenue par le groupe à celle obtenue par le meilleur des sous-flots dans un voisinage donné. Nous observons que dans 75% des cas la densité des groupes est à moins de 20% de l'optimal. Cet écart est relativement stable selon les jeux de données et les voisinages.

Les groupes que nous considérons comme pertinents ont donc une densité proche de l'optimum quand bien même cet optimum n'est pas atteignable par un groupe de liens. De plus, nous avons considéré les voisinages de manière séparée. Or, il se peut par exemple que le sous-flot optimum dans le voisinage au temps de début ne soit pas l'optimum dans le voisinage à la durée.

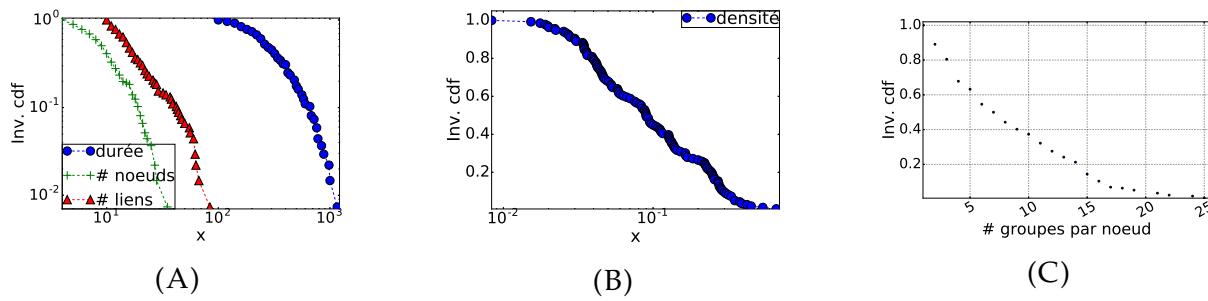


FIGURE 4.10 – Distributions cumulatives inverses du nombre de liens, de nœuds et de la durée en (A), de la densité en (B), du nombre de groupes par nœud en (C) pour les groupes capturés par notre méthode dans les données Socio Pattern.

	Socio Pattern	Rollernet	Reality Mining	Babouin
Représentativité	17.7%	95.4%	80.9%	39%

TABLE 4.5 – Proportion de liens appartenant à un groupe pertinent (représentativité) pour chaque jeu de données

4.4.2 Caractéristiques des groupes pertinents

Afin d'avoir une vision plus précise des groupes capturés comme pertinents, nous présentons les distributions cumulatives inverses du nombre de liens, de noeuds, de la durée et de la densité pour ces groupes dans la figure 4.10. Les distributions sont encore une fois hétérogènes sauf pour la densité. Les distributions pour les autres jeux de données sont présentées en appendice pages 118 à 122 dans les figures A.4,A.8 et A.12. En plus de ces distributions, nous avons également étudié comment ces groupes de liens représentent le flot de liens et comment ils se répartissent dans le temps et la topologie du flot de liens. Par exemple, les 136 groupes pertinents du jeux de données Socio Pattern capturent 3 507 liens, soit 17.7% du flot de liens initial. Les groupes pertinents sont donc bien une structure partielle du flot de liens. Les valeurs de représentativité sont présentées dans la table 4.5. On remarque notamment que même si peu de groupes sont capturés, ils représentent une plus grande proportion des liens du flots de liens pour les autres jeux de données.

Caractéristiques topologiques

Pour la répartition topologique des groupes, la distribution du nombre de groupes pertinents par nœud est présentée dans la figure 4.10C pour le jeu de données Socio Pattern. Tous les nœuds appartiennent à au moins un groupe et quelques nœuds appartiennent même à plus de 20 groupes. Avec une moyenne de 7.4 groupes par nœuds, les groupes pertinents forment donc une structure très recouvrante sur les nœuds. Pour les autres jeux données voir les figures A.4C,A.8C et A.12C. Nous observons également sur ces jeux de données un structure très chevauchante sur les nœuds, en particulier pour le jeu de donnée Babouin. Cette différence est sûrement due au nombre relativement faible de nœuds comparé au nombre de liens : 28 nœuds pour 95 616 liens.

	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 0.2$	$\lambda = 0.3$	$\lambda = 0.5$	$\lambda = 0.7$
Socio Pattern	0.30 (0.55)	0.30 (0.55)	0.30 (0.55)	0.30 (0.55)	0.29 (0.54)	0.33 (0.85)
Rollernet	0.40 (0.52)	0.39 (0.64)	0.41 (0.54)	0.37 (0.54)	0.31 (0.57)	0.26 (0.75)
Reality Mining	0.42 (0.76)	0.42 (0.77)	0.36 (0.88)	0.42 (0.86)	0.38 (0.87)	0.36 (1)
Baboons	0.33 (0.95)	0.38 (1)	0.40 (0.84)	0.46 (0.94)	0.46 (0.93)	0.44 (0.85)

TABLE 4.6 – Indice de Jaccard médian (et maximum) entre les groupes pertinents et les groupes trouvés par *bigclam* dans le graphe agrégé où les liens ayant un poids inférieur à $\lambda\%$ des liens ont été retirés. Les indices de Jaccard supérieurs à 0.8 sont en gras.

Il aurait peut-être été possible de détecter cette structure par une méthode capturant des communautés chevauchantes de noeuds dans un graphe. Afin de tester cette hypothèse, nous créons un graphe agrégé pondéré tel que le poids d'un lien entre deux noeuds soit égal à la somme des durées des liens entre ces deux noeuds dans le flot de liens. Ainsi, le poids des liens permet de garder une partie de l'information temporelle. Pour la détection de communautés chevauchantes, nous avons utilisé l'algorithme *bigclam* proposé par Yang *et al.* [YL13]. Nous l'avons décrit dans la section 1.2.2. Cette méthode ne considère que des graphes non pondérés. C'est pourquoi il est nécessaire de transformer le graphe pondéré en graphe non pondéré. Comme aucune valeur singulière n'est apparue lors de l'étude de la distribution des poids des liens, nous avons créé plusieurs graphes non pondérés selon la règle suivante. Un lien existe dans le graphe non pondéré si le poids associé au lien est supérieur ou égal à $\lambda\%$ des poids de tous les liens du graphe pondéré. Ainsi en faisant varier λ , il est possible de prendre en compte dans une certaine mesure la pondération. Une fois les graphes construits, nous appliquons l'algorithme *bigclam* sur chaque graphe et obtenons des partitions chevauchantes de noeuds.

Nous utilisons l'indice de Jaccard afin de comparer les partitions chevauchantes obtenues par *bigclam* et celle définie par les noeuds induits par les groupes pertinents. L'indice de Jaccard, défini dans la section 1.1, nous permet de trouver, pour chaque groupe pertinent, le groupe trouvé par *bigclam* qui est le plus proche. La table 4.6 liste les similarités medianes et maximales entre les groupes pertinents et ceux trouvés par *bigclam* selon la valeur de λ utilisée. Lorsque l'on étudie cette table, on remarque que les groupes pertinents et les groupes trouvés par *bigclam* sont différents car les indices de Jaccard médians sont très faibles. Il y a toutefois quelques groupes pertinents qui sont retrouvés par *bigclam* dans certains jeu de données. Seulement, cela n'arrive que pour quelques valeurs de λ et très peu de groupes. C'est pourquoi notre méthode permet de mettre en valeur une structure des noeuds qui n'est pas retrouvée par une méthode classique sur le graphe agrégé.

Caractéristiques temporelles

Pour l'aspect temporel, nous avons également étudié le chevauchement temporel entre les groupes pertinents. Pour ce faire, nous regardons la proportions du temps durant laquelle il n'existe aucun groupe présent. Pour tous les jeux de données excepté Rollernet, entre 82% et 95% du temps il n'y a aucun groupe présent. Cette proportion élevée s'explique par la nature des jeux de données qui couvrent de longue périodes,

ce qui inclut des nuits où aucun lien n'apparaît. Pour le jeu de donnée Rollernet, cette proportion de temps sans groupe pertinent n'est que de 15%.

De manière plus importante, il existe également des instants où plusieurs groupes pertinents sont présents. Dans le jeu de données Socio Pattern, c'est notamment le cas lors des pauses ou des repas le midi. Durant ces instants, il peut y avoir jusqu'à quatre groupes présents en même temps. Ces instants importants sont visibles via notre méthode mais ils sont bien évidemment facilement identifiables par d'autres méthodes plus simples telles que le degré temporel.

De manière globale, la structure temporelle des groupes pertinents est très différente de la structure topologique car une part importante du temps n'est impactée par aucun groupe pertinent. Il est toutefois intéressant de noter que les deux structures sont chevauchantes.

4.5 Conclusion et perspectives

4.5.1 Conclusion

Dans ce chapitre, nous nous sommes intéressés à la détection de groupes de liens pertinents dans un flot de liens. À l'inverse des méthodes existantes, nous considérons pleinement l'information temporelle et nous n'utilisons aucune agrégation pour évaluer nos résultats. Cette différence a été permise par l'utilisation du formalisme de flot et de ses métriques mélangeant vraiment structure et temps. De plus, notre méthode ne capture pas des ensembles de noeuds et des intervalles de temps mais des ensembles de liens.

Pour capturer des groupes de liens, nous avons proposé une transformation du flot de liens en un graphe statique non pondéré tel que les liens sont transformés en noeuds dans le graphe. Deux noeuds dans le graphe sont reliés si les liens correspondants dans le flot ont au moins un noeud et un instant de temps en commun. C'est pourquoi, rechercher une communauté dans le graphe statique permet de trouver des groupes liens connectés temporellement et structurellement. Nous utilisons l'algorithme de Louvain qui permet de capturer des partitions de noeuds dans un graphe. Nous obtenons ainsi un ensemble de groupes de liens potentiellement pertinents.

Afin de trouver les groupes pertinents parmi l'ensemble des groupes de la partition trouvée par Louvain, nous avons proposé un critère évaluant la pertinence du sous-flot défini par les liens du groupe. Un sous-flot est jugé pertinent en fonction de scores comparant la densité du sous-flot et la densité des sous-flots proches temporellement et topologiquement. Un sous-flot peut être caractérisé par les noeuds participants à ce sous-flot, l'instant d'apparition du premier lien du sous-flot et la durée du sous-flot. Il est donc naturel de considérer trois voisinages. Chaque voisinage est alors composé de sous-flots différent du sous-flot initial sur un seul aspect : l'ensemble de noeuds, le temps début ou la durée. Le score d'un groupe de liens dans un voisinage est ainsi défini par la proportion de sous-flots étant moins denses que lui. Finalement, le groupe de liens est jugé pertinent si ces scores sont plus élevés que des seuils qui sont fixés *a posteriori*. Ainsi, modifier ces seuils ne modifient pas les groupes candidats mais uniquement les groupes qui sont jugés pertinents.

Nous avons appliqué notre méthode sur quatre jeux de données réels représentant des interactions entre individus. Deux d'entre eux sont des interactions entre étudiants.

Un est un réseau de participant à une randonnée roller à Paris et le dernier est un réseau de babouins. Ils proviennent ainsi de contextes radicalement différents. Cela se traduit notamment sur la densité globale des flots de liens. Pour tous ces jeux de données, nous avons réussi à capturer des groupes ayant des scores très élevés.

La validation de ces résultats est délicate car il n'existe pas de vérité de terrain sur ces données et il n'existe aucun algorithme similaire. Cependant il existe des métadonnées pour deux des jeux de données. De cette manière, nous avons pu analyser manuellement quelques groupes considérés comme pertinents par notre méthode. Nous avons observé que les groupes étudiés correspondaient à des événements existants dans ces jeux de données comme le regroupement d'étudiants avant le premier cours de la semaine. Nous avons également étudié comment l'ensemble des groupes pertinents se répartissent dans le temps et la topologie du réseau. Nous avons observé qu'ils sont très chevauchants sur les nœuds et sur le temps mais qu'ils ne concernent cependant qu'une fraction du temps total.

Afin de tester la nouveauté de la structure que nous capturons, nous avons appliqué un algorithme de détection de communautés chevauchantes, *bigclam*, sur le graphe agrégé. Il apparaît que parmi l'ensemble des groupes pertinents seulement quelques un sont retrouvés par *bigclam*. Ce résultat illustre l'importance de prendre en compte le temps lors de l'étude d'un réseau.

4.5.2 Perspectives

Notre méthode repose sur la projection du flot de liens en un graphe statique, d'une méthode de détection de communauté, une taille de groupe minimum et des seuils qui sont fixé *a posteriori*. Les axes de recherche associés à ces travaux sont principalement de deux types : l'amélioration de la détection des groupes pertinents et l'amélioration de la validation des groupes détectés.

Amélioration de la détection des groupes pertinents

Notre méthode n'est pas encore automatique car il est nécessaire de fixer les seuils des scores et la taille minimum d'un groupe. La taille minimum ne semble pas être très délicate à choisir. Il serait intéressant de voir si notre méthode est capable de filtrer automatiquement les petits groupes. Nous n'avons cependant pas encore poursuivi cet axe de recherche car la prise en compte des petits groupes impacteraît fortement la distribution des scores et donc le choix des seuils. De plus, les seuils nécessitent déjà une connaissances des données.

C'est pourquoi, il serait intéressant de pouvoir automatiser le choix de seuil. Lors de l'application de notre méthode, nous avons à chaque fois observé une forte décroissance dans la distribution des scores. Une méthode prometteuse serait donc de se baser sur la détection de point d'infexion pour déterminer les seuils. Il est par exemple possible de détecter les moments où la dérivée seconde est nulle. Cependant, le comportement que nous avons observé n'est pas forcément universel et il faudrait donc étudier des flots de liens provenant de différents contextes avant de généraliser cette approche.

Il serait également intéressant de remettre en question la méthode de détection utilisée. Nous utilisons l'algorithme de Louvain sur la projection du flot de liens en un graphe statique. D'une part, il est possible d'utiliser d'autres algorithmes de détection

de communautés en tant que partition de noeuds tels que ceux listés dans la sous-section 1.2.1. Mais il est également possible de considérer les couvertures comme une liste de candidats potentiels. Il serait par exemple possible d'appliquer *bigclam* sur la projection. D'autre part, il est également nécessaire de réfléchir à l'impact sur les groupes candidats de l'utilisation de la projection du flot en un graphe car il est possible qu'un groupe pertinent ne soit pas détectable dans la projection. En effet, un lien dans la projection nécessite notamment un chevauchement temporel des liens dans le flot de liens. Si deux liens sont séparés par un intervalle de temps, alors ils ne seront pas reliés dans la projection peu importe la durée de l'intervalle les séparant. Il serait donc intéressant de détecter directement les groupes pertinents.

Il n'est pas possible, à partir de la projection, de calculer la densité et les voisinages d'un groupe de lien. Il est donc nécessaire de manipuler directement un flot de liens pour espérer trouver les groupes pertinents. Comme l'évaluation d'un groupe de liens est locale, une approche locale intégrant au fur et à mesure des liens dans un groupe candidat est envisageable. Le but est alors d'améliorer un groupe jusqu'à obtenir un équilibre de Nash, c'est à dire qu'il ne soit pas possible d'améliorer un score sans dégrader les deux autres. Même si cette approche ne permet pas la détection directe de groupes pertinent, il serait intéressant d'améliorer localement les groupes candidats proposés par l'algorithme de Louvain.

Amélioration de la validation des résultats

Nous avons comparé nos résultats uniquement à une méthode statique car les méthodes de l'état de l'art sur les réseaux dynamiques manipulent principalement des séries de graphes. Or, le changement de formalisme impacte fortement l'objet d'étude. Dans une série de graphes, il y a une agrégation temporelle. L'identité d'un lien est donc perdue lors de l'agrégation et il est ainsi délicat de comparer des groupes de liens dans un flot de liens et des groupes de noeuds dans une série de graphes. Pour pouvoir appliquer les méthodes de série de graphes, il est donc d'abord nécessaire de construire une ou plusieurs transformations du flot de lien en série de graphes. À partir de cette série de graphes, il est alors possible d'appliquer les méthodes existantes pour trouver des communautés dans chaque graphe. Il serait alors intéressant de calculer les scores obtenus par les sous-flots induit par les noeuds d'une communauté sur l'intervalle de temps où la communauté a été détectée dans la série de graphes.

Cette approche permettrait de comparer plus en profondeur les résultats mais pas de les valider. Comme il n'existe pas de données avec une vérité de terrain, une approche prometteuse serait de créer des modèles génératifs de flots de liens permettant de définir des contraintes sur la densité. Ainsi, il serait alors possible de tester de manière quantitative notre méthode. Cela permettrait de plus d'ouvrir la voie pour des méthodes de détection de communautés de liens dans les flots de liens. Nous abordons le problème de la détection de communautés de liens dans les graphes dans le chapitre suivant et nous esquissons des pistes de générations de flots de liens avec une structure dans le chapitre 6.

Chapitre 5

Expected Nodes : communautés de liens dans les graphes statiques

Sommaire

5.1	Travaux existants	74
5.2	Définition d' <i>Expected Nodes</i>	78
5.3	Comparaison	81
5.3.1	Cas du graphe complet	81
5.3.2	Cas de deux cliques chevauchantes	82
5.3.3	Graphe LFR	85
5.4	Calcul et optimisation	88
5.5	Conclusion	90
5.5.1	Perspectives	91

Nous avons vu précédemment les structures de liens dans les flots de liens dans le but de détecter des partitions de liens dans les flots de liens. Avant de s'attaquer à ce problème, il est important de comprendre les solutions existantes dans le cas statique. Les structures communautaires dans les graphes ont été beaucoup étudiées lorsqu'elles concernent les nœuds, voir le 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'intérêt : 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 connu. 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 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.

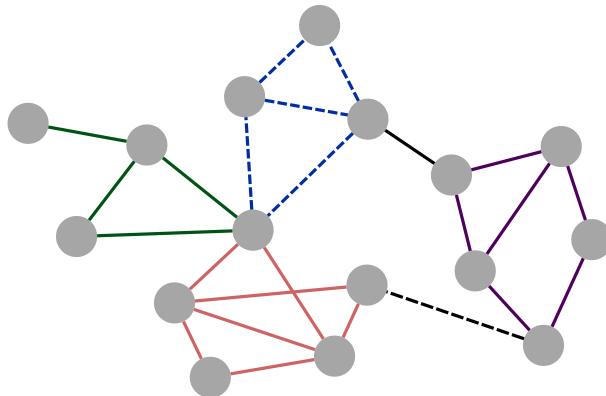


FIGURE 5.1 – Exemple de réseau de personnes avec une structure communautaire sur les liens qui est représentée par la couleur et le style de chaque lien.

Il est également possible de manipuler des partitions chevauchantes ou couvertures. Dans ce cas, chaque noeud 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 noeuds, 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 entière. Pour ce faire, il est nécessaire d'adapter les outils d'analyse pour évaluer directement les partitions de liens. Nous développons ici une approche similaire à ce qui est fait pour les partitions de noeuds et la modularité [NG04]. Le but est de créer une fonction de qualité permettant d'évaluer une partition de liens d'un graphe.

Dans la suite, nous utiliserons les notations suivantes. Soit $G = (V, E)$ un graphe non-orienté avec V l'ensemble des noeuds 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 noeuds internes au groupe L , $V_{out} = \{u \in V \setminus V_{in}, (u, v) \in E \wedge v \in V_{in}\}$ représente les noeuds 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 la figure 5.2).

5.1 Travaux existants

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 noeud dans le *line-graph*. Deux noeuds du *line-graph* sont reliés si les liens correspondants ont au moins un noeud 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 noeuds du *line-graph* représente une partition des liens du graphe initial. Sur l'exemple de la figure 5.3, les

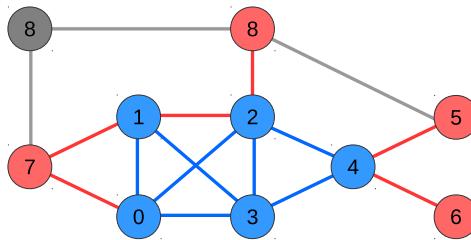


FIGURE 5.2 – Exemple d'un groupe de liens L (en bleu). Les liens rouges sont les liens adjacents L_{out} . Les nœuds internes V_{in} sont en bleu et les nœuds adjacents V_{out} en rouge.

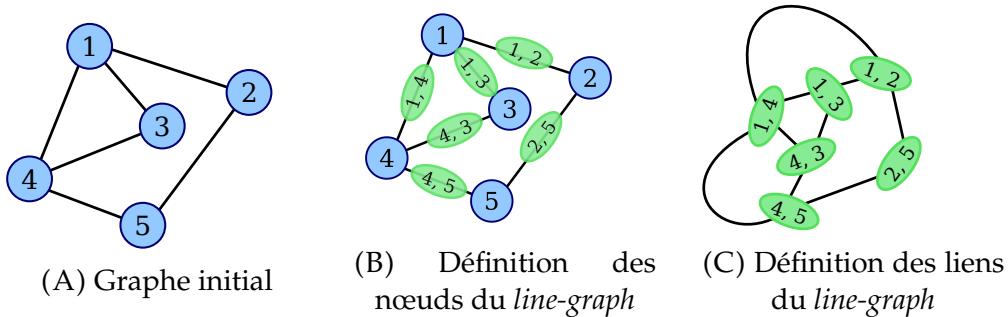


FIGURE 5.3 – Exemple de construction du *line-graph*¹.

liens $(1, 4)$, $(1, 3)$ et $(4, 3)$ du graphe forment un triangle dans le *line-graph* et pourraient être capturés comme étant une communauté. Ces liens dans le graphe initial forment également un triangle et peuvent être considérés comme une communauté valide.

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* ait 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 possibles. Le but est d'observer comment ces 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 nœuds. Une des pires structures communautaires d'un graphe est transformée dans le *line-graph* en la meilleure structure communautaire. En effet, chaque nœud 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 de taille 4 dans la figure 5.4A, on remarque que le *line-graph* est composé de 6 nœuds et de uniquement 12 liens. Plus généralement une clique de taille n dans le graphe initial donne lieu dans le *line-graph* à $\frac{n(n - 1)}{2}$ nœuds 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 trivial d'utiliser le *line-graph* pour trouver des communautés de liens. Il est nécessaire d'adapter les outils à ce type de graphe.

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

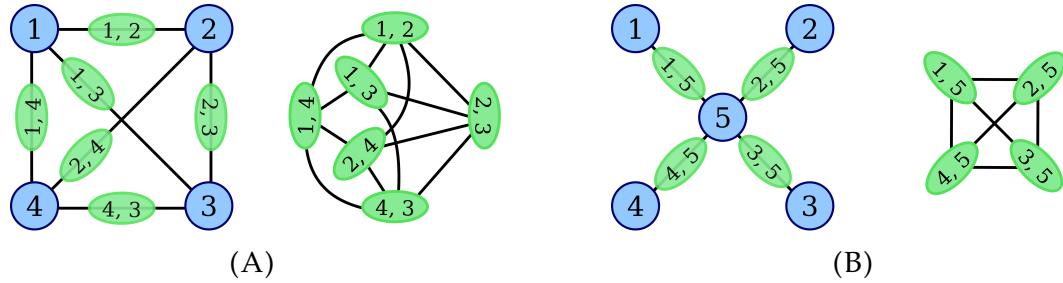


FIGURE 5.4 – Exemple de construction du *line-graph* d'une clique en (A) et d'une étoile en (B).

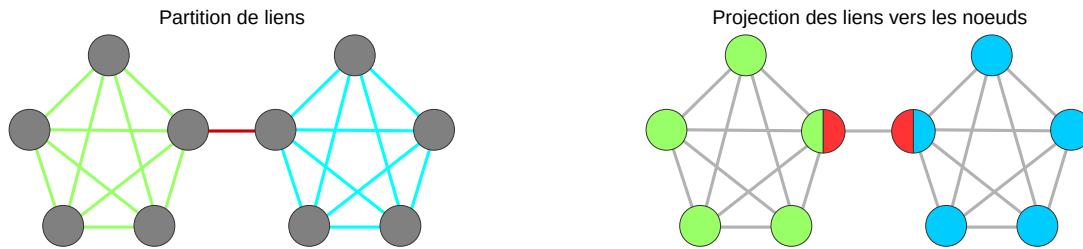


FIGURE 5.5 – Transformation d'une partition de liens à gauche en couverture de nœuds à droite. La couleur représente un groupe.

Il existe d'autres méthodes que le *line-graph* pour la détection et l'évaluation de partitions de liens.

Les algorithmes de propagation de labels ont été étendus pour capturer des partitions de liens [YWW13]. Cependant, ils ne permettent pas d'évaluer la qualité de la partition obtenue.

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

Par exemple dans l'exemple de la figure 5.5, il n'est pas évident que la communauté **rouge** constituée des deux noeuds 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 :

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

où $\min_D(N) = N - 1$ est le nombre minimum de liens nécessaire 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 possible 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 [LZW¹³, SCF¹³] ont par la suite utilisé la *Partition Density* comme fonction à optimiser dans des algorithmes génétiques. Leurs solutions semblent pour l'instant difficilement utilisables 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 [KK14] 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 d'une 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 égal 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 manière suivante :

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

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 noeuds 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 même noeuds induits. D'autre part, les fonctions de qualité $Evans_x$ se basent quant 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, qui est à l'origine de la modularité, permet de mettre en avant des structures qui diffèrent du modèle nul et qui sont donc intéressantes à capturer.

5.2 Définition d'*Expected Nodes*

Une idée souvent utilisée lors de la détection de communautés de noeuds est qu'une communauté devrait avoir beaucoup de liens en interne. Pour ce faire, la modularité compare le nombre de liens partagés par un groupe de noeuds au nombre attendu dans le modèle de configuration, que nous avons évoqué dans la section 1.2.1. Pour notre fonction de qualité *Expected Nodes*, nous utilisons également le modèle de configuration mais avant 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 noeuds internes. En effet, plus le nombre de noeuds internes est faible, plus le groupe de liens ressemble à une clique. De manière similaire à la modularité, nous utilisons le modèle de configuration pour calculer le nombre de noeuds internes attendu dans le modèle de configuration. Si le groupe de liens a moins de noeuds internes qu'attendu alors cela 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 noeuds interne, μ_G , d'un groupe de liens, L , dans le modèle de configuration. Afin d'y parvenir, il est nécessaire de comprendre en détail le modèle de configuration. Le modèle de configuration permet de mélanger les liens d'un graphe de tel sorte que les noeuds aient toujours autant de liaisons que dans le graphe initial. Un moyen de se représenter ce mélange est présenté dans la figure 5.6. De manière imagée, il s'agit de couper tous les liens en deux pour créer des demi-liens. Puis, il suffit de reconnecter aléatoirement tous les demi-liens pour obtenir une génération du modèle de configuration. Donc afin d'étudier comment un groupe de $|L|$ liens est transformé dans le modèle de configuration, il est nécessaire d'étudier comment $2|L|$ demi-liens sont mélangés dans le modèle.

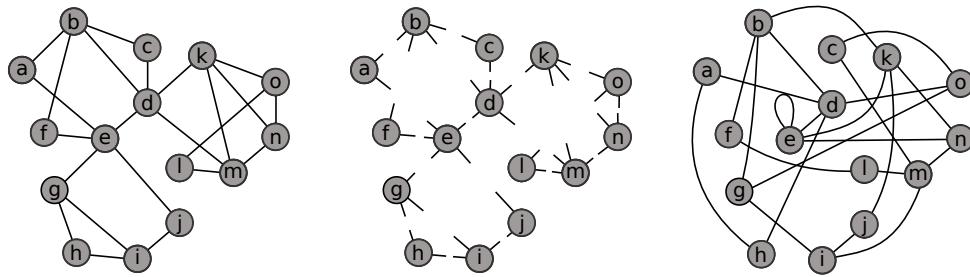


FIGURE 5.6 – Exemple d'une génération du modèle de configuration avec à gauche le graphe initial, au milieu la mise en avant des demi-liens et à droite une génération aléatoire du modèle de configuration

Un nœud est interne à L si au moins un de ses liens appartient à L . Ainsi pour calculer la probabilité qu'un nœud soit interne dans le modèle de configuration, il faut donc calculer la probabilité qu'au moins un de ses demi-liens soit choisi lors du tirage aléatoire et sans remise de $2|L|$ demi-liens parmi les $2|E|$ demi-liens du graphe. 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, s'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.

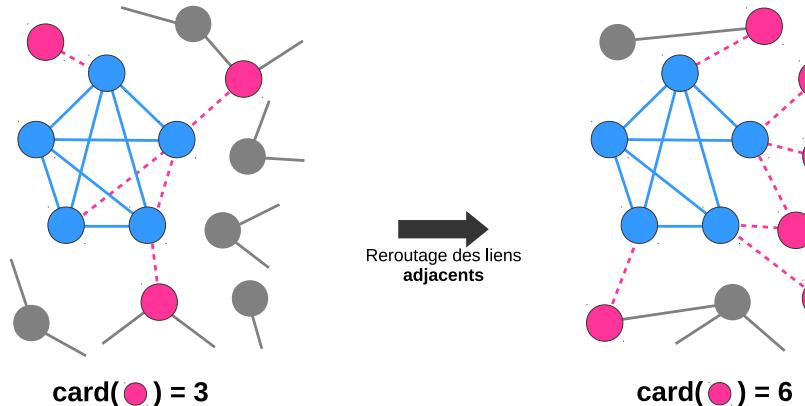


FIGURE 5.7 – Groupe de liens L en bleu et ces liens adjacents en rose pointillés dans le graphe initial à gauche. À droite, une réalisation du modèle de configuration où L a été figé.

Pour un nœud interne u , 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 :

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

Une illustration de ce processus est présentée dans la figure 5.7. Sur cette illustration, le groupe L a un très mauvais voisinage et cela se reflète par un nombre de nœuds adjacents observés plus faible qu'espéré. En particulier, ce sont les liens adjacents reliant deux nœuds internes qui pénalisent l'évaluation car ils comptent chacun pour deux demi-liens adjacents.

Comme il est intéressant de pénaliser les groupes ayant de mauvais voisins 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 nœuds

adjacents V_{out} , on pénalise la présence de liens adjacents entre deux nœuds internes et la présence de nœuds adjacents fortement connectés avec les nœuds internes 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 nœuds adjacents. Ainsi, la Q_{ext} permet de pénaliser les nœuds 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 intégrer les liens adjacents car intégrer 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 nœud adjacent n'est relié qu'à un seul lien adjacent ce qui est le cas idéal. Dans une situation plus générale, la qualité d'un lien isolé dépend du nombre de triangles dans lequel il se trouve. Moins le nombre de triangles est élevé, meilleure sera la qualité externe.

Enfin il est intéressant de noter que la qualité du groupe contenant tous les liens est nulle. En effet comme dit précédemment, $Q_{in}(E)$ est 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 trois jeux de test. Sur ces jeux de test, 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

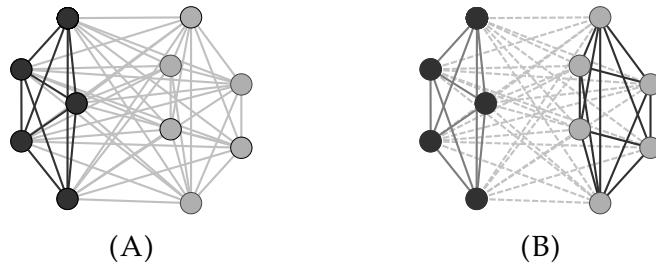


FIGURE 5.8 – Deux partitions de liens pour un graphe complet à 10 nœuds avec $p = 5$: (A) partition en deux groupes et (B) partition en trois groupes. Les nœuds noirs sont les nœuds appartenant à V' et la couleur d'un lien correspond à son groupe.

un graphe complet de 100 nœuds². Sur ce graphe, nous définissons plusieurs partitions. La première est la partition triviale où tous 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 les liens de $V \setminus V' \times V \setminus V'$ sont dans un troisième. Ces répartitions sont illustrées dans la figure 5.8.

Comme le graphe est un graphe complet, la meilleure solution est d'avoir un seul groupe contenant l'ensemble des liens, *i.e.* la partition triviale devrait avoir une meilleure évaluation que les autres partitions. La figure 5.9 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 est égal à 24 est complètement équivalente à la partition en deux groupes lorsque p est égal à 76. Par ailleurs, il est possible de définir formellement la qualité de chacune de ces 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 meilleure des partitions. La fonction *Evans3* diffère légèrement des deux autres car elle a une amplitude plus faible ($\approx 10^{-3}$).

5.3.2 Cas de deux cliques chevauchantes

Le second jeu de test est un peu plus évolué puisqu'il s'agit deux cliques de 100 nœuds chacune se chevauchant de plus en plus. Ainsi, nous générerons les graphes en fonction du nombre de nœuds appartenant aux deux cliques. Sur ce graphe, nous définissons également plusieurs partitions. La première est la partition triviale où tout

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

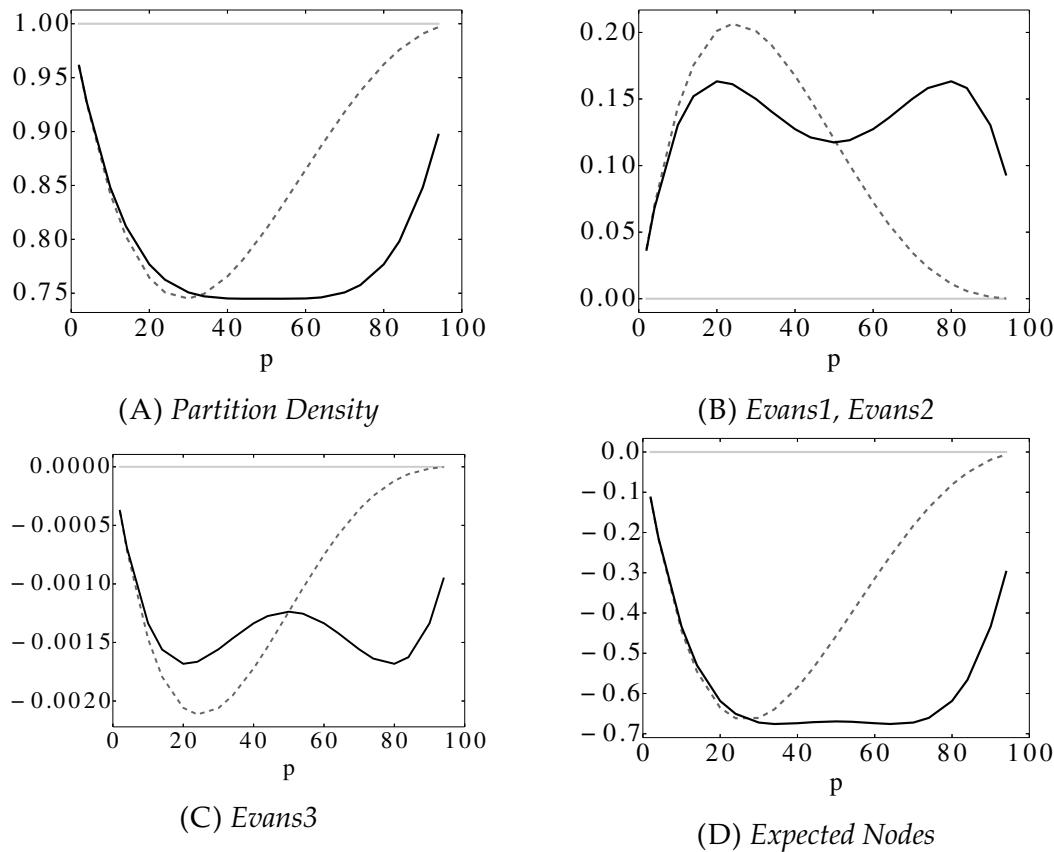


FIGURE 5.9 – Évaluation des 5 fonctions de qualité sur un graphe complet de 100 nœuds pour trois type de partitions. Les partitions testées sont présentées dans la section 5.3.1. Par définition, les résultats pour *Evans1* et *Evans2* sont identiques. Les lignes en gris, noir et pointillées représentent respectivement la partition triviale, les partitions en deux groupes et les partitions en trois groupes.

les liens sont dans un unique groupe. Nous définissons également cinq autres partitions qui contiennent 2, 3 ou 5 groupes. Soit V_1 l'ensemble de nœuds de la première clique, V_2 celui de la deuxième clique et $V_{12} = V_1 \cap V_2$ l'ensemble de nœuds commun à V_1 et V_2 . Cette taxonomie des nœuds est illustrée dans la figure 5.10B.

Par abus de notations, nous notons également $V_{11} = V_1 \setminus V_2$ et $V_{22} = V_2 \setminus V_1$. Il y a deux partitions en deux groupes : la partition 2A et 2B. Dans la partition 2A, les liens entre les nœuds de V_{22} forment un groupe alors que le reste des liens forme l'autre groupe. Dans la partition 2B, les liens entre les nœuds de V_1 forment un groupe alors que le reste des liens forme l'autre groupe.

Il y a également deux partitions en trois groupes : la partition 3A et 3B. Dans la partition 3A, les liens entre les nœuds de V_{12} forment un premier groupe et les liens appartenant à $V_1 \times V_{11}$ (resp. $V_2 \times V_{22}$) forment un deuxième (resp. troisième) groupe. Dans la partition 3B, les liens entre les nœuds de V_{11} (resp. V_{22}) forment un premier (resp. deuxième) groupe alors que le reste des liens forme un troisième groupe.

Enfin dans la partition 5 en cinq groupes, il existe 3 groupes correspondant aux liens entre les nœuds de respectivement V_{11} , V_{12} et V_{22} . Il y a également un groupe pour les liens appartenant à $V_{11} \times V_{12}$ et un autre pour ceux appartenant à $V_{12} \times V_{22}$.

Ces partitions sont illustrées dans la figure 5.10.

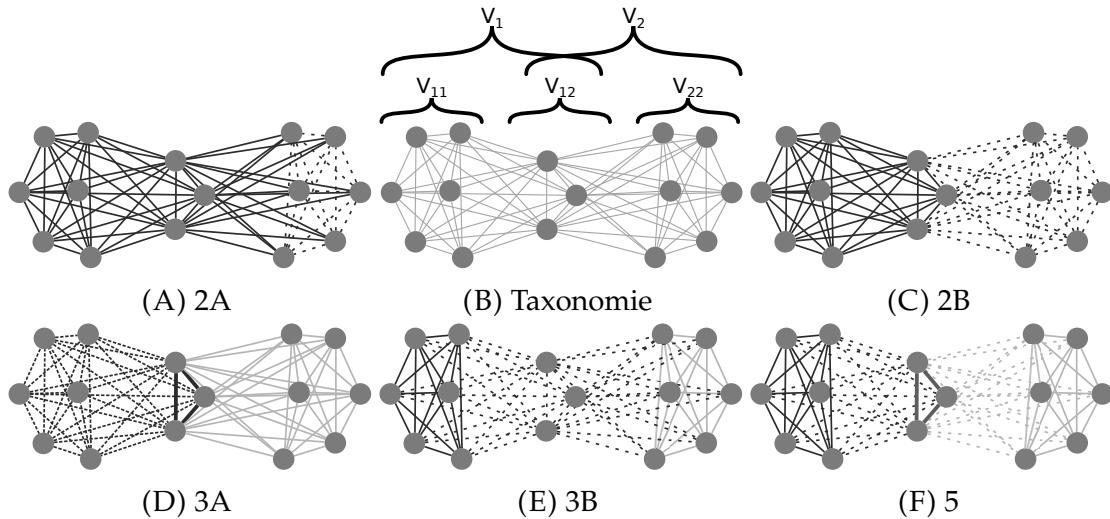


FIGURE 5.10 – Différentes partitions d'un graphe constitué de deux cliques de 9 se chevauchant sur 3 nœuds. En (B), taxonomie des nœuds du graphe. En (A) et (C), deux partitions en deux groupes. En (D) et (E), deux partitions en trois groupes. En (F), une partition en cinq groupes. La couleur d'un lien correspond à son groupe.

Pour chaque graphe en fonction du chevauchement p , nous testons ces cinq partitions 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*. La figure 5.11 présente les résultats.

Comme le graphe étudié est composé de deux cliques se chevauchant de plus en plus, il n'y a pas une seule vérité de terrain. Lorsque le chevauchement est faible, les partitions $2B$ et $3A$ devraient avoir les meilleures évaluations. Lorsque le chevauchement est plus important, c'est la partition 1 en une seule communauté qui devrait avoir la meilleure évaluation.

Dans le cas de *EvansX*, ce sont les partitions EX qui obtiennent les meilleures évaluations, en particulier pour *Evans1* et *Evans2*. L'optimisation de ces fonctions de qualité mène donc à trouver des partitions qui ont une qualité plus élevée que la vérité de terrain mais qui sont différentes. Ces fonctions de qualités ne sont donc pas à même de capturer une vérité de terrain comme la meilleure partition. Il est toutefois important de noter que dans le cas de *Evans3*, la différence entre $E3$ et $2B$ reste faible.

Pour les fonctions de qualité *Partition density* et *Expected Nodes*, c'est bien la partition $2B$ qui obtient le meilleur score lorsque le chevauchement est faible. Lorsque le chevauchement est plus important, c'est la partition 1 qui obtient la meilleure évaluation. La principale différence entre ces deux fonctions de qualité est la proportion de chevauchement à partir duquel la meilleure partition change. Dans le cas de *Partition density*, ce changement intervient lorsque le chevauchement dépasse 40 nœuds. Pour *Expected Nodes*, ce changement intervient lorsque le chevauchement dépasse 30 nœuds.

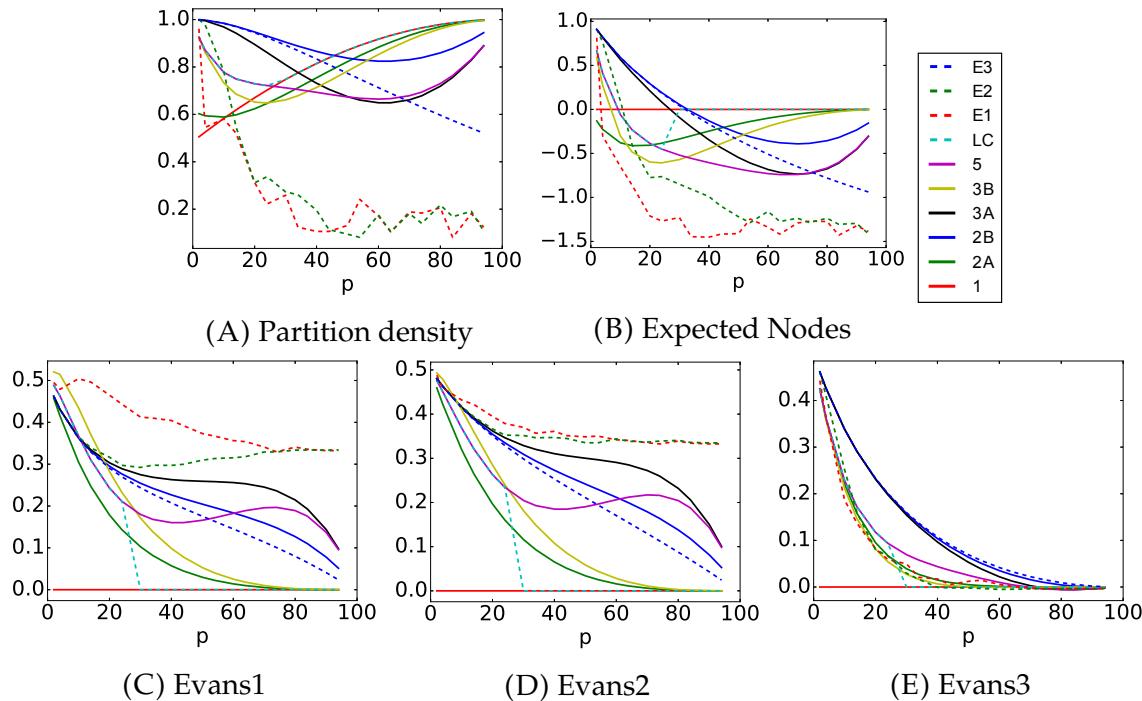


FIGURE 5.11 – Évaluation des cinq fonctions de qualité sur un graphe composé de deux cliques de 100 nœuds se chevauchant sur p nœuds pour dix partitions différentes. Les partitions testées sont présentées dans la section 5.3.2. Les lignes en trait plein et pointillé représentent respectivement les partitions fixées arbitrairement et les partitions trouvées empiriquement.

5.3.3 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. 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 5.12.

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 possibles 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.

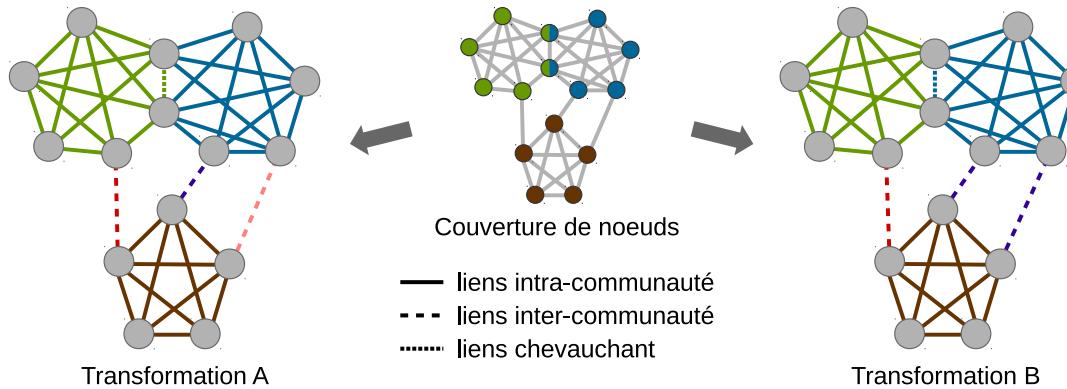


FIGURE 5.12 – Construction de TA et TB depuis une couverture de nœuds. La couleur des liens indique leur groupe.

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 formelle cette transformation qui est également illustrée dans la figure 5.12. 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 paramètres classiquement utilisé dans la littérature [For10]. Ainsi, nous avons générés 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 à -1 comme paramètre. Enfin, 90% des liens 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-communauté, 625 liens inter-communauté et seulement 5 liens chevauchants.

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 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 5.13.

Dans LC , TA , TB , il y a 720, 650, 70 groupes en moyenne. Dans $E1$, $E2$ et $E3$, il n'y a que 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 deux partitions les plus proches l'une de l'autre. Nous remarquons que les partitions $E1$, $E2$

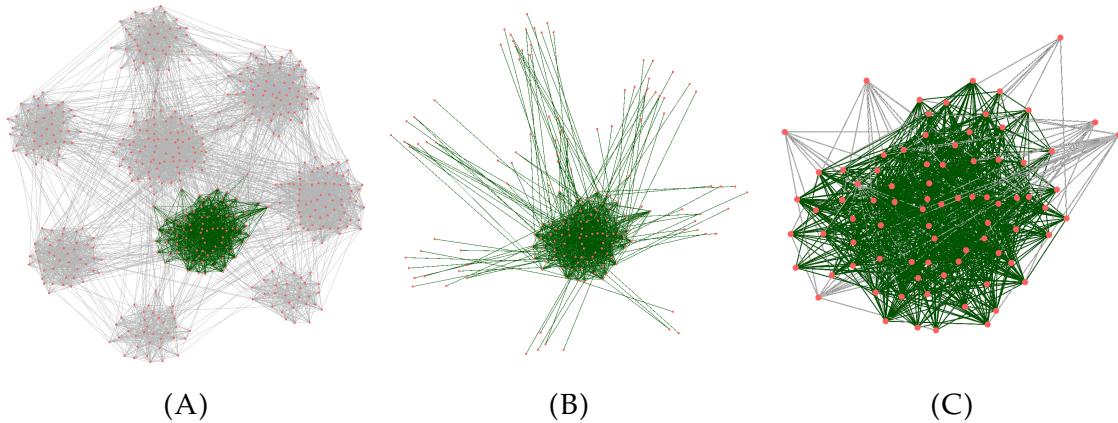


FIGURE 5.13 – Exemple de graphe généré par le LFR en (A) avec une communauté de liens intra-communauté de la vérité de terrain mise en avant en vert. En (B) zoom sur le groupe le plus proche dans $E2$ dont les liens sont en verts. En (C) zoom sur le groupe le plus proche dans LC dont les liens sont en verts. Dans la figure (C), les liens de la communauté de la vérité de terrain sont en gris.

et $E3$ sont similaires. Ensuite, nous remarquons que les partitions EX diffèrent de TA et de TB uniquement sur les liens inter-communautés. En effet s'ils ne sont pas pris en compte lors de la comparaison, alors EX , TA et TB sont identiques. Il semble en effet que les liens inter-communautés soient arbitrairement distribués entre les plus grosses communautés adjacentes pour les partitions EX , ce qui est visible dans la figure 5.13B. 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 5.13C. 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 qualité. Comme le processus de génération de graphe est aléatoire, les évaluations présentées dans la figure 5.14 représentent 30 générations. On remarque tout d'abord que ni TA ni TB n'a la meilleure évaluation selon $EvansX$ (figures 5.14C à 5.14E) ou *Partition Density* (figure 5.14A) même si TA et TB représentent nos vérités de terrain. Dans le cas de $EvansX$, ce sont 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 5.12. Ce phénomène est très pénalisé par *Expected Nodes*. Enfin selon *Expected Nodes*, les partitions EX ont une faible qualité et cela est également dû aux liens inter-communautés. En effet en étant intégrés à une communauté adjacente, ils augmentent fortement le nombre de noeuds internes ce qui fait baisser Q_{in} .

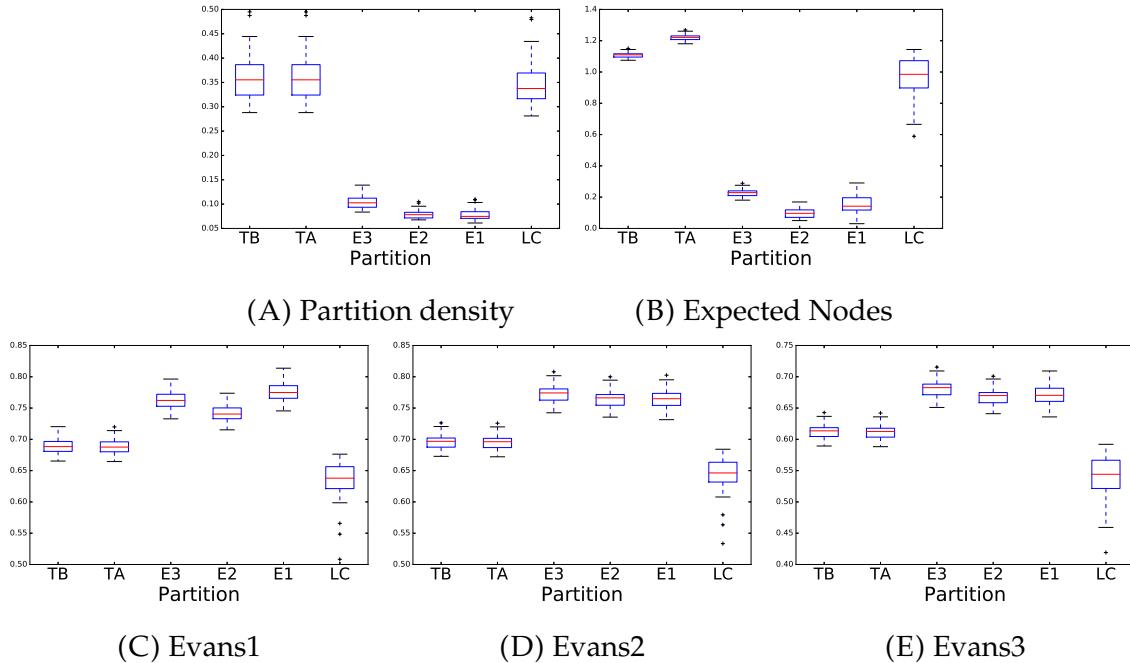


FIGURE 5.14 – Boîtes à moustaches des évaluations des cinq fonctions de qualité pour les différentes partitions de liens. La boîte représente le premier et troisième quartile ainsi que la médiane. Les moustaches s'étendent sur 1.5 fois l'écart entre le premier et le troisième quartile. Les croix sont les points au delà des moustaches.

Avec ce test, nous mettons en évidence les limites des fonctions de qualité existantes. Les fonctions de qualité *EvansX* ne peuvent pas permettre de capturer les vérités de terrain générées par LFR. En effet, l'algorithme de Louvain est capable de trouver des partitions ayant une évaluation plus élevée 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* ne semble pas à même de différencier clairement des partitions clairement non similaires. 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 coûteux. 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 uniquement 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 coûteux 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 5.15 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.

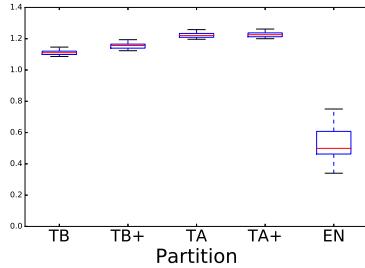


FIGURE 5.15 – Boîte à moustaches des évaluations selon *Expected Nodes* pour différentes partitions. *TA* et *TB* sont les vérités de terrains, *EN* est la partition trouvée par notre algorithme et enfin *TA+* et *TB+* sont les versions améliorées par notre algorithme de *TA* et *TB*.

Dans le cas de *TA*, chaque lien inter-communauté constitue initialement une communauté. Or, il se peut que plusieurs liens inter-communautés soient connectés aux même noeuds. 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 5.15 où les partitions *TA+* et *TB+* sont les versions améliorées de *TA* et *TB*.

5.5 Conclusion

Nous avons introduit de nouveaux critères pour l'évaluation des partitions de liens tenant compte de la répartition des noeuds internes et externes d'un groupe. À partir de ces critères, nous avons défini une mesure de qualité, *Expected Nodes*, basée sur la différence entre le nombre de noeuds induits par un groupe de liens et le nombre de noeuds attendu dans un modèle nul. Cette approche est similaire à celle qui a donné lieu à la modularité. En plus de s'inspirer de la modularité, *Expected Nodes* ne prend pas uniquement en compte la répartition de ses noeuds induits mais également celle des noeuds 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 générées par LFR sont des quasi-optimum locaux selon *Expected Nodes*. Bien qu'il soit possible d'améliorer légèrement les vérités de terrain, la structure de l'optimum local est très proche de la partition initiale.

5.5.1 Perspectives

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 mouvement envisagés pour modifier une solution courante ne soient pas suffisants pour échapper à des maximums locaux qui sont bien inférieurs au maximum global. Il faudrait donc en 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 élevé. 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 donné. Pour 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 noeud 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 augmenter la qualité externe. De cette manière, il devrait être possible d'éviter des maximums locaux. 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 mais de le rendre plus rapide. Dans sa version courante, l'algorithme est trop coûteux 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 pour le 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 à partir 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 de 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 triviaux à calculer. Il faudrait des parcours locaux sur $(A \cup B)$ et son voisinage pour arriver à mettre

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

à jours 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 testée qui aurait une bien meilleure qualité que les vérités de terrain. 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 analogue, 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 jusqu'à 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 liée 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 possibles. 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 nœuds peuvent être reliés par plus qu'un lien. Les notions de tirages aléatoires peuvent s'adapter à ce genre de graphe assez facilement. Il y

a toutefois une différence majeure 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

Génération de flots de liens avec structure communautaire

Sommaire

6.1 Travaux existants	96
6.1.1 Séries de graphes	96
6.1.2 Flots de liens	97
6.2 Méthode de génération	98
6.2.1 Caractéristiques du générateur	100
6.3 Applications	101
6.3.1 Étude de méthodes statiques appliquées aux flots de liens . . .	101
6.3.2 Vers des fonctions de qualité dans les flots de liens	103
6.4 Conclusion	107
6.4.1 Perspectives	108

Nous avons vu dans le chapitre précédent que la génération de graphes ayant une structure est un atout pour tester et valider des fonctions de qualité. Pour être utile à la validation de fonctions de qualité, un générateur doit être capable de construire des graphes ayant des caractéristiques vraisemblables de structures communautaires. Pour ce faire, plusieurs contraintes peuvent être considérées. Ce domaine de recherche est d'ailleurs toujours actif [TRC11, OD14] car il est parfois nécessaire d'intégrer de nouvelles contraintes sur les graphes générés. Il existe cependant peu de méthodes pour générer des réseaux dynamiques, que ce soit sous la forme de série de graphes, de graphes temporel ou de flots de liens.

Dans ce chapitre, nous présentons un premier générateur de flots de liens sans durée avec une structure communautaire sur les liens. L'intuition derrière ce générateur est la suivante. Dans un réseau de personnes, une interaction existe entre deux personnes dans le cadre d'un intérêt commun délimité dans le temps. Ce point commun peut être un entraînement de sport, le travail ou une réunion de famille. Par exemple, des collègues communiquent principalement pendant la journée et rarement le soir. Au contraire, un groupe d'amis communiquent principalement le soir et le week end. Ainsi, toutes les personnes partageant le même point commun interagissent entre elles aux mêmes instants. Cela signifie que les dynamiques de communications entre deux personnes sont principalement liées à la raison de la communication. Avec cette hypothèse, il suffit alors de connaître l'ensemble des intérêts d'une personne pour comprendre la dynamique de ses interactions. De plus comme un lien est généré à cause d'un intérêt commun, il est possible d'assigner à ce lien la communauté correspondant

à cet intérêt. Pour générer un flot de liens, il suffit alors de répartir des points d'intérêt à chaque personne puis de générer des liens entre deux personnes lorsqu'un intérêt commun les relie. C'est selon ce principe que nous construisons notre générateur de flots de liens avec structure communautaire sur les liens.

En modifiant la répartition des point d'intérêts dans le temps et entre les personnes, il est ainsi possible de générer des flots de liens très divers.

Nous explorons les possibilités ouvertes par ce générateur via deux pistes distinctes. Tout d'abord, nous testons différentes méthodes de détection de communautés sur des projections du flot de liens en un graphe. Puis, nous faisons une première proposition pour une fonction de qualité évaluant les partitions de liens des flots de liens.

Le chapitre est organisé de la manière suivante. Dans la section 6.1, nous revenons sur les travaux existants qui traitent de la génération de réseaux dynamiques. Puis dans la section 6.2, nous présentons notre méthode de génération de flots de liens. Enfin dans la section 6.3, nous présentons l'utilisation du générateur pour tester des méthodes de détection de communautés statique et la définition d'une fonction de qualité.

6.1 Travaux existants

Comme nous l'avons vu dans le chapitre 1, les méthodes proposées sont différentes selon le formalisme utilisé.

6.1.1 Séries de graphes

Dans le cas de séries de graphes, il est possible de générer un graphe en fonction du graphe précédent dans la série. Par exemple, les *edge-Markovian dynamic graphs* [CMM⁺08] génèrent l'apparition et la disparition de liens à l'aide d'un processus de Markov à deux états : lien présent ou lien absent.

Les *activity driven model* [PGPSV12, LSK15, MSPS15] reposent également sur le graphe généré à l'instant précédent. Dans ce genre de modèles, un lien est créé entre deux noeuds en fonction de la propension intrinsèque de ces noeuds à créer des liens¹ et potentiellement de l'existence ou de l'absence de ce lien dans le graphe précédent. Ce genre de modèles capture très bien l'hétérogénéité des noeuds mais ne permet pas de générer une structure communautaire.

Il est également possible d'appliquer un modèle génératif sur chaque graphe de la série indépendamment. C'est ce que propose de Granell *et al.* [GDA⁺15] qui utilise le SBM pour générer un graphe. Après chaque génération, des modifications sont appliquées au SBM pour représenter l'accroissement, le rétrécissement, la séparation ou la fusion de communautés. Voir l'illustration dans la figure 6.1. Les méthodes de SBM sur des séries de graphes, évoquées dans la sous-section 1.3.1, peuvent également être utilisées pour la génération. Cependant, elles sont plus restreintes car elles presupposent des contraintes sur l'évolution du SBM afin qu'il soit identifiable.

1. Le degré dans le graphe précédent est généralement utilisé.

2. Image provenant de <http://journals.aps.org/pre/abstract/10.1103/PhysRevE.92.012805>.

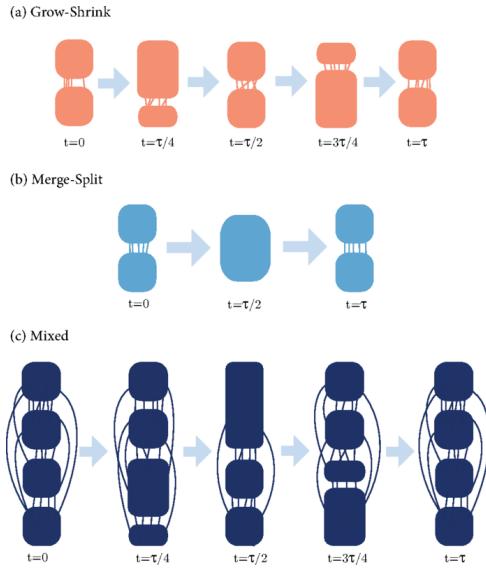


FIGURE 6.1 – Représentation schématique d'évolution possible d'une structure communautaire de séries de graphes pouvant être générées par la méthode de Granell *et al.* [GDA⁺15].²

Ces méthodes génèrent aisément des structures communautaires mais ne permettent de générer que des séries de graphes.

6.1.2 Flots de liens

Dans les flots de liens, il n'existe pas à notre connaissance de travaux étudiant la génération de flots de liens avec une structure communautaire. Il existe principalement des méthodes étudiant les distributions de temps inter-contacts [MSMA08, MSCA09, VGB14]. Il y a dans la littérature un large consensus sur le fait que la distribution des temps inter-contacts est souvent à queue lourde (*heavy-tailed*) dans les jeux de données réels [KKP⁺11, KKBK12, KP15]. D'autre part, les temps inter-contacts ne semblent pas suivre un processus de Markov simple car il y a des corrélations et des effets de mémoire entre les apparitions de liens.

Toutes ces méthodes génèrent des flots de liens mais ne considèrent aucune structure. Quelques méthodes se distinguent de ces travaux et génèrent une structure même s'il ne s'agit pas forcément de structure communautaire. C'est le cas des travaux de Starnini *et al.* [SBPS13] et de Zhang *et al.* [ZLLC15]. Dans ces méthodes, un flot de liens est généré à partir d'un système multi-agents où chaque agent se déplace dans un espace 2D borné. Un lien existe entre deux agents lorsqu'ils sont à distance de communication. Dans le modèle de Zhang *et al.*, la direction prise par chaque agent n'est pas aléatoire mais en fonction de l'agent voisin étant le plus attractif. Il s'agit donc d'un modèle de mobilité préférentielle. Bien que ces méthodes génèrent des structures. Il n'agit pas de structure communautaire explicite et elles ne peuvent pas être représentées sur les liens du flot de liens.

Barrat *et al.* [BFLY13] proposent une autre méthode basée sur l'utilisation de marches aléatoires. Ils utilisent un graphe statique pondéré et orienté qui représente l'ensemble des interactions qui pourront exister dans le flot de liens. Ce graphe peut provenir de données réelles ou être généré. À partir de ce graphe, différentes marches aléatoires

sont simulées. Chaque marche est caractérisée par un temps de début, un nœud de départ, un nombre de sauts et les temps séparant chaque saut. Ces caractéristiques peuvent être définies arbitrairement ou selon des distributions données. Chaque marche donne lieu à une liste d'interactions : $\{(t_1, u_0, u_1), \dots, (t_k, u_{k-1}, u_k)\}$ pour une marche de k sauts. Lors de la marche aléatoire, le prochain lien est choisi en fonction de son poids dans le graphe pondéré et le poids du lien choisi est décrémenté de 1 dans le graphe pondéré. Un lien ne peut être choisi que si son poids est supérieur à 0. L'union des marches simulées forme ainsi le flot de liens.

Si le graphe statique utilisé possède une structure communautaire, il est possible que le flot de liens généré ait également une structure communautaire. Cependant cette dépendance ne permet pas de générer des structures communautaires qui soient invisibles dans le graphe statique. Dans l'exemple *Merge-Split* de la figure 6.1, il existe deux communautés aux instants $t = 0$ et $t = \tau$ mais ces deux communautés sont fusionnées à l'instant $t = \tau/2$. Il existe donc une structure communautaire dans le réseau dynamique initial mais cette structure n'est pas directement identifiable dans le graphe agrégé. C'est pourquoi, il se peut qu'il n'existe aucune structure communautaire dans le flot de liens généré à partir de ce graphe statique.

Résumé

Dans le cas des séries de graphes, il existe des propositions de générateur avec structure communautaire. Dans le cadre de flots de liens, les générateurs ne permettent pas ou alors que partiellement de générer une structure communautaire.

6.2 Méthode de génération

Nous définissons maintenant notre générateur de flots de liens sans durée ayant une structure communautaire. Notre approche est similaire au SBM car nous considérons que l'apparition de liens entre deux nœuds ne dépend que des communautés auxquelles ils appartiennent. Plus ils ont de communautés en commun, plus il existera de liens entre ces nœuds. S'ils n'ont aucune communauté en commun, alors il n'y aura pas ou peu de liens entre ces nœuds. Tout comme le SBM, nous utilisons un graphe d'affiliation des nœuds aux communautés où il existe un lien entre un nœud et une communauté si le nœud appartient à la communauté.

Dans le cadre de notre générateur, une communauté représente un intérêt et une personne peut naturellement avoir plusieurs intérêts. Ainsi en reprenant l'exemple précédent, il existe dans le graphe d'affiliation autant de nœuds que de personnes et il existe une communauté représentant les entraînements de sport du mercredi soir, une représentant une réunion de famille, etc.

De ce graphe biparti, les approches statiques déduisent la probabilité que deux nœuds soient reliés par un lien. Dans le contexte de flots de liens, il ne suffit pas de calculer la probabilité qu'un lien apparaisse dans le cadre d'une communauté. Il faut générer un ensemble d'interactions temporelles entre ces deux nœuds au sein de cette communauté. C'est pourquoi à chaque communauté est associé un processus stochastique de génération de liens dépendant du temps. Ainsi, chaque communauté partagée par deux nœuds donne lieu à la génération d'un ensemble de liens selon le processus stochastique de cette communauté. Par exemple, le processus stochastique de la communauté sport est défini par une forte activité durant les séances d'entraînement et

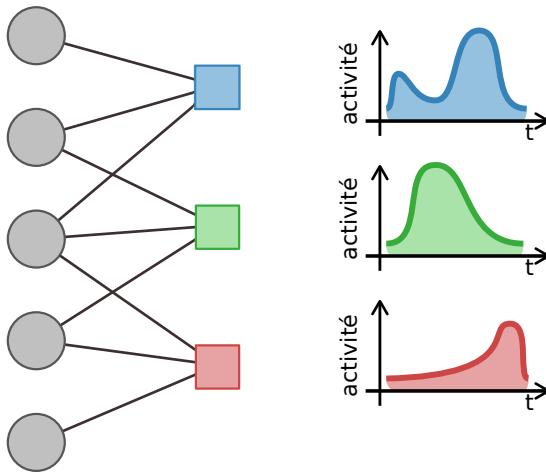


FIGURE 6.2 – Schéma représentant la structure communautaire d'un flot de liens dans notre générateur. À gauche : un nœud rond est un individu et un nœud carré est une communauté. À droite : Activité au cours du temps associée à chaque communauté du graphe biparti de gauche.

une faible activité en dehors de ces instants. Une représentation schématique de cette situation est présentée dans la figure 6.2.

Le problème est alors de définir un processus de génération pour chaque communauté. Un processus de Poisson homogène permet de générer des temps inter-contacts suivant une loi exponentielle de paramètre λ . Cela implique que des liens apparaissent toujours avec une moyenne de λ liens par unité de temps. Or, cette hypothèse d'activité constante au cours du temps n'est pas cohérente avec notre modèle. Par exemple, les réunions de familles sont souvent calmes au début puis deviennent plus animées par la suite. Il faut donc pouvoir faire varier l'intensité de l'activité au cours du temps. Si λ varie, alors un processus de Poisson non-homogène est simulé. C'est pourquoi, nous considérons des processus de Poisson non-homogènes pour générer des liens au sein d'une communauté.

Avec $N(t)$ représentant le nombre de liens existants dans l'intervalle $[\alpha, t]$, le nombre de liens apparaissant entre deux instants, a et b , suit alors la distribution suivante :

$$P[N(b) - N(a) = k] = \frac{e^{-\Lambda_{a,b}} (\Lambda_{a,b})^k}{k!} \quad k = 0, 1, \dots, \quad (6.1)$$

où $\Lambda_{a,b} = \int_a^b \lambda(t) dt$. Par conséquent la probabilité qu'il y ait un et un seul lien ($k = 1$) dans l'intervalle suit une loi exponentielle de paramètre $\Lambda_{a,b}$: $P[N(b) - N(a) = 1] = \Lambda_{a,b} e^{-\Lambda_{a,b}}$. La fonction de répartition associée est $P[N(b) - N(a) \leq x] = 1 - e^{-\Lambda_{a,b} x}$. Pour des valeurs de a et x fixées, cette fonction de répartition est une fonction continue, croissante avec b et elle est injective. Cette fonction est également surjective si elle est strictement croissante, c'est à dire si $\Lambda_{a,b}$ est strictement croissant.

Le processus pour générer un ensemble de liens entre deux nœuds dans une communauté est le suivant. La génération commence à l'instant $t_0 = \alpha$ puis il est nécessaire de générer t_1 , l'instant d'apparition du prochain lien. Cet instant est le premier instant t' tel que $N(t') - N(t_0) = 1$ et il est relié à la probabilité suivante : $P[N(t_1) - N(t_0) \leq 1] = 1 - e^{-\Lambda_{t_0, t_1}}$. Or, cette application est bijective. Il est donc possible de retrouver

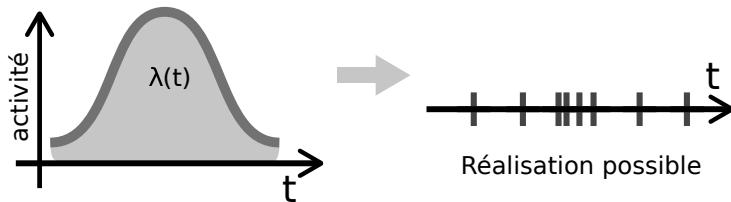


FIGURE 6.3 – Exemple des temps d'activation du lien entre deux nœuds donnés.

à partir d'une probabilité l'instant lié à cette probabilité. Il s'agit d'une méthode d'inversion. Ainsi pour générer t_1 , on génère une valeur aléatoire uniformément répartie entre 0 et 1 puis il suffit de retrouver l'instant lié à cette probabilité. Comme $\lambda(t)$ peut être quelconque, le calcul de l'instant lié à cette probabilité se fait via une recherche dichotomique. C'est à dire qu'un instant d'apparition de lien est choisi au hasard puis modifié pour se rapprocher de la probabilité voulue. Pour générer les apparitions de liens suivantes, il suffit de recommencer ce processus à partir de t_1 et de chercher l'instant t_2 tel que $N(t_2) - N(t_1) = 1$. Un exemple de génération de temps d'apparition est présenté dans la figure 6.3.

En répétant le processus ci-dessus pour chaque paire de nœuds appartenant à une même communauté, on obtient un flot de liens avec une structure communautaire sur les liens.

6.2.1 Caractéristiques du générateur

Nous discutons maintenant des caractéristiques de ce générateur. Tout d'abord au sein d'une communauté, il est possible de représenter énormément d'activités temporelles différentes. Une intensification (resp. baisse) des interactions au cours du temps est représentée par une activité croissante (resp. décroissante). Il est possible de représenter des réunions cycliques par une activité périodique. Ce modèle d'activation des liens reste cependant une approximation car il n'y a aucune prise en compte des possibles corrélations au sein d'une communauté. De même, les paires de nœuds d'une communauté correspondent des processus indépendants et identiquement distribués. Il n'y a donc au sein d'une communauté aucune différence entre les nœuds. Il n'est, de plus, pas possible de représenter l'intégration progressive de nœuds dans une communauté.

Vis-à-vis de l'ensemble des communautés, le modèle est encore une fois assez libre car un nœud peut appartenir à plusieurs communautés. Chaque communauté est traitée de manière indépendante et une même paire de nœuds peut donner lieu à la génération de liens de différentes communautés. Ainsi même si les nœuds sont équivalents au sein d'une communauté, il existe tout de même une hétérogénéité des nœuds en fonction du nombre de communautés auxquels ils appartiennent.

Avec cette formulation, le générateur dépend du graphe biparti d'affiliation des nœuds aux communautés et de l'activité temporelle de chaque communauté. L'approche que nous suivons est purement générative. Le but n'est donc pas d'inférer ces paramètres sur un exemple mais de pouvoir les fixer manuellement. Afin de faciliter l'utilisation de ce générateur, nous utilisons différentes méthodes pour fixer ces nombreux paramètres dans la sous-section 6.3.2.

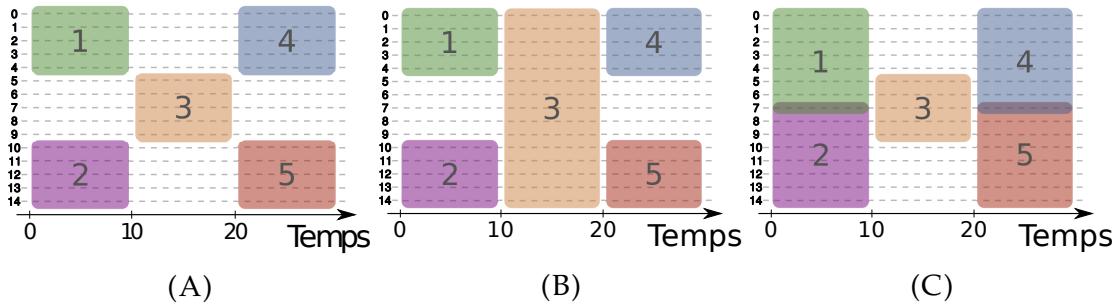


FIGURE 6.4 – Représentation schématique de trois types de flot de liens ayant une structure communautaire avec cinq communautés. Chaque rectangle de couleur représente une communauté ayant une activité constante.

6.3 Applications

6.3.1 Étude de méthodes statiques appliquées aux flots de liens

Lors de la génération de graphes statiques avec structure communautaire chevauchante, il y a principalement deux critères impactant une réalisation : le chevauchement de la structure et le rapport entre densité intérieure et extérieure à une communauté. Dans le cas des flots de liens, il faut aussi ajouter le chevauchement temporel. Afin de faciliter l'étude, nous ne générerons aucun lien en dehors des communautés définies. C'est pourquoi les différentes situations dépendent uniquement du recouvrement topologique et temporel des communautés.

Les flots de liens que nous générerons contiennent 15 nœuds pendant 30 unités de temps et il y existe cinq communautés ayant chacune une durée de 10 unités de temps. Afin de simplifier la méthode de test, chaque communauté a une activité constante de 0.5 durant l'intervalle où elle existe. Avec ces paramètres fixés, nous faisons varier le chevauchement topologique de trois manières différentes qui sont représentées dans les figures suivantes :

- 6.4A** chaque communauté est bien séparée des quatre autres. Il s'agit de l'exemple le plus simple car il n'y a aucun chevauchement topologique des communautés ;
- 6.4B** il existe un chevauchement topologique entre la communauté 3 et les autres ;
- 6.4C** il y a, en plus du chevauchement avec la communauté 3, un faible chevauchement topologique entre les communautés 1 et 2 et entre les communautés 4 et 5.

Pour ces trois types de flots de liens, nous avons appliqué, Louvain, une méthode d'optimisation de modularité [BGLL08] et CarOp, une méthode égo-centrée [DGG12] sur une projection du flot de liens en un graphe statique. La méthode CarOp permet de trouver une communauté égo-centrée à partir d'un nœud du graphe. Il est donc nécessaire de fournir un lien initial qui servira à retrouver les communautés générées. Nous choisissons pour chaque communauté générée un lien apparaissant au milieu de sa durée de vie ; par exemple un lien à l'instant 15 est choisi pour initialiser une communauté qui sera *a priori* proche de la communauté 3.

Nous avons défini précédemment une projection du flot de liens en un graphe statique mais uniquement dans le cas où les liens ont une durée. Dans les flots de liens générés ici, il n'y a pas de durée sur les liens. C'est pourquoi nous utilisons une projection différente qui transforme le flot de liens en un graphe statique pondéré. Chaque lien du flot est toujours représenté par un noeud dans le graphe statique et il y a un lien entre deux noeuds du graphe si les liens correspondants ont un noeud en commun. Afin de tenir compte de la séparation temporelle, un lien dans le graphe statique est pondéré par une fonction décroissante du temps séparant les deux liens dans le flot. Nous avons choisi comme fonction décroissante : $e^{-\lambda d}$ où d est le temps séparant les deux liens et λ une constante positive. Ainsi, les poids dans la projection sont répartis entre 0 et 1. Plus λ a une valeur importante, plus le poids d'un lien est faible. Par exemple pour $\lambda = 3$, une séparation temporelle de 1 unité de temps donne lieu à poids de 0.05 dans la projection. Dans cette situation, le graphe de la projection ne capture plus vraiment de relations topologiques car un lien n'est alors relié avec un poids non négligeable qu'avec ses voisins les plus proches. Dans le cas inverse quand λ est égal à zéro, la projection du flot de liens est complètement identique au *line-graph* du multigraph agrégé. Dans la suite, nous avons à chaque fois fait varier λ de manière exhaustive.

Lors de nos tests, nous nous sommes appuyés sur notre outil de visualisation pour comparer visuellement³ la vérité de terrain, les partitions trouvées par Louvain et CarOp selon le λ choisi.

Dans la situation la plus simple représentée dans la figure 6.4A, nous observons que les deux méthodes sont capables de retrouver la vérité de terrain.

Dans la situation représentée dans la figure 6.4B, la vérité de terrain est retrouvée par CarOp mais non par Louvain. Dans les partitions trouvées par Louvain, la communauté 3 n'est jamais détectée peu importe la valeur de λ utilisée pour la projection. Selon la valeur de λ , les liens de la communauté 3 sont soit distribués entre les autres communautés soit répartis en plusieurs communautés autonomes. Comme cette méthode optimise la modularité sur une structure proche du *line-graph* d'un multigraph, elle est similaire aux méthodes proposées par Evans *et al.* [EL09] que nous avons présentées dans la section 5.1. Elle souffre donc des mêmes biais⁴.

Dans la situation représentée dans la figure 6.4C, la vérité de terrain est retrouvée par Louvain mais non par CarOp. Comme les communautés sont petites, la méthode de Louvain arrive à capturer les différentes communautés. La méthode CarOp, quant à elle, n'est pas capable de différencier les communautés 1 et 2. Ces deux communautés sont fusionnées en une seule par CarOp. CarOp utilise la proximité des liens dans la projection. Or, un lien au début de la communauté 1 est plus proche des liens au débuts de la communauté 2 que des liens à la fin de la communauté 1. C'est pourquoi CarOp ne différencie pas les communautés 1 et 2.

3. Une évaluation plus quantitative avec la *NMI* est possible mais ne semble pas nécessaire à cette étape.

4. Plus une clique d'un graphe est grande, moins sa transformation dans le *line-graph* est dense.

Résumé

Il s'agit de premiers tests sur des méthodes existantes de détection de communautés sur des projections du flot de liens en graphes pondérés. Les méthodes que nous avons testées ne permettent pas de capturer la vérité de terrain dans l'ensemble des situations générées. Ce fait est attendu car nous les appliquons sur des projections du flot de liens. Il est toutefois intéressant de noter qu'il est possible de retrouver la vérité de terrain par une méthode statique sous certaines conditions qu'il reste encore à pleinement comprendre.

6.3.2 Vers des fonctions de qualité dans les flots de liens

Les approches statiques ne permettent pas toujours de capturer la vérité de terrain. Nous faisons donc une première proposition de fonction de qualité évaluant les partitions de liens des flots de liens et nous nous appuyons sur notre générateur et des partitions trouvées empiriquement pour étudier les caractéristiques de cette fonction de qualité.

Jeux de paramètres pour la génération de flots de liens

Le générateur dépend de beaucoup de paramètres qui sont difficiles et fastidieux à choisir manuellement pour de grands exemples. Il faut définir un graphe biparti d'affiliation des individus aux communautés et l'activité temporelle de chaque communauté. Afin de simplifier ce processus, il est possible de s'appuyer sur un générateur de graphe biparti.

De manière analogue au graphe, un graphe biparti peut être généré de telle sorte qu'il ait une densité donnée ou qu'il ait une distribution des degrés particulière. Nous faisons le choix de générer des graphes bipartis où le degré moyen des noeuds est fixé et où chaque individu est relié à au moins une communauté. Ainsi, un degré moyen de 1 implique qu'il n'y a aucun chevauchement entre les communautés. Un degré moyen de 1.8 implique qu'un noeud d'une communauté est relié en moyenne à 0.8 autres communautés et donc qu'une communauté de taille k partage des noeuds avec au maximum $0.8k$ autres communautés.

Pour l'activité des communautés, nous imposons la même activité constante et la même durée d'activité pour toutes les communautés. Les communautés se différencient ainsi par le moment où cet intervalle commence. Il suffit de décider du moment où l'intervalle débute pour définir l'activité d'une communauté. Pour une durée fixe Δ , nous choisissons de tirer de manière uniforme le temps de début d'une communauté dans $[\alpha, \omega - \Delta]$. Ainsi, plus le flot de liens est long par rapport à la durée d'activité des communautés, moins il y aura de chevauchement temporel.

Pour générer un flot de liens, le nombre de paramètres est maintenant plus restreint. Il suffit de fixer manuellement le nombre de noeuds, le nombre de communautés, le degré moyen dans le graphe biparti, la durée d'une communauté, l'activité d'une communauté et la durée totale du flot de liens. Les trois premiers paramètres permettent de contrôler le chevauchement topologique alors que les trois derniers permettent de contrôler le chevauchement temporel.

Proposition de fonctions de qualité

Nous l'avons évoquée à de nombreuses reprises. La modularité est une fonction de qualité évaluant une partition de nœuds, $\mathcal{V} = \{V_1, \dots, V_k\}$, d'un graphe. Une des formulations pour la calculer est la suivante :

$$Q_G(\mathcal{V}) = \sum_{i=1}^k Q_G(V_i) = \sum_{i=1}^k \left(\frac{d_{in}(V_i)}{2m} - \left(\frac{d(V_i)}{2m} \right)^2 \right). \quad (6.2)$$

Pour rappel, $d(V_i)$ est la somme des degrés des nœuds de V_i et $d_{in}(V_i)$ est la somme des degrés des nœuds de V_i lorsqu'uniquement les liens inclus dans $V_i \times V_i$ sont considérés. Il est possible de généraliser la modularité aux flots de liens en manipulant une partition des nœuds à chaque instant. Cette partition temporelle, $\mathcal{V}(t)$, doit respecter les contraintes d'une partition pour tout instant $t \in [\alpha, \omega]$. Avec ces notations, on peut calculer la modularité moyenne de $\mathcal{V}(t)$ sur un flot de liens L :

$$Q_L(\mathcal{V}) = \frac{1}{\omega - \alpha} \int_{\alpha}^{\omega} Q_{G(L_t)}(\mathcal{V}(t)) dt. \quad (6.3)$$

Cette formulation met en avant l'intégrale de la modularité du graphe mais la modularité, elle même, est une somme sur les groupes de nœuds. Nous transformons donc l'équation précédente pour mettre en avant la modularité d'un groupe de nœuds sur un intervalle de temps. Si dans la partition temporelle un groupe de nœud V' est une communauté sur un intervalle $[\beta, \psi]$ alors sa qualité, $Q_L(V', \beta, \psi)$, peut se calculer de la manière suivante :

$$Q_L(V', \beta, \psi) = \int_{\beta}^{\psi} \frac{d_{in}(t, V')}{d(t, V)} - \left(\frac{d(V', t)}{d(V, t)} \right)^2 dt. \quad (6.4)$$

$d(t, V')$ et $d_{in}(t, V')$ sont définis dans la section 2.3 et sont respectivement la somme des degrés temporels des nœuds de V_i et la somme des degrés temporels des nœuds de V_i lorsqu'uniquement les liens inclus dans $V_i \times V_i$ sont considérés. Avec cette formulation, on voit apparaître le degré temporel dans le calcul de cette version temporelle de la modularité d'un groupe. La vérité de terrain générée n'est cependant pas une partition temporelle des nœuds mais une partition de liens du flot de liens. Il faut donc adapter la formulation précédente pour manipuler des groupes de liens au lieu de groupes de nœuds sur un intervalle de temps. Ainsi, nous définissons la modularité d'un groupe de liens C_i :

$$Q_L(C_i) = \int_{\beta(C_i)}^{\psi(C_i)} \frac{d(t, C_i)}{d(t, V)} - \left(\frac{d(t, V(C_i))}{d(t, V)} \right)^2 dt. \quad (6.5)$$

Il s'agit de l'intégrale de la différence entre la probabilité qu'un lien appartienne à C_i et la probabilité qu'il existe un lien entre les nœuds induits par C_i . La qualité d'une partition de liens \mathcal{C} :

$$Q_L(\mathcal{C}) = \frac{1}{\omega - \alpha} \sum_{C_i \in \mathcal{C}} Q_L(C_i). \quad (6.6)$$

Cependant en considérant un groupe de liens au lieu d'un groupe de nœuds, la modularité n'est plus bornée entre -1 et 1 . La modularité classique est bornée entre

-1 et 1 et par conséquent la version temporelle de la modularité de l'équation 6.3 est également bornée car il s'agit de la moyenne de la modularité au cours du temps. La modularité temporelle de groupes de liens de l'équation 6.6 n'est pas bornée. En effet, cette formulation utilise le degré temporel des noeuds induits ce qui implique qu'un noeud peut être considéré à plusieurs reprises par différent groupes de liens. C'est le cas lorsqu'un noeud a des liens appartenant à plusieurs groupes différents.

Même si cette version n'est pas bornée, elle permet tout de même d'évaluer une partition de liens en comparant la proportions de liens appartenant au groupe à la proportion attendue dans le modèle de configuration à chaque instant. De plus, la partition contenant l'ensemble des liens obtient également une qualité nulle. Elle semble donc être une bonne candidate pour évaluer une partition de liens.

Résultats

Avec le générateur d'une part et la modularité temporelle d'autre part, nous avons procédé de manière analogue à ce que nous avons fait pour *Expected Nodes* dans la section 5.3. Nous générerons plusieurs types de flots de liens en contrôlant le chevauchement temporel et topologique. Pour chaque flot de liens généré, nous comparons la qualité de trois partitions de liens : la vérité de terrain (V), la partition trouvée par la méthode de Louvain sur la projection pondérée du flot de liens (Lo)⁵ et la partition triviale où chaque lien est dans sa propre communauté (S). La partition Lo est obtenue de la même manière que dans la section 6.3.1, c'est-à-dire en gardant la meilleure partition parmi celles trouvées via différentes valeurs de λ choisies pour la projection. Nous n'utilisons pas CarOp car elle ne retourne pas directement une partition de liens mais une couverture de liens.

Les liens du flot de liens généré n'ont pas de durée. Or, pour utiliser le degré temporel, il est nécessaire que les liens aient une durée. Nous ajoutons donc une durée Δ à chaque lien ($\xi(L, \Delta)$) et il est alors nécessaire de simplifier le flot de liens résultant ($\sigma(L)$). La simplification d'un flot de liens revient à remplacer deux liens (b, e, u, v) et (b', e', u, v) se chevauchant temporellement par un lien $(\min(b, b'), \max(e, e'), u, v)$. Cela ne pose pas de problème lorsque ces liens appartiennent à la même communauté. C'est est problématique lorsqu'ils n'appartiennent pas à la même communauté. Cette situation est illustrée dans la figure 6.5.

Il est donc nécessaire de différencier la simplification lorsque l'on s'intéresse au degré d'une communauté en particulier ou au degré du flot de liens en général. Dans le premier cas, on étudie $\sigma(\xi(L(C_i), \Delta))$, le sou-flot induit par les liens de la communauté et dans le second on étudie $\sigma(\xi(L, \Delta))$. Ces changements, bien que lourds, sont nécessaires pour rendre simple le flot de liens et pouvoir calculer le degré temporel d'une communauté, d'un ensemble de noeuds et du flot de liens. Afin de tester plusieurs Δ rapidement, nous nous sommes appuyés sur l'outil *parallel* [Tan11].

Pour la génération, nous considérons 100 noeuds entre 10 communautés durant chacune 20 unité de temps avec une activité constante de 0.1. Avec une activité de 0.1, il y a en moyenne deux liens appartenant à la communauté pour chaque paire de noeuds. Nous choisissons une durée totale du flot de liens de 50 ou 200 pour contrôler le chevauchement temporel. Nous choisissons un degré moyen dans le biparti de 1.1 ou 2

5. Nous l'avons définie dans la section 6.3.1.

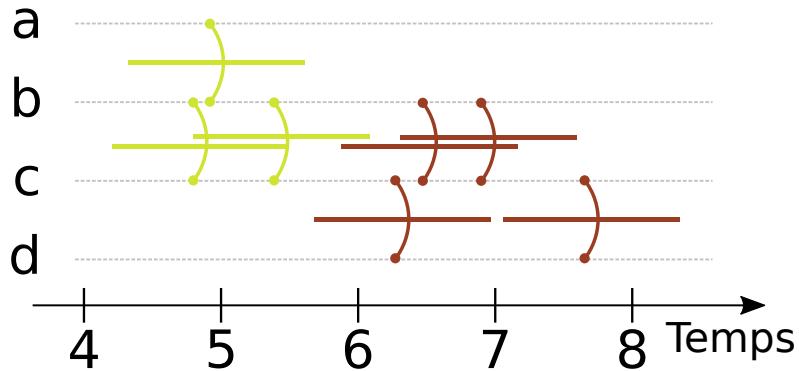


FIGURE 6.5 – Flot de liens avec deux communautés en vert et rouge où chaque lien a une durée égal à 1.5. La simplification de ce flot de liens est problématique pour le lien entre b et c à l'instant 6.

pour contrôler le chevauchement temporel. Les flots de liens ainsi générés ont entre 1200 et 4000 liens selon le chevauchement choisi.

Les valeurs de qualité pour chaque situation et chaque partition en fonction de la durée des liens sont présentées dans la figure 6.6. On remarque également que la vérité de terrain est la meilleure partition peu importe le cas de figure lorsque Δ est supérieur à 2. Lorsqu'il y a un important chevauchement topologique, figure 6.6C et 6.6D, la vérité de terrain a toutefois une qualité plus faible. C'est en particulier visible dans la figure 6.6C où le chevauchement temporel est également important. Dans cette situation, la qualité obtenue par la vérité de terrain est négative alors que la partition contenant l'ensemble des liens a une qualité nulle. Avoir une qualité négative dans ce cas de figure n'est toute fois pas complètement surprenant car la structure est difficilement discernable avec des chevauchements si importants.

Lorsque Δ tend vers 0, la qualité des partitions tend également vers 0 car le degré moyen tend vers 0. Par ailleurs, nous avons remarqué dans la section 6.3.1 que la méthode de Louvain est parfois à même de retrouver la vérité de terrain, notamment lorsque le chevauchement est faible. Il y a très peu de chevauchement dans la situation représentée dans la figure 6.6B et nous observons que la partition Lo est identique à la vérité de terrain pour ce cas particulier. Enfin, la partition où chaque lien est dans sa propre communauté obtient toujours une qualité élevée lorsque Δ est compris entre 10^{-2} et 2.

Ces observations permettent de tirer des premières conclusions sur cette fonction de qualité. Tout d'abord, la durée des liens a un impact fort sur l'évaluation d'une partition. Par exemple lorsque la durée est courte, notre fonction de qualité n'évalue pas la vérité de terrain comme la meilleure partition. Ce phénomène pourrait être dû au fait qu'il n'existe que très peu de liens présents simultanément lorsque Δ est faible.

Le chevauchement topologique semble également avoir un impact plus important sur les évaluations que le chevauchement temporel. Cet impact n'est en revanche pas le même pour toutes les partitions. Pour la vérité de terrain, le chevauchement topologique diminue la qualité alors que pour la partition où chaque lien est dans sa propre communauté le chevauchement temporel augmente la qualité.

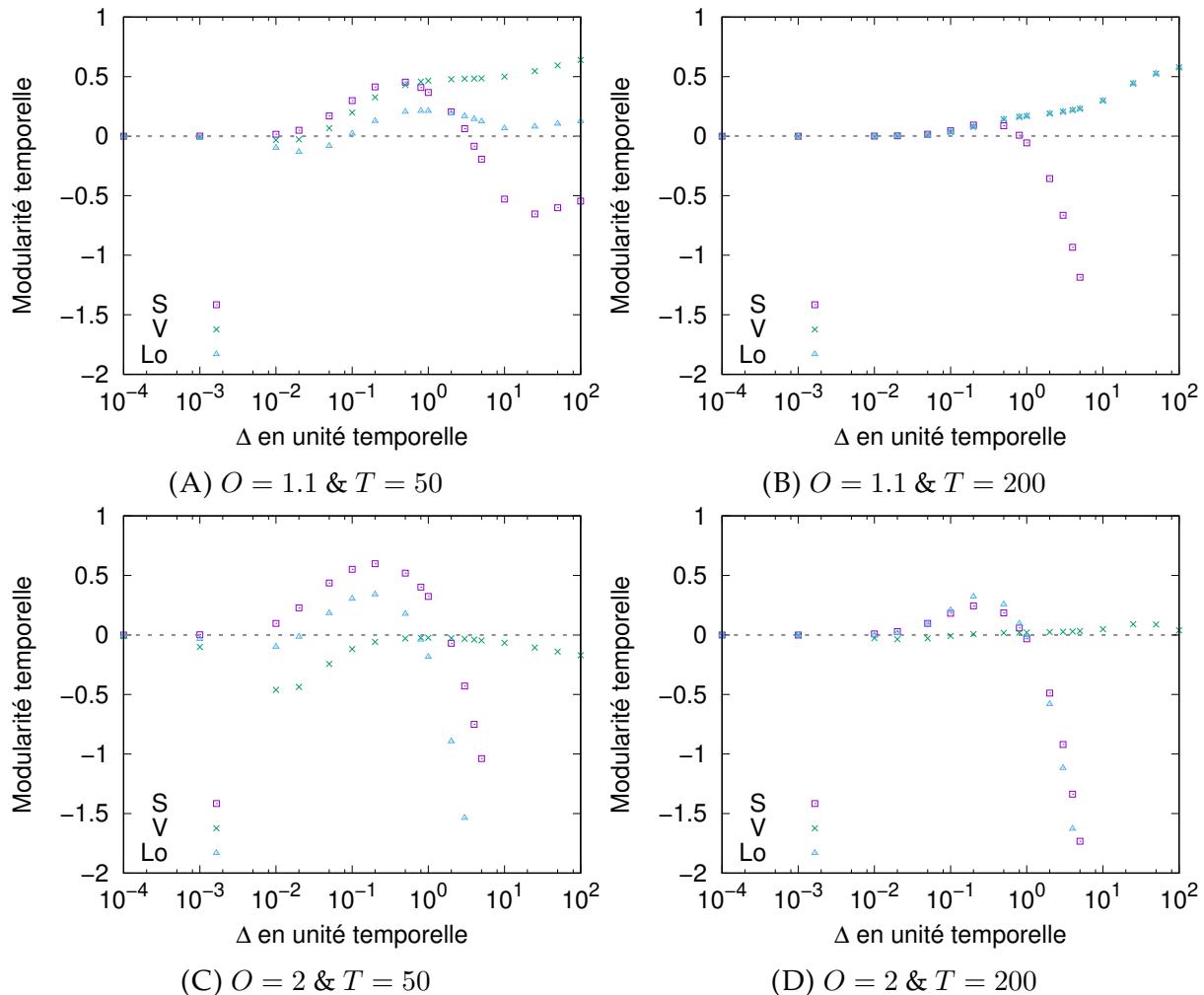


FIGURE 6.6 – Évaluation en fonction de la durée Δ ajoutée à chaque lien de la partition vérité de terrain (V), de la partition trouvée par la méthode de Louvain (Lo) et de la partition où chaque lien est dans sa propre communauté (S). Les trois figures correspondent à différents choix de chevauchement topologique (O) et temporel (T).

Résumé

La transposition de la modularité permet d'évaluer des partitions de liens d'un flot de liens. Cette fonction de qualité réussie à évaluer la vérité de terrain comme la meilleure fonction de qualité dans certaines conditions ; lorsque les liens durent suffisamment longtemps. Cependant, ces travaux ne sont que préliminaires et il est nécessaire d'étudier d'autres cas de figure avant de comprendre cette fonction de qualité. Il est en particulier important de tester d'autres partitions trouvées empiriquement, notamment avec un algorithme d'optimisation dédié.

6.4 Conclusion

Dans ce chapitre, nous nous sommes intéressés à la génération de flots de liens avec une structure communautaire sur les liens. Il s'agit de la première méthode de ce genre car les méthodes existantes se focalisent sur la génération de flot de liens

ayant des distributions des temps inter-contact similaires à celles observées dans les jeux de données réelles. Notre approche est différente. Nous modélisons l'existence de groupes de nœuds communiquant pendant des intervalles de temps donnés. Ainsi, il est possible de simuler des événements durant lesquels des liens apparaissent entre les personnes concernées par ces événements. Pour ce faire, notre méthode s'inspire du SBM. Il faut définir un graphe biparti d'affiliation des nœuds aux communautés et l'activité temporelle de chaque communauté. Une fois ces paramètres définis, des liens entre deux nœuds sont générés en fonction de processus de Poisson non-homogènes qui sont spécifiques aux communautés partagées par ces deux nœuds.

Grâce à ce générateur, nous avons pu tester deux méthodes de détection de communautés sur des projections du flot de liens en graphes statiques pondérés. Ces premiers résultats sont très intéressants car ces méthodes sont parfois à même de retrouver la vérité de terrain alors qu'elles se basent sur un graphe statique. Cependant lorsque le chevauchement topologique est important, les méthodes statiques que nous avons testées n'ont pas retrouvé la vérité de terrain. Il est donc nécessaire de proposer des méthodes pour évaluer et capturer des partitions de liens dans les flots de liens.

Nous avons donc proposé une première fonction de qualité proche de la modularité. Nous l'avons testée sur des flots de liens présentant différents chevauchement topologiques et temporels. Sur chaque flot de liens, nous avons évalué la vérité de terrain, une partition trouvée empiriquement et une partition triviale. Les résultats sont encourageants car la vérité de terrain est souvent évaluée comme la meilleure partition. Cependant, il est nécessaire d'étudier d'autres cas de figure avant de comprendre cette fonction de qualité. En particulier, il est important de tester d'autres partitions afin de s'assurer que la vérité de terrain soit proche de l'optimum.

6.4.1 Perspectives

Ces premiers travaux concernent un domaine qui n'a, à notre connaissance, jamais été étudié auparavant. Il y a donc de nombreuses poursuites pour l'amélioration de ce générateur.

Pour qu'un générateur soit utile, il doit générer des structures vraisemblables. Il est par exemple peu judicieux d'utiliser un graphe généré selon le modèle de Erdös-Rényi pour représenter un réseau social. C'est une question que nous n'avons pas abordée jusqu'à maintenant mais qui est primordiale. Le modèle de génération que nous utilisons est très général. C'est le choix des paramètres qui rend les flots de liens générés vraisemblable ou non. Il faut donc étudier comment fixer ses paramètres.

Il est par exemple possible de changer l'activité des communautés pour que leur densité et leur durée soient similaires à celles observées dans un jeu de données. De même, il serait intéressant d'évaluer quel serait un graphe d'affiliation vraisemblable dans le cas des flots de liens. Pour ce faire, les données de courriels du chapitre 3 et les groupes pertinents détectés dans le chapitre 4 peuvent être d'une grande utilité.

En plus de ces premiers aspects, il est également possible de modifier le modèle de génération pour le rendre plus flexible au prix de l'ajout d'autres paramètres. Par exemple, les liens générés n'ont pas de durée. Il serait intéressant de tirer la durée d'un lien selon une distribution donnée. Il se pose alors la question de la cohérence entre la durée d'un lien et le processus de Poisson non-homogène sous-jacent : est-il normal de générer un lien de longue durée durant une activité faible de la communauté ? Encore

une fois pour répondre à ce genre de question, il est nécessaire de s'appuyer sur des observations empiriques.

Enfin, une des hypothèses de notre générateur est l'équivalence des nœuds au sein d'une communauté. Or, il se peut que certains nœuds participent plus que d'autres. Un moyen de prendre en compte cette diversité est d'incorporer un système d'attachement préférentiel. Dans les graphes, le système préférentiel se définit de la manière suivante. Chaque nœud a un score de popularité⁶ qui détermine la propension des autres nœuds à créer un lien avec lui. Ainsi plus un nœud est populaire, plus il a de liens avec les autres nœuds. Dans le cadre de la génération de flot de liens, il serait possible que le processus de génération des liens entre deux nœuds ne dépende plus uniquement de la communauté mais également de la popularité respective des nœuds. Cette popularité, temporelle ou non, devrait également être fixée par l'utilisateur mais elle permettrait de mettre en avant un leader au sein d'une communauté.

6. Le degré est généralement utilisé comme score de popularité.

Chapitre 7

Bilan

7.1 Résumé et contributions apportées

Dans cette thèse, nous nous sommes efforcés de décrire et de capturer les structures des séquences d’interactions tout en nous basant sur une base formelle la plus solide possible. Nous avons utilisé le formalisme de flot de liens qui relie les propriétés temporelles et topologiques des séquences d’interactions. Ce formalisme est suffisamment expressif pour définir des métriques évaluant des groupes de liens sans déformer ou perdre de l’information sur leur dynamique. Les notions de degré et de densité, que nous avons présentées dans le chapitre 2, en sont des exemples. En plus d’aider à définir ces notions, nous les avons également implémentées dans une librairie de manipulation de flots de liens¹. Ainsi avec d’une part un formalisme clairement défini et d’autre part une implémentation accessible, on peut espérer faciliter l’utilisation de la densité mais surtout permettre à quiconque d’étendre ce formalisme en définissant d’autres métriques.

À partir de ce formalisme, nous avons décrit les structures existantes dans les flots de liens, dans le chapitre 3. Pour ce faire, il est nécessaire de connaître une séquence d’interactions où la structure est connue. Il n’en existait² aucune au début de cette thèse, à notre connaissance. C’est pourquoi, nous avons collecté un jeu de données regroupant les courriels d’une liste de diffusion. Ce jeu de données est très intéressant de par sa temporalité, presque 20 ans, et par sa précision, de l’ordre de la seconde. Grâce à ces informations, nous avons pu étudier la structure de discussions du point de vue de la densité. Nous avons notamment pu mettre en évidence que les discussions sont plus densément connectées en interne qu’avec le reste du flot de liens. Ces travaux ont donné lieu à une publication dans une conférence internationale [GVFS⁺16]. Suite à ce constat, nous avons essayé de retrouver automatiquement les discussions via une projection du flot de liens en un graphe statique. Bien que la partition trouvée ne corresponde pas exactement aux discussions, les groupes de la partition n’ont pas des caractéristiques triviales notamment en terme de taille et de densité.

Dans le chapitre 4, nous avons étudié plus en détail comment évaluer la pertinence de groupes de liens et nous avons utilisé les groupes détectés dans la projection du flot de liens en un graphe statique comme candidats en tant que groupes pertinents. Le but est de réussir à identifier les groupes pertinents parmi les groupes de cette partition. Cette identification est nécessaire car il se peut que parmi tous ces groupes de liens certains ne soient pas pertinents. En effet, ils ne sont pas obtenus directement sur le

1. <https://bitbucket.org/nGaumont/liblinkstream/>

2. Un jeu de données représentant le forum *reddit* a été rendu public juillet 2015.

flot de liens mais sur une projection de ce flot en un graphe statique. Or, il y a perte d'information lorsque l'on manipule un graphe statique au lieu d'un flot de liens.

Comme nous l'avons constaté dans les chapitres 2 et 3, la densité permet d'évaluer un groupe mais, seule, elle ne suffit pas pour définir un groupe pertinent. En effet, un groupe n'est pertinent que par rapport au reste du flot de liens. Par exemple, un groupe ayant une densité de 0.4 est très pertinent dans un flot de liens ayant une densité globale de 0.01 mais il ne l'est pas du tout si le flot de liens a une densité de 0.6. Pour évaluer un groupe, il est donc nécessaire de prendre en compte le contexte dans lequel il apparaît. Nous avons défini trois voisinages constitués de sous-flots différant du sous-flot du groupe soit sur le temps de début, soit sur la durée, soit sur l'ensemble de nœuds. Un groupe est ensuite jugé pertinent si le groupe est plus dense qu'une grande partie de ses voisinages. Nous avons appliqué notre méthode sur quatre jeux de données et nous avons pu valider manuellement quelques groupes grâce aux métadonnées existantes.

La structure résultant des groupes pertinents est une partition partielle des liens du flot de liens. Cependant en observant les nœuds induits et les instants de présence des groupes, nous avons constaté que les groupes forment une structure très chevauchante sur les nœuds et uniquement partiellement chevauchante sur le temps. Nous avons testé si une méthode statique pouvait retrouver cette même structure chevauchante et nous avons constaté qu'uniquement quelques groupes pertinents étaient capturés par la méthode statique. Ceci montre que nous avons mis avant une structure nouvelle via la partition partielle que sont les groupes pertinents. Ces travaux ont donné lieu à une première publication dans une conférence nationale [Gau16] et sont soumis au journal *Social Network Analysis and Mining*.

À partir de cette structure partielle, il est intéressant d'essayer de détecter une partition complète du flot de liens. La détection de structure dans un réseau dynamique ou non nécessite un moyen d'évaluer le résultat obtenu. C'est pourquoi, nous avons étudié plus en détails la détection de communautés sous la forme de partitions de liens et ce dans le cas statique et dynamique. Comme nous l'avons vu dans le chapitre 1, il existe assez peu de méthodes de détection de communautés sous la forme de partitions de liens et aucune ne s'est vraiment attachée à reproduire ce qui a fait le succès de la modularité. C'est pourquoi, dans le chapitre 5, nous avons proposé une nouvelle méthode pour l'évaluation d'une partition de liens dans le cas statique qui se base également sur la notion de modèle nul. Un groupe de liens est ainsi une bonne communauté s'il induit moins de nœuds qu'attendu dans le modèle nul. De plus, un groupe de liens est également évalué en fonction de son voisinage. Cette fois ci, un voisinage est jugé bon lorsqu'il consiste en moins de nœuds adjacents qu'attendu dans le modèle nul car le voisinage doit être peu dense.

Nous avons mené des tests sur des graphes ayant une structure simple : une clique ou deux cliques se chevauchant et sur des graphes générés aléatoirement. Sur ces exemples, notre fonction de qualité, *Expected Nodes*, est la seule à différencier la vérité de terrain d'autres partitions moins pertinentes. Ces travaux ont été publiés dans une conférence nationale [GQ14] et une conférence internationale [GQML15]. Fort de ces premiers constats, nous avons implémenté un premier algorithme d'optimisation d'*Expected Nodes*. Cependant malgré nos efforts pour rendre plus efficace l'évaluation d'*Expected Nodes*, notre algorithme n'est pas adapté à l'étude de grands graphes.

Fort de nos observations dans le cas dynamique et de notre proposition dans le cas statique, nous avons considéré les partitions de liens dans les flots de liens dans le chapitre 6. Pour ce faire, nous nous sommes intéressé à la génération de flots de liens avec une structure donnée et à la manière d'évaluer une telle partition. Pour générer un flot de liens, nous nous inspirons du *Stochastic Block Model*. Ainsi, les liens entre deux noeuds sont générés dans le cadre d'une communauté commune et le temps d'attente entre deux apparitions d'un lien dépend de l'activité de cette communauté. Ainsi, il est possible de générer des flots de liens très variés tout en ayant une vérité de terrain sur les liens.

À partir de ce mécanisme de génération, nous avons dans un premier temps mis en avant l'inefficacité de certaines méthodes statiques dans une projection du flot de liens à retrouver la structure générée. Dans un second temps, nous avons esquissé de premiers travaux étudiant des extensions de la modularité dans les flots de liens. Ces résultats nous ont permis d'entrevoir de nombreuses possibilités pour définir une fonction de qualité dans les flots de liens.

7.2 Perspectives

L'ensemble de ces travaux ouvre de nombreuses pistes de recherche que nous avons déjà évoquées de manière individuelle à la fin de chaque chapitre. De manière plus générale, il y a deux pistes distinctes aux travaux de cette thèse : l'extension du formalisme de flot de liens et l'approfondissement des métriques et applications dans les flots de liens.

7.2.1 Extensions du formalisme de flot de liens

Le formalisme de flot de liens est assez général pour représenter beaucoup de situations différentes mais il y en a cependant certaines qui ne sont pas parfaitement représentées par un flot de liens classique. Il s'agit d'extensions similaires à celles existantes dans les graphes. Il serait intéressant de considérer l'orientation des liens et les flots de liens bipartis. Ces ajouts peuvent se faire de manière assez simple : respectivement $(b, e, u, v) \neq (b, e, v, u)$ et $\forall (b, e, u, v) \in E, partie(u) \neq partie(v)$ avec $partie(u) \in \{1, 2\}$. Il est alors seulement nécessaire d'adapter les notions de degré et de densité pour tenir compte de ces spécificités.

L'ajout de poids sur les liens est bien sûr une extension possible mais il y a cependant deux extensions possibles : soit le poids est fixe ; soit il peut changer au cours du temps. Le second cas englobe le premier et permet de représenter beaucoup plus de situations comme par exemple les flots de liens non simples, *i.e.* plusieurs liens peuvent exister en même temps entre deux noeuds.

Nous avons principalement discuté des notions de degré et de densité dans cette thèse. Il existe bien évidemment d'autres notions qui ont déjà été définies dans un cadre temporel comme les notions de chemin et de centralité. Il serait intéressant d'intégrer ces notions dans le formalisme de flots de liens afin de rendre l'ensemble des notions cohérentes entre elles. Par exemple, il serait intéressant de définir une mesure de centralité à partir du degré temporel. On pourrait comparer le degré temporel d'un noeuds vis-à-vis du degré temporel moyen de ses voisins afin de définir une forme de centralité locale.

En plus de l'extension des métriques de la théorie des graphes, il est intéressant de considérer d'autres types de mesures. En effet, les flots de liens regroupent de l'information structurelle et temporelle. Or avec les mesures de la théorie des graphes, nous apportons principalement de l'information structurelle. À l'opposé de ce processus, le traitement du signal est focalisé sur l'information temporelle. Il serait très intéressant de tirer parti de ce domaine pour enrichir les méthodes d'analyses des flots de liens. Il est par exemple possible de considérer un flot de liens comme un signal. Il serait alors intéressant de comprendre ce que signifie une transformée de Fourier dans les flots de liens.

Beaucoup de ces pistes sont d'ailleurs explorées dans la thèse de Tiphaine Viard [Via16].

7.2.2 Approfondissement des applications dans les flots de liens

Dans cette thèse, nous avons étudié les groupes de liens comme structure communautaire. Cette notion, bien que très spécifique, mène à de nombreuses perspectives sur la notion de communauté et la génération de flots de liens, comme nous l'avons évoqué à la fin du chapitre 4 et dans le chapitre 6. Mais il y a bien d'autres pistes.

Il serait, par exemple, intéressant de comparer des groupes de liens. En effet, deux groupes de liens distincts peuvent partager exactement les mêmes noeuds et apparaître dans le même intervalle et donc être similaires. Le simple indice de jaccard n'est donc pas suffisant pour comparer deux groupes de liens. Il faut donc une similarité tirant parti de la topologie et du temps. Si une telle relation existe, alors il serait très intéressant d'étudier le graphe des similarités entre les groupes pertinents. Il s'agirait d'un graphe orienté par le temps et pondéré par la similarité entre les groupes. On peut espérer trouver dans un tel graphe les groupes de liens reliant des noeuds similaires. Il serait aussi sûrement possible de faire un parallèle entre ce graphe et les évolutions des communautés dans une série de graphes. Par exemple, un noeud ayant deux fils (resp. parents) représenterait alors la séparation d'une communauté en deux (resp. la fusion de deux communautés).

Par ailleurs dans le chapitre 6, nous avons tenté de définir une fonction de qualité globale pour évaluer une partition de lien. Il serait intéressant de poursuivre la piste inverse et d'essayer de définir une fonction de qualité locale. Le but serait alors d'avoir une fonction ne prenant en compte que le voisinage immédiat d'un groupe de liens pour l'évaluer contrairement à notre notion de pertinence qui prend en compte tout le voisinage.

Enfin, il serait intéressant d'étudier les problèmes connexes à la détection de communautés de liens. La compression de l'information en est un exemple car une structure communautaire peut être vue comme un *résumé* de l'objet d'étude initial. Avec ce point de vue, il se pose alors la question de la perte d'information induite par l'utilisation d'une structure communautaire pour représenter le flot de liens globale.

La prédiction de liens est un autre exemple. Dans un graphe, la prédiction de liens consiste en trouver les paires de noeuds qui devraient être reliées dans le graphe. Comme les noeuds d'une communauté sont densément reliés entre eux, connaître la structure communautaire aide à prédire les paires de noeuds qui devraient être reliées. Dans les flots de liens, la situation est tout de fois plus complexe car, en plus de prédire une interaction, il faut également prédire le temps d'apparition et la durée de l'interaction.

Résumé

Au cours de cette thèse, nous avons étudié le problème de la détection de communautés dans les séquences d'interactions. La prise en compte de la dynamique dans la littérature est encore récente et il n'existe aucune taxonomie faisant consensus. Chaque proposition nécessite ainsi de redéfinir ses propres notions ce qui se fait au détriment de la liaison entre les concepts existants. Il s'agit d'un frein pour le développement de méthodes de détection de communautés dans les séquences d'interactions. Plutôt que de résoudre la détection de communautés, les travaux de cette thèse permettent surtout de mieux définir le problème. Il est ainsi important de comprendre les différences entre les formalismes de série de graphes, de graphe temporels et de flot de liens. À partir de ce premier choix, il devient alors possible d'essayer de décrire une communauté dans ce contexte. Nous avons le parti pris de considérer le formalisme de flots de liens et les partitions de liens comme structures communautaires. De ce présupposé, nous avons défini plusieurs métriques et décrit une structure communautaire existante avant de proposer une méthode d'évaluation de la pertinence d'un groupe. Enfin nous avons esquissé de premiers travaux de générations de flots de liens avec une structure communautaire.

Afin de mener ces travaux liant théorie et application, nous avons constamment essayé de partager les résultats obtenus : les données récoltées, la théorie proposée et l'implémentation développée. Avec cette démarche, nous espérons ainsi faciliter les échanges entre chercheurs pour faire avancer la recherche dans ce domaine où encore beaucoup reste à faire tant au niveau théorique que pratique.

Annexe A

Détection de groupes pertinents dans les flots de liens

A.1 Rollernet dataset

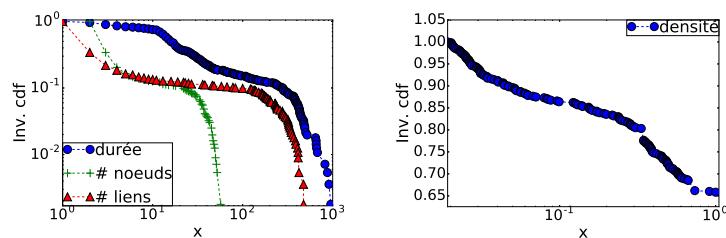
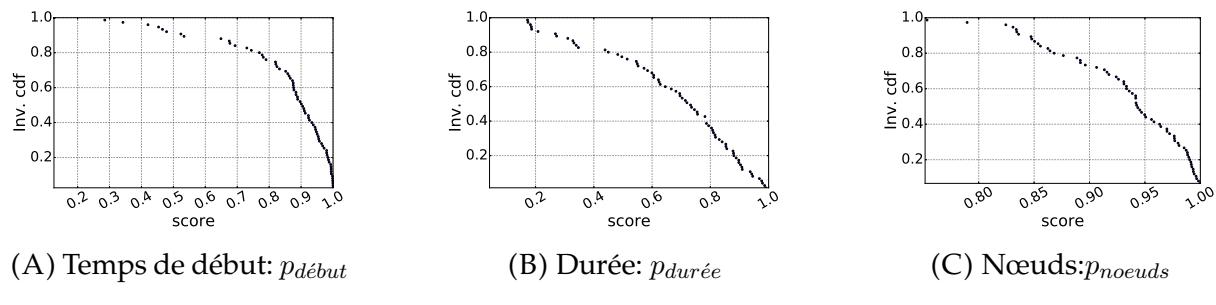


FIGURE A.1 – Distribution cumulative inverse du nombre de liens, de nœuds et de la durée en (A) et de la densité en (B) pour les candidats trouvés par la méthode de Louvain dans le jeu de données Rollernet.



(A) Temps de début: $p_{\text{début}}$

(B) Durée: $p_{\text{durée}}$

(C) Nœuds: p_{noeuds}

FIGURE A.2 – Distribution cumulative inverse des scores pour chaque voisinage dans le jeu de données Rollernet.

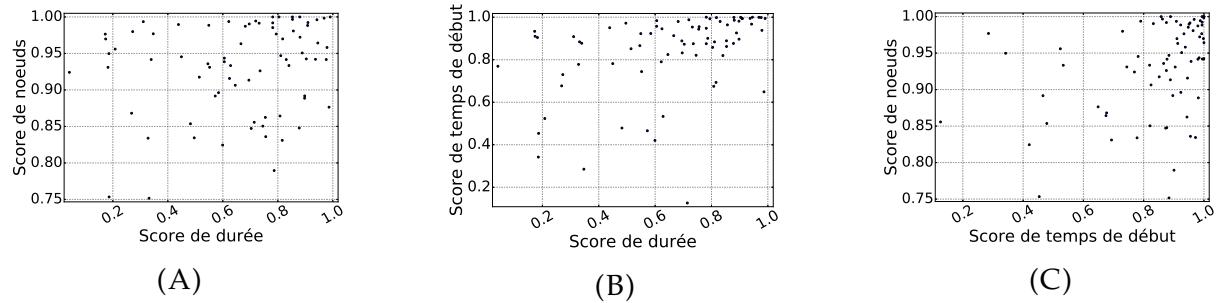


FIGURE A.3 – Corrélations des scores selon le voisinage dans le jeu de données Rollernet.

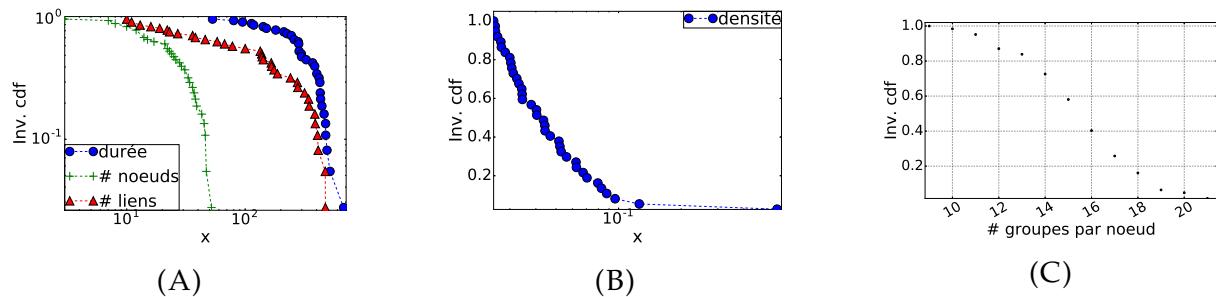


FIGURE A.4 – Distributions cumulatives inverses du nombre de liens, de noeuds et de la durée en (A), de la densité en (B), du nombre de groupes par noeud en (C) pour les groupes capturés par notre méthode dans les données Rollernet.

A.2 Baboon dataset

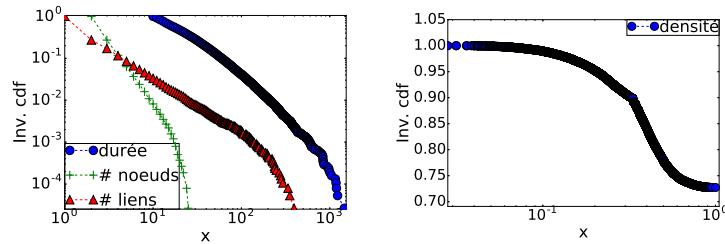
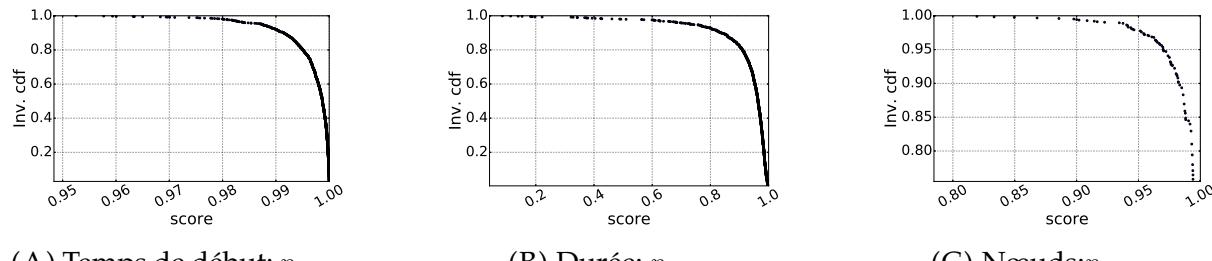


FIGURE A.5 – Distribution cumulative inverse du nombre de liens, de nœuds et de la durée en (A) et de la densité en (B) pour les candidats trouvés par la méthode de Louvain dans le jeu de données Babouins.



(A) Temps de début: $p_{\text{début}}$ (B) Durée: $p_{\text{durée}}$ (C) Nœuds: p_{noeuds}

FIGURE A.6 – Distribution cumulative inverse des scores pour chaque voisinage dans le jeu de données Babouins.

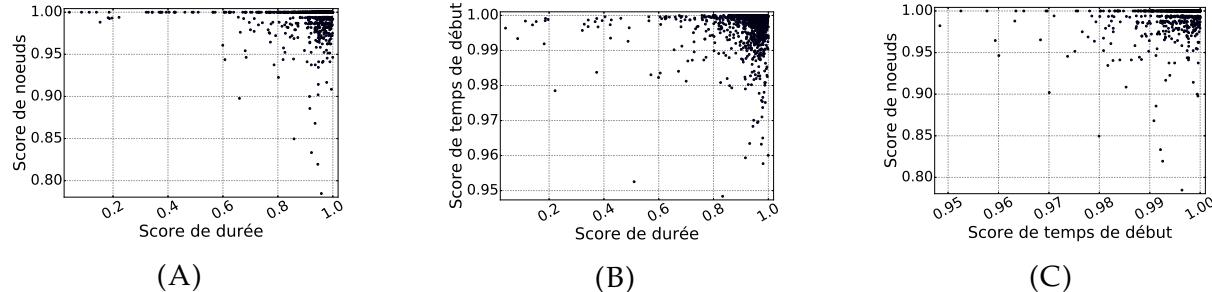


FIGURE A.7 – Corrélations des scores selon le voisinage dans le jeu de données Babouins.

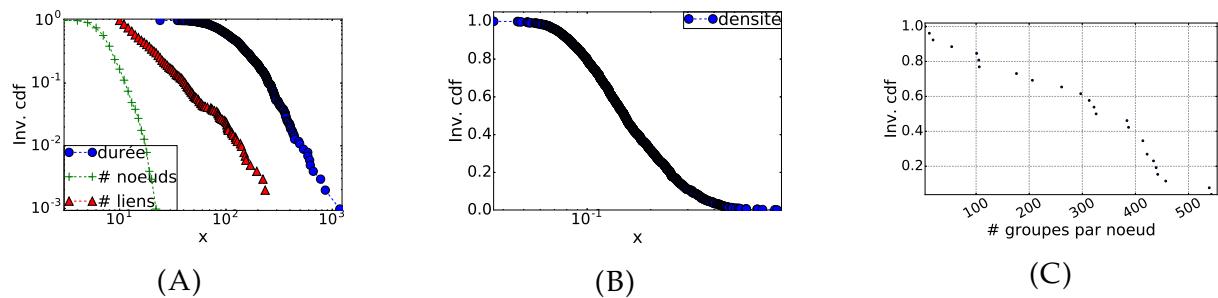


FIGURE A.8 – Distributions cumulatives inverses du nombre de liens, de nœuds et de la durée en (A), de la densité en (B), du nombre de groupes par nœud en (C) pour les groupes capturés par notre méthode dans les données Babouins.

A.3 Reality mining dataset

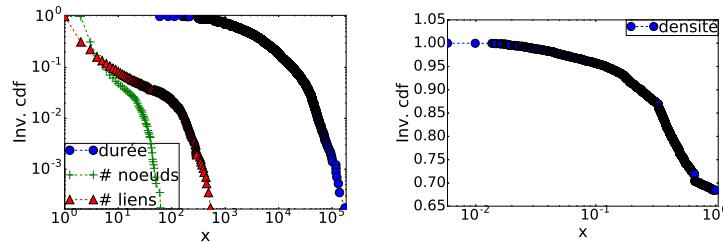


FIGURE A.9 – Distribution cumulative inverse du nombre de liens, de nœuds et de la durée en (A) et de la densité en (B) pour les candidats trouvés par la méthode de Louvain dans le jeu de données Reality Mining.

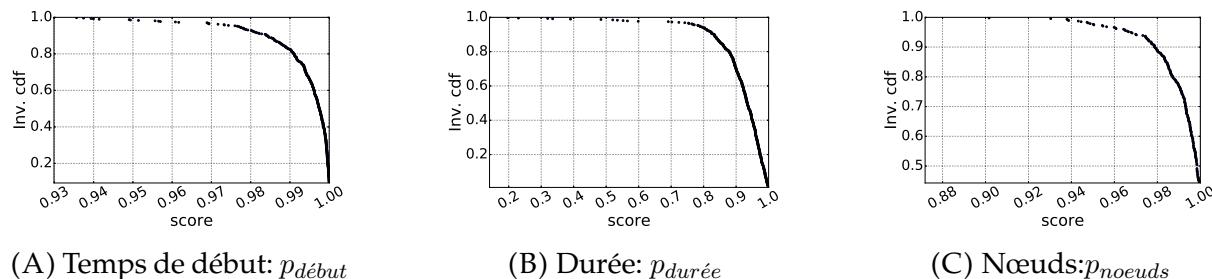


FIGURE A.10 – Distribution cumulative inverse des scores pour chaque voisinage dans le jeu de données Reality Mining.

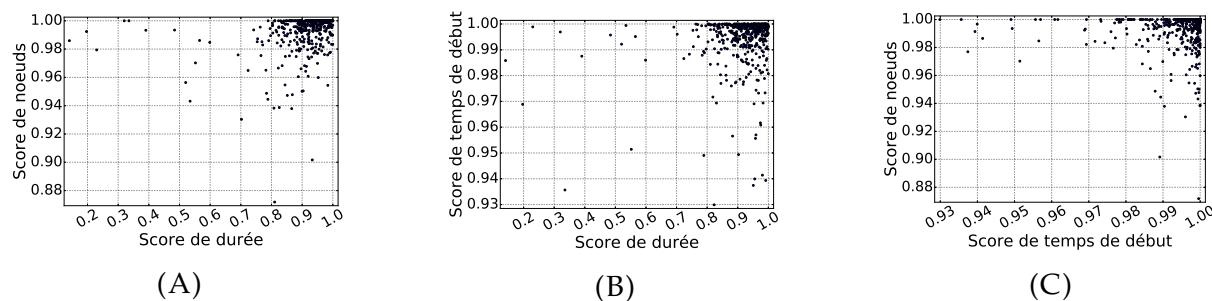


FIGURE A.11 – Corrélations des scores selon le voisinage dans le jeu de données Reality Mining.

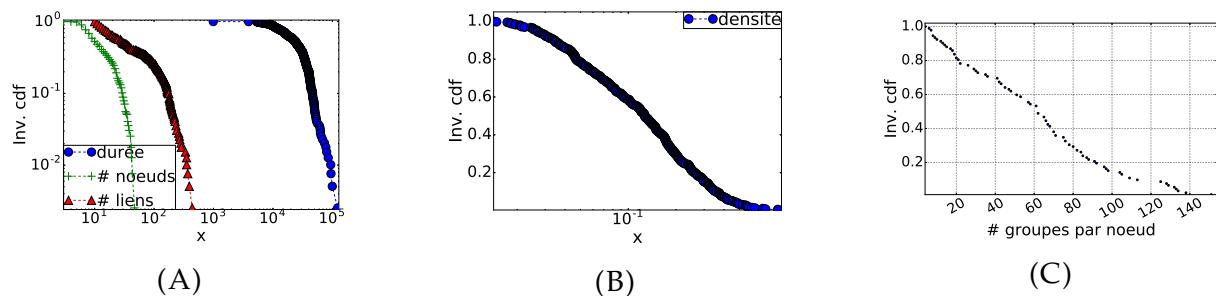


FIGURE A.12 – Distributions cumulatives inverses du nombre de liens, de nœuds et de la durée en (A), de la densité en (B), du nombre de groupes par nœud en (C) pour les groupes capturés par notre méthode dans les données Reality Mining.

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.
- [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 Sen-gupta. 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. Springer International Publishing, 2007.
- [BFLY13] Alain Barrat, Bastien Fernandez, Kevin K. Lin, and Lai-Sang Young. Modeling Temporal Networks Using Random Itineraries. *Physical Review Letters*, 110(15) :158702, apr 2013.
- [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.
- [BMS11] Petko Bogdanov, Misael Mongiovì, and Ambuj K. Singh. Mining Heavy Subgraphs in Time-Evolving Networks. In *2011 IEEE 11th International Conference on Data Mining*, pages 81–90. IEEE, dec 2011.
- [BP16] V Batagelj and S Praprotnik. An algebraic approach to temporal network analysis based on temporal quantities. *Social Network Analysis and Mining*, 2016.
- [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.
- [CKW15] Margaret C. Crofoot, Roland W. Kays, and Martin Wikelski. Data from : Shared decision-making drives collective movement in wild baboons, 2015.
- [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.
- [CMM⁺08] Andrea E.F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time in edge-Markovian dynamic graphs. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing - PODC '08*, page 213, New York, New York, USA, 2008. ACM Press.
- [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.
- [DDDGA05] Leon Danon, Jordi Duch, Albert Diaz-Guilera, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics : Theory and Experiment*, 2005(09) :10, 2005.
- [DDG⁺15] Armando Di Nardo, Michele Di Natale, Carlo Giudicianni, Dino Musmarrà, 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 Vandermarliere, 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.
- [ELS15] Alessandro Epasto, Silvio Lattanzi, and Mauro Sozio. Efficient Densest Subgraph Computation in Evolving Graphs. In *Proceedings of the 24th International Conference on World Wide Web*, pages 300–310. International World Wide Web Conferences Steering Committee, may 2015.
- [EPL09] Nathan Eagle, Alex Sandy Pentland, and David Lazer. Inferring Social Network Structure using Mobile Phone Data. *Pnas*, 106(usually 1) :15274–15278, 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.
- [FB14] Julie Fournet and Alain Barrat. Contact patterns among high school students. *PloS one*, 9(9) :e107878, jan 2014.
- [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.
- [Gau16] Noe Gaumont. Trouver des séquences de contacts pertinentes dans un flot de liens. In *ALGOTEL 2016 - 18èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, ALGOTEL 2016 - 18èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications, Bayonne, France, may 2016.
- [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. Benchmark model to assess community structure in evolving networks. *Physical Review E*, 92(1) :012805, jul 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] L Gauvin, A Panisson, A Barrat, and C Cattuto. Revealing latent factors of temporal networks for mesoscale intervention in epidemic spread. *arXiv preprint arXiv* :, 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.
- [GQ14] Noé Gaumont and François Queyroi. Partitionnement des Liens d'un Graphe : Critères et Mesures. In *ALGOTEL 2014 – 16èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2014.
- [GQML15] Noé Gaumont, François Queyroi, Clémence Magnien, and Matthieu Latapy. Expected Nodes : a quality function for the detection of link communities. In *Complex Networks VI, Studies in Computational Intelligence*. Springer International Publishing, 2015.
- [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*. Springer International Publishing, 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] P Holme. Information content of contact-pattern representations and predictability of epidemic outbreaks. *Scientific reports*, 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.
- [JPB⁺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.
- [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 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.
- [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.
- [KK14] Paul Kim and Sangwook Kim. Detecting overlapping and hierarchical communities in complex network using interaction-based edge clustering. *Physica A : Statistical Mechanics and its Applications*, sep 2014.
- [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 call sequences. *Proceedings of the National Academy of Sciences of the United States of America*, 110(45) :18070–5, 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.
- [KP15] Mikko Kivelä and Mason A. Porter. Estimating interevent time distributions from finite observation periods in communication networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 92(5), 2015.
- [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. *Physical Review E*, 80(5) :056117, 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.
- [LSK15] Guillaume Laurent, Jari Saramäki, and Márton Karsai. From calls to communities : a model for time-varying social networks. *European Physical Journal B*, 88(11) :1–10, 2015.
- [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 Proceedings WI06*, 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. *arXiv preprint arXiv:1506.07464*, 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.
- [MRV15] C Matias, T Rebafka, and F Villers. Estimation and clustering in a semiparametric Poisson process stochastic block model for longitudinal networks. *arXiv preprint arXiv:1512.07075*, 2015.
- [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.
- [MTR12] Bivas Mitra, Lionel Tabourier, and Camille Roth. Intrinsically dynamic network communities. *Computer Networks*, 56(3) :1041–1053, feb 2012.
- [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.
- [OD14] Darko Obradovic and Maximilien Danisch. Direct generation of random graphs exactly realising a prescribed degree sequence. In *2014 6th International Conference on Computational Aspects of Social Networks*, pages 1–6. IEEE, jul 2014.
- [PBV07] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136) :664–667, 2007.
- [PC13] M Plantíé 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] JI Perotti, HH Jo, P Holme, and J Saramäki. Temporal network sparsity and the slowing down of spreading. *arXiv preprint arXiv* :1411.5553, 2014.
- [PL05] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *Computer and Information Sciences (IS-CIS)*, 10 :284–293, 2005.
- [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.
- [RTG14] Polina Rozenshtein, Nikolaj Tatti, and Aristides Gionis. Discovering Dynamic Communities in Interaction Networks. In Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8725 of *Lecture Notes in Computer Science*, pages 678–693. Springer Berlin Heidelberg, 2014.
- [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.
- [SHZ⁺14] Heli Sun, Jianbin Huang, Xin Zhang, Jiao Liu, Dong Wang, Huailiang Liu, Jianhua Zou, and Qinbao Song. IncOrder : Incremental density-based community detection in dynamic networks. *Knowledge-Based Systems*, oct 2014.
- [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.
- [SPFCC15] A. Strandburg-Peshkin, D. R. Farine, I. D. Couzin, and M. C. Crofoot. Shared decision-making drives collective movement in wild baboons. *Science*, 348(6241) :1358–1361, jun 2015.
- [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] N Stanley, S Shai, D Taylor, and P Mucha. Clustering network layers with the strata multilayer stochastic block model. *ieeexplore.ieee.org*, 2015.
- [STM15] Leo Speidel, Taro Takaguchi, and Naoki Masuda. Community detection in directed acyclic graphs. *The European Physical Journal B*, 88(8) :203, aug 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.
- [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.
- [Tan11] O Tange. GNU Parallel - The Command-Line Power Tool. *;login : The USENIX Magazine*, 36(1) :42–47, feb 2011.
- [TLB⁺09] P.-U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Dias de Amorim, and J. Whitbeck. The Accordion Phenomenon : Analysis, Characterization, and Impact on DTN Routing. In *IEEE INFOCOM 2009 - The 28th Conference on Computer Communications*, pages 1116–1124. IEEE, apr 2009.
- [Tra15] VA Traag. Faster unfolding of communities : Speeding up the Louvain algorithm. *Physical Review E*, 2015.
- [TRC11] Lionel Tabourier, Camille Roth, and Jean-Philippe Cointet. Generating constrained random graphs using multiple edge switches. *Journal of Experimental Algorithmics*, 16 :1–15, may 2011.
- [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*. Cambridge University Press, 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.

- [Via16] Tiphaine Viard. *Flots de liens pour la modélisation d'interactions temporelles et application à l'analyse de trafic IP*. PhD thesis, Université Pierre et Marie Curie, 2016.
- [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.
- [VLM15] Jordan Viard, Matthieu Latapy, and Clémence Magnien. Revealing contact patterns among high-school students using maximal cliques in link streams. In *First International Workshop on Dynamics in Networks (DyNo)*, Paris, France, 2015.
- [VLM16] Tiphaine Viard, Matthieu Latapy, and Clémence Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609 :245–252, 2016.
- [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.
- [WGD13] Joyce Jiyoung Whang, David F. Gleich, and Inderjit S. Dhillon. Overlapping community detection using seed set expansion. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, CIKM '13, pages 2099–2108, New York, NY, USA, 2013. ACM.
- [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.
- [WZF15] K Wehmuth, A Ziviani, and E Fleury. A unifying model for representing time-varying graphs. *Data Science and Advanced*, 2015.
- [XH14] Kevin S. Xu and Alfred O. Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal on Selected Topics in Signal Processing*, 8(4) :552–562, 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.

- [YCZ⁺11] Tianbao Yang, Yun Chi, Shenghuo Zhu, Yihong Gong, and Rong Jin. Detecting communities and their evolutions in dynamic social networks—a Bayesian approach. *Machine Learning*, 82(2) :157–189, 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] P Zhang. A revisit to evaluating accuracy of community detection using the normalized mutual information. *arXiv preprint arXiv :1501.03844*, 2015.
- [ZLLC15] Yi-Qing Zhang, Xiang Li, Di Liang, and Jin Cui. Characterizing Bursts of Aggregate Pairs With Individual Poissonian Activity and Preferential Mobility. *IEEE Communications Letters*, 19(7) :1225–1228, jul 2015.