Rapport de stage
Structure multi-échelle des grands graphes

Thomas Aynaud
supervisé par Jean-Loup Guillaume et Matthieu Latapy

Stage de mars à septembre 2008
LIP6 - CNRS - Université Pierre et Marie Curie

Introduction

L'analyse de nombreux phénomènes conduit à considérer des grands réseaux. Ces réseaux apparaissent dans de nombreux domaines tels que la sociologie, la biologie, la linguistique, l'informatique, etc. Il est courant de les représenter par des graphes. Cette structure de données permet en effet de modéliser efficacement les relations entre différents acteurs d'un système complexe.

On appelle souvent ces graphes des *graphes de terrain*, ou *complex networks* en anglais. La plupart des graphes de terrain, bien que provenant de contextes très divers, ont des propriétés communes non triviales qui ont fait l'objet de nombreuses études [1, 2]. Par exemple, la distance entre deux nœuds de ces graphes est souvent très faible comparativement à la taille du graphe. Ce phénomène a été constaté par Milgram en 1967 dans la célèbre expérience des six degrés de séparation [3], puis étudié plus en détail par Watts et Strogratz en 1998 [4]. Les graphes ayant cette caractéristique sont appelés des graphes *petits mondes* (*small-world networks*). Une autre caractéristique commune est l'hétérogénéité des degrés des nœuds : il existe des nœuds ayant un très fort degré et la distribution de degrés obéit souvent à une loi de puissance. Les graphes ayant cette caractéristique sont appelés *scale-free networks*.

L'étude de ces propriétés communes est cruciale, d'une part pour comprendre la structure de ces graphes en général et d'autre part pour développer une algorithmique adaptée. La taille des graphes considérés rend, en effet, difficilement envisageable d'utiliser des algorithmes dont la complexité est plus que quadratique en la taille du graphe, et souvent une complexité linéaire s'avère nécessaire. Les objectifs peuvent être très variés selon les disciplines : l'analyse de l'infrastructure des réseaux informatiques va, par exemple, chercher à détecter leurs points faibles [5], les moteurs de recherches vont essayer d'évaluer la pertinence des pages en se basant sur un graphe du Web [6] ou on peut vouloir étudier la propagation des épidémies dans un réseau social de contacts physiques [7].

Il est apparu que les nœuds des graphes de terrain peuvent être souvent classés en communautés, qui elles même sont susceptibles d'en contenir, et ainsi à plusieurs niveaux de profondeur. On peut par exemple imaginer une communauté de toutes les relations d'une personne, elle même découpée en un cercle familial, un cercle d'amis et un cercle de collègues. Le cercle familial peut à nouveau se découper en la famille de sa mère, et la famille de son père.

Ce mémoire s'attache à étudier cette structure multi-échelle. Notre travail s'est orienté vers deux axes :

- D'une part, nous avons eu une approche expérimentale et avons calculé diverses grandeurs usuelles dans l'analyse des réseaux sur les graphes entre les communautés et à l'intérieur des communautés à différentes échelles. Ce travail a été effectué sur deux structures hiérarchiques différentes générées à l'aide d'un outil de détection de communautés.
- D'autre part, nous avons essayé d'utiliser la structure multi-échelle pour la visualisation des très grands graphes en les simplifiant en leurs communautés, en les dessinant par des méthodes classiques de visualisation et en permettant à l'utilisateur de préciser la vision de certaines parties du graphe.

Dans une première partie, nous clarifierons ce que l'on entend par *communauté* et présenterons quelques résultats de ce domaine, puis, dans une deuxième partie, nous expliciterons les structures multi-échelles que nous avons étudiées et les graphes réels sur lesquels nous avons mené nos expériences. Nous présenterons ensuite dans une troisième partie les grandeurs calculées sur ces structures multi-échelles et les résultats obtenus. Nous finirons, dans une quatrième partie, en présentant nos travaux sur la visualisation de graphes avant de conclure.

1 Communautés

1.1 Notion intuitive

Sur un réseau social, on a une idée intuitive de communautés entre des personnes : il y a un groupe familial, un groupe entre amis (voire plusieurs), les collègues, etc. De même, si l'on considère les graphes du web (les nœuds sont les pages web et il y a un lien entre deux pages web si l'on a un lien hypertexte entre elles), on a aussi une notion intuitive de communautés. Une communauté de sites web pourrait alors être des ensembles de pages traitant du même sujet. Si on représente un réseau d'ordinateurs comme l'internet, des communautés pourraient être les clients du même fournisseur d'accès ou des regroupements géographiques par exemple.

Des structures communautaires ont ainsi pu être définies et observées dans de nombreux réseaux comme des réseaux sociaux [8], des réseaux d'interactions biologiques [9] et l'Internet [10]. Il devient alors intéressant d'essayer de les définir mathématiquement et de les calculer automatiquement.

Une approche courante est de modéliser les réseaux par des graphes, et d'essayer de détecter des communautés à partir de la structure de ces graphes (en ayant donc connaissance uniquement des nœuds et des liens entre ces nœuds, sans données attachées aux nœuds comme par exemple un nom et des attributs). Actuellement, détecter des communautés signifie partitionner l'ensemble des nœuds du graphe, chaque partie étant une communauté. En cherchant une partition, un nœud fait partie d'une et une seule communauté, ce qui est une très forte limitation. En effet, intuitivement dans un réseau social, une personne ne ferait pratiquement jamais partie d'une et une seule communauté. On peut trouver des pistes pour avoir des nœuds appartenant à plusieurs communautés à la fois dans [11].

Une définition utilisée fréquemment est que les communautés sont des parties de l'ensemble des nœuds qui vont être fortement connectées intérieurement et faiblement entre elles. C'est à dire des parties qui vont contenir beaucoup de liens entre nœuds internes, et très peu entre nœuds de parties différentes comme par exemple les nœuds dans les cercles pointillés de la figure 1. Il existe d'autres définitions, aucune ne faisant aujourd'hui l'unanimité.

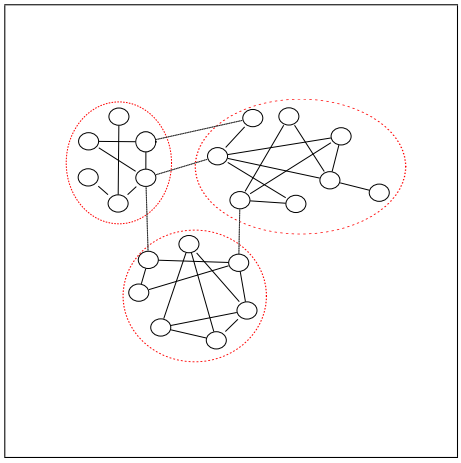

FIG. 1 – Exemple de communautés

1.2 Une définition

De manière plus formelle, on va chercher à optimiser une fonction de qualité associant une valeur à une partition de l'ensemble des nœuds. Il existe plusieurs fonctions de qualité, la plus utilisée étant la *modularité* définie dans [12]. Pour une partition π de l'ensemble des nœuds, en notant L le nombre de liens du graphe, d_s le degré total d'une partie s et l_s le nombre de liens à l'intérieur d'une partie s, la modularité Q est :

$$Q(\pi) = \sum_{s \in \pi} \left(\frac{l_s}{L} - \left(\frac{d_s}{2L} \right)^2 \right)$$

Cette grandeur peut être vue comme la somme des différences entre la proportion de liens à l'intérieur d'une partie s (soit $\frac{l_s}{L}$) et la proportion de liens que devrait avoir une partie d'un graphe aléatoire de même taille (soit $\left(\frac{d_s}{2L}\right)^2$).

C'est cette grandeur, comprise entre -1 et 1, que l'on cherche à maximiser. Il ne faut pas confondre la notion intuitive de communauté que l'on a et la partition obtenue en maximisant la modularité, il existe d'autres fonctions de qualité.

1.3 Méthodes de calcul

Maximiser la modularité est un problème NP-complet [13] et il existe de très nombreuses heuristiques pour tenter de fournir une bonne approximation [14, 15, 16, 17, 18]. Ces heuristiques sont habituellement divisées en deux classes : les approches séparatives et les approches agglomératives.

Les approches séparatives essayent de scinder le graphe en plusieurs communautés en retirant progressivement des arêtes. A chaque étape, les composantes connexes du graphe sont identifiées à des communautés. Le processus est répété tant qu'il reste des arêtes. On obtient ainsi une structure hiérarchique. La racine est le graphe entier et chaque communauté est un sommet de l'arbre qui sont reliés en fonction des séparations : quand on coupe une composante connexe associée au sommet $s1$ de l'arbre en deux nouvelles associées aux sommets $s2$ et $s3$, $s1$ a pour fils $s2$ et $s3$. Les méthodes séparatives diffèrent les unes des autres par la façon de choisir les arêtes à retirer. Par exemple, la technique proposée dans [15] retire les arêtes qui ont la plus forte centralité. La centralité d'une arête est le nombre de plus courts chemins entre deux nœuds du graphe qui passent par cette arête. Intuitivement, les communautés doivent être reliées entre elles par peu de liens. Les liens inter-communautés vont donc être

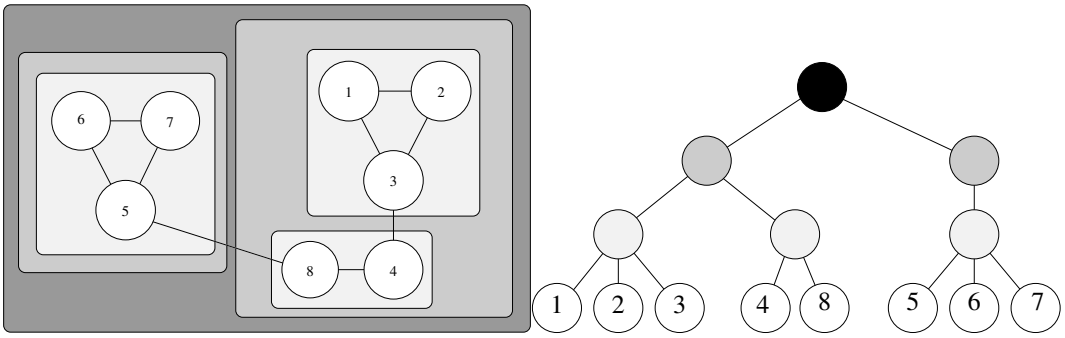

FIG. 2 – Exemple de décomposition en communautés et son dendrogramme

utilisés par tous les chemins entre des nœuds de communautés différentes, et donc avoir une centralité très élevée.

Les approches aglomératives, à l'inverse, commencent par placer chaque nœud dans une communauté unique (autant de nœuds que de communautés initialement) et regroupe les communautés deux par deux jusqu'à n'obtenir qu'une seule communauté. Ici encore, on construit un arbre. Initialement, on a une forêt de feuilles simples constituée des nœuds du graphe initial, et créer une communauté revient à regrouper deux arbres de la forêt en créant un père commun aux deux racines. Au final, quand tous les nœuds sont regroupés, il ne reste qu'un arbre, de feuilles les nœuds du graphe. Les algorithmes diffèrent ici dans la méthode de choix des communautés à regrouper. L'approche gloutonne est utilisée dans [18] et consiste à sélectionner la paire de communautés dont le regroupement maximise le gain de modularité.

Ces deux types d'algorithmes produisent en sortie des arbres appelés dendrogrammes (voir la figure 2). Étant donné un tel arbre, on peut définir pour chaque niveau de profondeur i une partition : les nœuds du graphe initial sont dans la même partie s'ils ont le même père de profondeur i dans l'arbre. Il reste à calculer la modularité partition par partition pour trouver la plus intéressante.

Durant tout le stage, nous avons utilisé l'heuristique décrite dans [19] qui est actuellement l'une des plus performantes, à la fois en terme de qualité (bonne maximisation de la modularité) et de vitesse. Cette heuristique fonctionne par agrégations successives des nœuds en communautés :

1 Supposons avoir initialement un réseau $G = (V, E)$, en affectant un poids 1 à tous les liens si le réseau n'est pas pondéré. Chaque nœud est initialement placé dans une communauté propre. Il y a donc autant de communautés que de nœuds dans la partition initiale.

2 Ensuite, pour chaque nœud i, pour chacun de ses voisins j, soit $g_i(j)$ la différence de modularité en plaçant i dans la même communauté que j. S'il existe un voisin j tel que $g_i(j)$ soit positif, i est déplacé dans la communauté du voisin qui maximise le gain. Sinon, i reste dans sa communauté. Cette opération est itérée jusqu'à ne plus obtenir de gain pour aucun nœud.

3 On construit alors un nouveau réseau où les nœuds sont les communautés trouvées et où il y a un lien entre une communauté c_1 et une communauté c_2 de poids w si la somme des poids des liens de l'ensemble $\{(x, y) \in E, x \in c_1, y \in c_2\}$ est w. On applique à nouveau le procédé en 2 sur ce nouveau graphe tant qu'on obtient une amélioration de la modularité pour le graphe initial.

Le résultat final de l'algorithme est la dernière partition obtenue qui a conduit à une amélioration de la modularité. Chaque partie de cette partition est une communauté. On peut aussi bien sûr construire un dendrogramme durant le déroulement de l'algorithme en rajoutant une racine finale. Les feuilles sont les nœuds du graphe initial (en blanc sur la figure 2). A chaque étape du calcul, on crée un nouveau niveau dans l'arbre (un niveau de gris sur la figure 2) où chaque nœud est associé à un regroupement et a pour fils les nœuds que contient ce regroupement. Le dernier niveau correspond à la décomposition finale en communautés (gris foncé sur la figure 2), à laquelle on rajoute arbitrairement une racine (noir sur la figure 2). La décomposition en communautés maximisant la modularité est la dernière partition. Le dendogramme a ici la particularité d'avoir des nœuds d'arité supérieure ou égale à 2 alors que la plupart des algorithmes donnent un arbre binaire.

2 Objets étudiés

2.1 Une structure multi-échelle sous forme d'arbre

On s'intéresse aux représentations du graphe à différentes échelles. Pour cela, on va utiliser l'algorithme de décomposition en communautés pour regrouper les nœuds et donc obtenir une forme de schématisation du graphe initial.

On va donc associer à notre graphe initial un arbre particulier, dont les feuilles sont les nœuds du graphe.

Pour un tel arbre A, en notant $f(s)$ l'ensemble des feuilles du sous arbre de A de racine s, et pour toute profondeur i, on peut construire un graphe $G_i = (V_i, E_i)$ ainsi :

- $V_i = \{$les sommets de l'arbre A à profondeur $i\}$: les nœuds sont les parties de la partition associée au niveau de profondeur i de l'arbre
- E_i : il y a un lien entre les nœuds u et v s'il y a dans le graphe initial un lien entre un élément de u et un élément de v. On peut aussi pondérer le liens u, v de G_i par le nombre de liens qu'il y a dans le graphe initial entre des éléments de u et des éléments de v.

Chaque graphe G_i peut être vu comme une schématisation des graphes G_j avec $j > i$ dans laquelle certains nœuds ont été regroupés.

2.2 Les structures hiérarchiques associées

Afin que les partitions aient du sens, on va utiliser l'algorithme de détection de communautés décrit précédemment pour calculer ces arbres. Ces graphes permettent d'étudier l'évolution et la structure du graphe entre les communautés à différentes échelles. A un graphe initial donné, on peut en particulier attacher deux arbres :

- Premièrement, on peut considérer le dendrogramme construit lors du déroulement de l'algorithme de détection des communautés. On appellera les graphes définis par cet arbre les graphes entre communautés de niveau i où i est la profondeur dans l'arbre. Le graphe entre communautés de niveau 1 sera donc le graphe entre les communautés calculées par l'algorithme de détection.
- Inversement, on peut partir d'une racine correspondant au graphe initial complet. On la divise en k communautés, ce qui donne k fils de la racine, chacun correspondant aux noeuds d'une communauté. Chaque fils f correspond à un sous graphe, qu'on peut lui même décomposer, ce qui crée des fils de f. On peut continuer jusqu'à obtenir une

décomposition triviale où un sommet s correspond à un sous graphe qui est décomposé en une seule communauté c. On ajoute alors autant de feuilles filles de s qu'il y a de nœuds dans c, chacune correspondant à un nœud. Cela construit donc un arbre qui a pour feuilles les nœuds du graphe initial et on peut associer à chaque niveau des graphes que l'on appellera *graphes extraits au niveau i* où i est la profondeur dans l'arbre où on a sélectionné la partition. Ainsi, le graphe extrait au niveau 1 sera aussi le graphe entre les communautés de niveau 1.

2.3 La structure à l'intérieur des regroupements

Un autre objet d'étude est la structure du graphe à l'intérieur d'une communauté, et non entre elles. On a donc dans un premier temps extrait, lors du déroulement de l'algorithme de calcul de communautés, le sous graphe du graphe initial contenant les nœuds de la plus grande communauté trouvée. On les appellera *graphe à l'intérieur de la plus grande communauté au niveau i*.

De la même manière, on a procédé aussi dans le sens inverse en commençant par calculer les communautés puis en extrayant le sous graphe correspondant à la plus grande et en répétant récursivement l'opération sur ce graphe, jusqu'à ce qu'on obtienne une décomposition en une seule communauté. On appellera ces graphes *graphe extrait récursivement au niveau i*.

2.4 Les réseaux initiaux

Nous avions à notre disposition quatre grands graphes :

- Le premier, *webndu*, est un graphe du web. Chaque nœud est une page web, et il y a un lien entre deux nœuds si on avait un lien hypertexte de l'une vers l'autre. C'est une carte complète du domaine .nd.edu en 1999 qui contient 325729 pages et plus d'un million de liens [20]. Initialement orienté, on l'a considéré comme un graphe non orienté.
- Le second, *arxiv*, est un graphe de citation entre auteurs d'articles de recherche dans le domaine de la physique. A chaque auteur correspond un nœud du graphe, et il y a un lien entre deux auteurs s'ils ont cosigné un ou plusieurs articles. Il a été extrait de la base de donnée d'articles de recherche *arxiv.org* par Newman dans [21]. Les nœuds de degré 1 ont été supprimés, ce qui explique la différence avec les données de l'article original mais ne change rien pour les communautés.
- Enfin, les deux derniers sont extraits d'une trace d'échange pair à pair sur le réseau eDonkey collectée précédemment [22]. Il en a été extrait une heure de données sous la forme de liens : client_demandeur $\leftrightarrow$ fichier_demandé. Les données sont donc dans un premier temps un graphe biparti. Afin d'obtenir un graphe entre les fichiers et un entre les clients, ce graphe a été *projeté*. L'opération dite de projection transforme un graphe $G_{initial} = (V_1 \cup V_2, E)$ en un graphe $G_{\text{projeté}} = (V_1, E_1)$ avec $\{u, v\} \in E_1$ si et seulement si il existe $w \in V_2$ tel que $\{u, w\} \in E$ et $\{w, v\} \in E$. La figure 3 montre un exemple de projection. On a donc obtenu deux graphes, l'un reliant les clients du réseau pair à pair recherchant les mêmes fichiers, et l'un entre les fichiers demandés par les mêmes personnes. On appellera ces graphes *clients pair à pair* et *fichiers pair à pair*

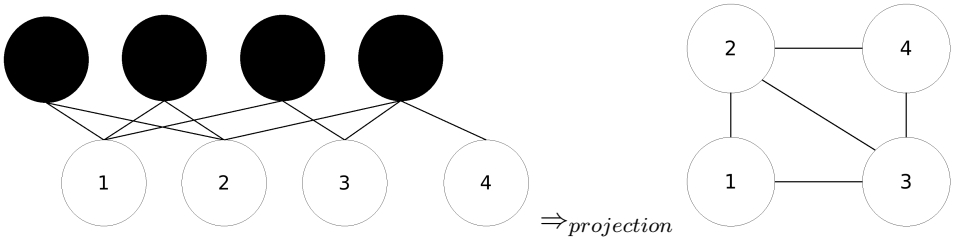

FIG. 3 – Un graphe biparti et son projeté

3 Résultats expérimentaux

3.1 Comparaison des deux structures hiérarchiques

Afin de comparer les deux décompositions hiérarchiques proposées, nous avons décidé de calculer l'information mutuelle normalisée [23, 24] entre les partitions engendrées par ces structures aux différents niveaux. Il s'agit d'une normalisation de la notion d'information mutuelle utilisée en théorie de l'information. Pour deux variables aléatoires X et Y, leur information mutuelle $I(X,Y)$ est :

$$I(X,Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)\,P(y)}$$

L'information mutuelle est positive ou nulle, symétrique, et vaut 0 si et seulement si X et Y sont indépendantes. On peut montrer que $I(X,X)$ vaut $H(X)$ l'entropie de X et que $I(X,Y) \leq \min(H(X),H(Y))$. L'information mutuelle normalisée $NMI(X,Y)$ entre deux variables aléatoires X et Y vaut :

$$NMI(X,Y) = \frac{2I(X,Y)}{H(X)+H(Y)}$$

Elle est comprise entre 0 et 1, vaut 0 pour deux variables indépendantes et 1 si $X = Y$.

A une partition A on associe la variable aléatoire X_A qui vaut pour chaque nœud v la partie de A à laquelle il appartient. On a alors que $P(x) = \frac{|x|}{n}$ avec $|x|$ le cardinal de la partie x et n le nombre de nœuds. De même, $P(x,y) = \frac{|x \cap y|}{n}$. L'information mutuelle normalisée entre deux partitions s'exprime alors ainsi :

$$NMI(A,B) = \frac{-2\sum_{a\in A}\sum_{b\in B} |a\cap b| \log\left(\frac{|a\cap b| n}{|a||b|}\right)}{\sum_{a\in A} |a| \log\left(\frac{|a|}{n}\right) + \sum_{b\in B} |b| \log\left(\frac{|b|}{n}\right)}$$

C'est une mesure de similarité entre les partitions comprise entre 0 (partitions très différentes) et 1 (partitions égales).

Afin d'avoir des arbres de même profondeur, on prolonge les feuilles quand nécessaire. Pour avoir un arbre de profondeur p, à chaque feuille f qui est à une profondeur k inférieure strictement à p, on ajoute une succession de $p-k$ sommets, le premier étant un fils de f, de façon à prolonger la branche par un fil de longueur suffisante.

Profondeur	Arxiv	Webndu	Clients p2p	Fichiers p2p
1	5.395	6.695	6.161	4.872
2	5.395	6.707	6.259	4.910
3	6.026	7.046	7.167	5.453
4	7.855	8.240	9.873	7.859
5	10.595	11.654	13.895	11.549
6	13.188	18.200	18.484	17.372
7	13.195	18.302	18.724	17.737
8	x	18.319	18.746	17.788
9	x	18.313	18.746	17.787
10	x	x	18.746	x
Information mutuelle entre les partitions				

Profondeur	Arxiv	Webndu	Clients p2p	Fichiers p2p
1	1	1	1	1
2	0.758	0.770	0.720	0.666
3	0.721	0.665	0.672	0.590
4	0.794	0.667	0.763	0.690
5	0.898	0.795	0.880	0.824
6	0.999	0.997	0.993	0.988
7	1	0.999	0.9994	0.999
8	x	1	1	1
9	x	1	1	1
10	x	x	1	x
Information mutuelle normalisée entre les partitions				

FIG. 4 – Information mutuelle entre les différentes partitions

Les résultats sont présentés sur la figure 4, où on trouve, pour chaque niveau i, les résultats de l'information mutuelle et de l'information mutuelle normalisée entre la partition du graphe entre communautés de niveau i et la partition du graphe extrait de niveau i.

On constate que les deux partitions commencent par s'éloigner quand on descend dans l'arbre puis se rejoignent. Le premier niveau de l'arbre engendre la même partition dans les deux cas, car il s'agit de la décomposition finale en communautés telle que calculée par l'algorithme, d'où la valeur de 1 de l'information mutuelle. Le niveau le plus bas est aussi la même partition pour les deux arbres : chaque nœud du graphe est dans son singleton.

3.2 Grandeurs mesurées

Sur chaque graphe, nous avons calculé diverses grandeurs classiques dans l'étude des graphes de terrain. On trouve, pour un graphe $G = (V, E)$ en notant k_i le degré du nœud i :

- Les nombres de nœuds et d'arêtes
- La densité : $\frac{2|E|}{|V|(|V|-1)}$. Comprise entre 0 et 1, elle exprime si le graphe est proche d'un graphe complet (densité à 1) ou d'un graphe sans arête (densité à 0)
- La transitivité : $\frac{3N_\triangle}{N_\vee}$ avec $N_\triangle$ le nombre de triangles (triplets de nœuds $\{i, j, k\}$ avec $\{i, j\}, \{j, k\}, \{k, i\} \in E$ et $N_\vee$ le nombre d'ensemble de trois nœuds avec au moins deux arêtes.
- Le clustering : $\frac{1}{|V|} \sum_{i \in V} \frac{2N_\triangle(i)}{k_i(k_i - 1)}$ avec $N_\triangle(i)$ le nombre de triangles contenant i
- La distance moyenne (approchée) : pour chaque nœud i on calcule $d_i = \frac{1}{|V|} \sum_{j \in V} d(i, j)$ et on en prend la moyenne. Vu la taille des graphes considérés, on s'est contenté de prendre la moyenne sur dix nœuds choisis aléatoirement.
- Le degré moyen, $\frac{1}{|V|} \sum_{i \in V} k_i$
- Le degré maximal, $\max_{i \in V}(k_i)$
- La distribution de degré, $p(k)$ la probabilité pour un nœud d'être de degré k

3.3 Résultats

Les résultats obtenus sont contenus dans les figures [5,15,16,17]. On y trouve, pour chaque graphe initial cinq tableaux contenant les données. Le dernier niveau des graphes entre communautés et des graphes extraits sont le graphe initial et ne sont pas recopiés. Les grandeurs ont été arrondies pour plus de lisibilité.

3.3.1 Graphes entre communautés aux différents niveaux

Le nombre de niveaux est relativement stable (entre 5 et 6). On constate que le nombre de nœuds diminue fortement durant le déroulement global de l'algorithme (les niveaux suivent la profondeur dans l'arbre et non le déroulement de l'algorithme). La décroissance est d'abord très forte puis se stabilise à la fin de l'algorithme. Durant les dernières étapes, seulement quelques nœuds sont fusionnés.

Le nombre de liens suit la même évolution : une décroissance très forte au début de l'algorithme et une stabilisation. Quand on trace log (nb de liens) en fonction du log (nb de nœuds), on obtient une droite pour tous les graphes, ce qui tend à indiquer que nb de liens $\sim$ (nb de nœuds)$^\alpha$ pour un α dépendant du graphe (figure 6).

Graphe initial									
	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	
	325729	1.14e+06	2.15e-05	10740	0.087	0.444	6.338	7.00	
Graphe entre communautés niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	modul arité
1	292	1913	0.0381	241	0.223	0.677	2.25	13.10	0.92644
2	296	1936	0.0376	245	0.217	0.676	2.364	13.08	0.92642
3	466	2796	0.0215	409	0.113	0.655	2.24	12	0.926
4	2138	10067	0.0035	1514	0.0234	0.618	2.67	9.42	0.919
5	19318	81771	0.0003	7373	0.0057	0.506	3.90	8.47	0.812
Graphe à l'intérieur de la plus grande communauté niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	
1	50183	116012	0.00009	4024	0.0186	0.253	5.40	4.62	
2	50183	116012	0.00009	4024	0.0186	0.253	4.75	4.62	
3	50087	115869	0.00009	4024	0.0186	0.254	4.71	4.63	
4	46650	109578	0.00010	3908	0.0190	0.258	4.92	4.70	
5	20002	53792	0.00027	2406	0.0264	0.309	4.68	5.38	
Graphe extrait niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	
1	292	1913	0.0382	241	0.2232	0.68	2.28	13.10	0.926
2	8974	29120	0.0006	2661	0.0417	0.59	5.17486	6.49	0.803
3	72257	296600	0.0001	2661	0.3812	0.59	6.47	8.21	0.470
4	167302	646890	4.42e-05	8540	0.1724	0.52	6.34	7.73	0.207
5	262977	918841	2.59e-05	10369	0.1244	0.46	6.72	6.99	0.072
6	311189	1.08e+06	2.22e-05	10645	0.1034	0.45	6.95	6.95	0.015
7	324185	1.13e+06	2.15e-05	10719	0.0896	0.45	6.85	6.99	0.001
Graphe extrait récursivement niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	modul arité
1	50183	116012	9.21e-05	4024	0.019	0.253	5.12	4.62	0.778
2	3619	8742	0.0013	158	0.133232	0.287	4.97	4.83	0.789
3	600	1156	0.0064	79	0.0797133	0.336	4.26	3.85	0.700
4	106	241	0.0433	35	0.112025	0.245	3.132	4.55	0.407
5	27	38	0.1082	8	0.0952381	0.082	2.923	2.81	0.439
6	8	10	0.3571	5	0.272727	0.311	1.55	2.5	0.165

FIG. 5 – Résultats pour Webndu

L'algorithme est donc très efficace durant ses premières étapes : on obtient très vite une partition proche de la décomposition finale en communautés. Cela explique pourquoi il y a relativement peu d'étapes durant l'algorithme et son efficacité (on travaille très vite sur un graphe bien plus petit que le graphe initial).

La densité évolue similairement au nombre de liens. Elle augmente régulièrement, et on obtient aussi une droite quand on trace log (densité) en fonction de log (nb de nœuds) (fi-

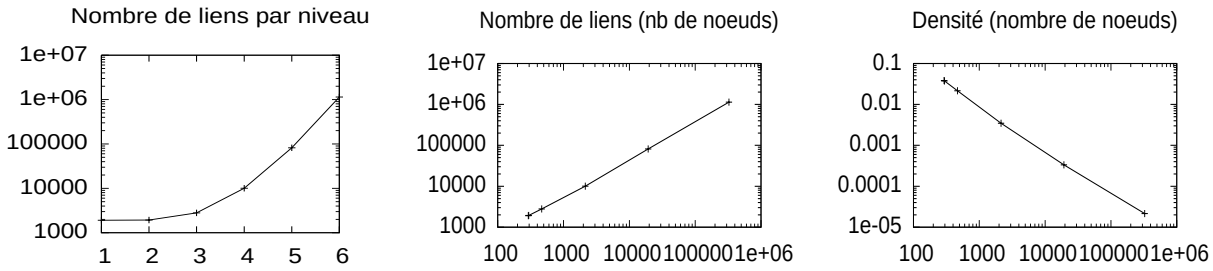

FIG. 6 – Résultats concernant l'évolution du nombre de liens et de la densité : nombre de liens par niveau (échelle log en y), nombre de liens en fonction du nombre de nœuds (échelle log en x et en y) et densité en fonction du nombre de nœuds (échelle log en x et en y)

gure 6).

Le degré maximum et le degré moyen n'évoluent pas selon des tendances remarquables : sur certains graphes, ils croissent puis diminuent, sur d'autres ils ne font que diminuer, sur d'autres ils augmentent. Si on effectue le tracé de la distribution de degré en échelle loglog (figure 7), donc en représentant le $\log(P_k)$ en fonction de $\log(k)$ où P_k est la probabilité d'être de degré k, on obtient presque dans les niveaux les plus profonds des droites. Ceci est signe d'une distribution de degrés en loi de puissance, et donc d'un étalement des degrés possibles des nœuds : il y a des nœuds de très forts degrés (des hubs) et la valeur moyenne est peu significative. Quand le niveau diminue, le caractère rectiligne est bien moins marqué suivant les graphes. Néanmoins, le spectre des degrés possibles reste étendu. Ceci tend à prouver que le réseau reste *scale free* aux différents niveaux.

La transitivité et le clustering sont croissants durant le déroulement de l'algorithme. Le clustering au niveau i est supérieur au clustering au niveau j avec $j > i$. On peut noter par contre parfois une décroissance lors de la première étape (passage du graphe initial aux premières communautés). Cela peut sans doute s'expliquer par le fait que les triangles sont regroupés en une communauté assez rapidement.

La distance moyenne est décroissante durant le déroulement de l'algorithme (elle est plus faible au niveau 1 qu'aux niveaux supérieurs) et est donc très faible à toutes les étapes. Avec une distance moyenne décroissante et un clustering croissant, si le graphe initial est *petit monde* (distance moyenne faible et clustering élevé), le graphe entre les communautés reste un graphe *petit monde* à chaque niveau hiérarchique.

Les grandes propriétés telles que le fait d'être petit monde et scale free se conservent donc aux différents niveaux. Le graphe est de plus en plus dense quand on monte dans l'arbre, ce qui explique l'augmentation du clustering et de la transitivité. Le fait que les degrés moyens et le degré maximum n'évoluent pas selon des tendances générales s'explique par le fait que le graphe est *scale free*.

3.3.2 Graphe à l'intérieur de la plus grande communauté aux différents niveaux

Alors que le graphe entre communautés était un graphe n'ayant a priori pas de liens ou de nœuds en commun avec le graphe initial, le graphe à l'intérieur de la plus grande communauté est un sous graphe du graphe initial. Il représente entre 10% et 20% des nœuds à la fin du calcul. Il y a au moins 60 communautés dans les graphes étudiés, et par conséquent, la plus

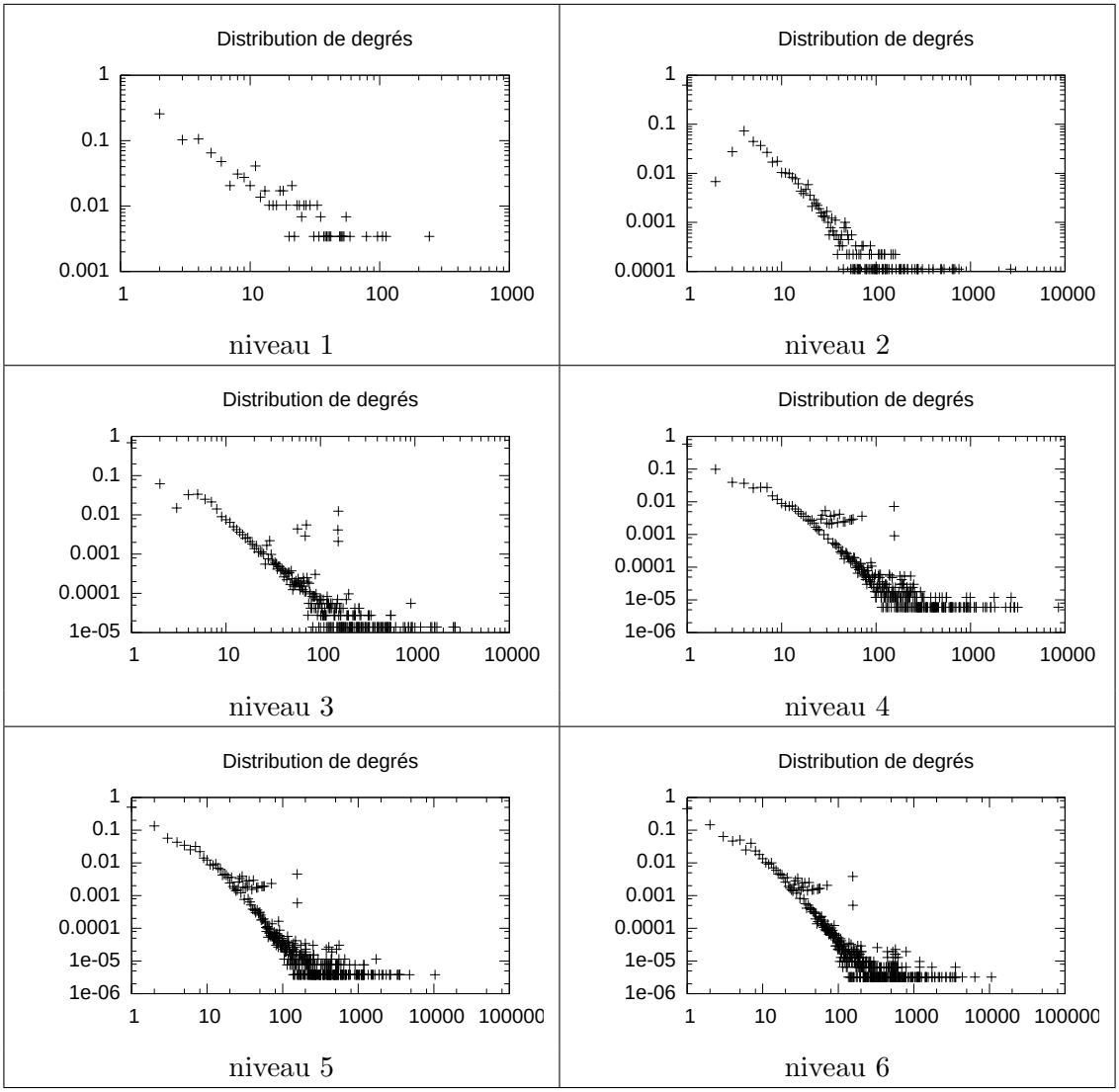

FIG. 7 – Distribution de degrés pour le graphe entre les communautés niveau par niveau

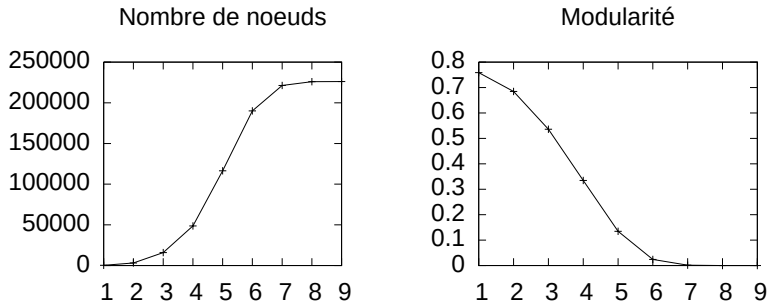

FIG. 8 – Nombre de nœuds par niveau et modularité par niveau

grande communauté est très grande par rapport aux autres.

Le graphe à l'intérieur de la plus grande communauté lors du regroupement des nœuds durant l'algorithme converge très vite vers un graphe proche du graphe final en nombre de nœuds et de liens. Après la deuxième étape, la taille du graphe n'évolue presque plus. L'évolution du graphe est trop rapide pour en déduire des tendances à plusieurs échelles : il n'y a pas assez d'étapes.

On constate quand même que ces graphes sont plus denses que le graphe initial. Ceci est la définition d'une communauté : une partie fortement connectée intérieurement (donc plus dense que le reste du graphe) et faiblement extérieurement. A l'exception de *webndu*, on gagne un facteur dix en densité

En revanche, le clustering et la transitivité semblent être plus faible dans ces sous graphes que dans le graphe initial. Ce résultat est étrange, la taille du graphe étant plus petite et la densité plus grande tendraient plutôt à un agrandissement de ces grandeurs. Nous n'avons pas d'explications convaincantes de ce phénomène.

La distance moyenne à l'intérieur de ces graphes est inférieure à celle des graphes initiaux. Ce n'était pas évident, les plus courts chemins pouvant passer par un hub en dehors de cette communauté. Cela est néanmoins cohérent avec le fait que ces graphe soient plus petits que les graphes initiaux et que leurs densités soient bien plus importantes.

La plus grande communauté à chaque étape apparaît donc comme un sous graphe du graphe initial particulièrement dense, mais n'a pas de propriété remarquable. Le clustering est un peu plus faible, mais la différence n'est pas très importante (sur webndu, le cas le plus extrême, le clustering a été divisé par $1,7$ alors que la taille du graphe est divisé par $6,5$). La convergence très rapide vers la plus grande communauté empêche de faire une analyse niveau par niveau de ces graphes.

3.3.3 Graphes extraits aux différents niveaux

On étudie maintenant les graphes entre les communautés obtenues lorsqu'on extrait les communautés récursivement jusqu'à obtenir une décomposition triviale où il n'y a qu'une seule communauté.

Le nombre de niveaux est là aussi presque constant et situé entre 7 et 10. Les derniers niveaux semblent peu intéressants, car seulement quelques communautés peuvent encore être décomposées.

Le nombre de nœuds croît en deux étapes (représenté sur la figure 8) : au début, la crois-

sance est exponentielle, puis se stabilise deux ou trois niveaux avant le dernier. Les quelques nœuds ajoutés à la fin correspondent aux derniers sous graphes que l'on peut décomposer et le nombre de niveaux où il y a des changements sensibles est donc proche de celui des graphes entre communautés.

Le nombre de liens croît similairement au nombre de nœuds. On obtient aussi une droite en traçant log(nombre de liens) en fonction de log(nombre de noeuds).

L'évolution du degré maximum et du degré moyen dépend énormément du graphe considéré. Là encore, le tracé de la distribution de degré en échelle loglog, on constate aussi que les degrés se répartissent sur un grand spectre de valeurs, ce qui explique la grande variation de la moyenne et du maximum. Les graphes se comportent ici encore comme des réseaux *scale free* aux différents niveaux.

La distance moyenne reste comprise entre la valeur prise par le graphe des communautés au niveau 1 et le graphe initial et reste donc très faible.

Le clustering en revanche évolue différemment suivant le graphe initial. Pour le graphe webndu, le clustering en fonction du niveau est décroissant. Sur les autres graphes, le comportement typique est représenté sur la figure 9. Il commence par décroître fortement puis remonte vers la valeur du graphe initial.

La transitivité évolue aussi différemment suivant le graphe initial. Le comportement est une oscillation autour de la valeur du graphe initial pour webndu et arxiv et similaire à celui du clustering pour fichiers pair à pair et clients pair à pair.

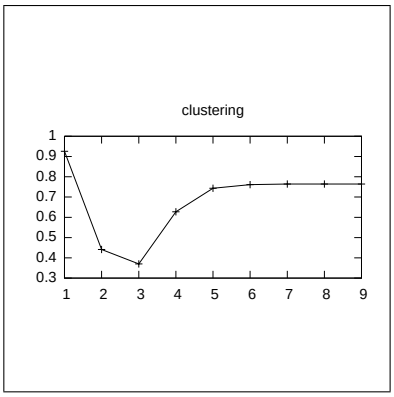

FIG. 9 – Clustering en fonction du niveau

La modularité décroît relativement régulièrement. La structure communautaire n'est pas très bien identifiée : si on compare un niveau du graphe entre communautés et un autre niveau du graphe extrait de manière à ce qu'ils aient le même nombre de nœuds, le graphe entre communautés atteint une modularité sensiblement plus forte.

Les graphes extraits aux différents niveaux semblent moins proches structurellement du graphe initial que les graphes entre communautés. Le clustering et la transitivité n'évoluent pas de manière monotone, ce qui tend à montrer que l'extraction de sous graphes puis l'extraction de communautés dedans casse une partie des relations. Si on exclut les derniers niveaux, très proches du graphe initial, ces grandeurs évoluent de manière irrégulière. Étudier des sous graphes indépendamment de cette manière conduit à étudier des graphes qui n'ont plus la même structure que le graphe initial. La faiblesse de la modularité tend à indiquer que la structure de communautés ainsi trouvée aux niveaux plus profonds n'est pas intéressante.

3.3.4 Graphes extraits récursivement aux différents niveaux

Regardons maintenant la plus grande communauté. Nous avons extrait son sous graphe correspondant dans le graphe initial, puis avons extrait les communautés dans ce sous graphe et avons recommencé récursivement sur ce sous graphe. Nous avons donc une série de sous graphes du graphe initial imbriqués, donnant une information sur la structure locale contrairement à la série de graphes précédente.

Le nombre de nœuds et le nombre de liens en fonction du niveau décroissent exponentiellement. Le degré maximum est lui aussi décroissant. Cette apparente stabilité est sans doute

liée uniquement à la diminution de la taille du graphe. En effet, la distribution de degré est ici encore rectiligne en échelle loglog, signe d'une loi de puissance et donc que les degrés varient beaucoup suivant les nœuds. Le degré moyen varie, lui, différemment suivant le graphe initial, signe qu'il n'y a pas de grandeur moyenne typique du graphe initial. Les sous graphes sont aussi scale free quand leur taille permet d'avoir une distribution de degrés significative.

La densité est croissante et reste toujours supérieure à celle du graphe initial. Les parties extraites du graphe initial sont en effet censées être plus denses intérieurement, vu qu'elles sont issues de communautés. La transitivité et le clustering dépendent beaucoup du graphe initial. A l'exception du graphe webndu, ces deux grandeurs restent assez élevées. La distance moyenne en revanche reste, elle, très faible.

Vu que l'on redécompose récursivement en communautés, on a à chaque étape une valeur atteinte par la modularité. Elle reste quasiment constante voire croît un petit peu pendant deux ou trois étapes, puis décroît d'un coup et continue à décroître régulièrement. Cela laisse supposer qu'il existe effectivement plusieurs niveaux de décompositions en communautés. Il faut s'interroger au moment où la modularité commence à chuter si on n'est pas dans un cas où il n'existe pas de vraies communautés intéressantes et où l'algorithme effectue un classement arbitraire ne correspondant à aucune communauté réelle. La modularité maximale dépendant en partie de la taille du graphe considéré, la chute de la modularité peut aussi s'expliquer par la diminution de la taille du graphe.

Au final, ré-extraire des communautés à l'intérieur de celles trouvées par l'algorithme est intéressant pour l'étude locale de ces communautés. On brise la structure globale du graphe initial, mais chaque communauté a elle même, indépendamment des autres, une structure communautaire propre. On n'a hélas pu mener cette étude que sur la plus grosse communauté. La taille des communautés est très variable, et celles qui sont très petites n'ont peut être pas ces sous communautés.

Nous avons donc défini deux structures hiérarchiques qui permettent d'obtenir quatre séries de graphes. La première, les graphes entre communautés, apparait effectivement comme un repliement relativement régulier du graphe initial. Les grandeurs étudiées évoluent vers un graphe plus petit et plus dense. La série de graphes à l'intérieur de la plus grande communauté n'est pas très intéressante du point de vue multi-échelle, car cette plus grande communauté n'évolue pas beaucoup. Elle est néanmoins toujours bien un sous graphe dense du graphe initial. Les conclusions concernant les graphes extraits aux différents niveaux sont plus mitigées. Les grandeurs classiques évoluent très différement suivant les graphes initiaux et la modularité est moins bonne que pour la première série. Par contre, si on considère le graphe extrait récursivement, on trouve effectivement une structure communautaire de très forte modularité. Cela peut indiquer d'une part que toutes les communautés n'ont pas de sous structure communautaires, et que celles qui n'en ont pas font chuter la modularité globale, ou que ces sous structures recouvrent plusieurs sous-graphe, et qu'on les supprime en extrayant localement des sous graphes.

4 Application au dessin de graphes

Le deuxième objectif du stage était d'utiliser la séparation en communautés et la structure multi-échelle des très grands graphes pour essayer de les représenter. Nous avons décidé d'utiliser des techniques classiques de dessin de graphe. Il en existe de très nombreuses [25]. Le but de ces algorithmes est d'obtenir une représentation esthétique des graphes dans un

espace en deux ou trois dimensions. Il y a divers critères d'esthétisme comme par exemple :

- Le nombre d'arêtes se coupant, à minimiser
- La taille des arêtes, qui doit être la plus uniforme possible.
- La répartition des nœuds, devant couvrir le plus possible l'espace alloué.
- La représentation de certaines propriétés du graphe, comme des symétries ou que la distance dans la représentation du graphe entre deux nœuds soit corrélée à la distance dans le graphe.
- Des critères plus subjectifs de lisibilité.

4.1 Algorithmes de forces

L'une des approches les plus généralistes repose sur une résolution de contraintes physiques sur le graphe. Diverses forces sont appliquées entre les nœuds et le résultat physique est simulé en partant d'un placement aléatoire des nœuds. Les forces utilisées classiquement sont de deux natures :

1 Les liens entre les nœuds sont vus comme des ressorts. Cette force pousse les sommets à être de longueur l, une constante dépendant du ressort que l'on modélise
2 Les nœuds sont vus comme des particules chargées se repoussant mutuellement suivant la loi de Coulomb.

En notant en gras les vecteurs, chaque nœud v à la position $\mathbf{v}$ est donc soumis à la somme $\mathbf{\Sigma}(v)$ de :

- Pour chaque lien $v, w \in E$ connectant ce nœud, une force de la forme $\mathbf{F} = -k\delta\mathbf{u}_{vw}$ avec k la constante de raideur du ressort, δ la différence entre la longueur du lien et la longueur à vide l du ressort et $\mathbf{u}_{vw}$ le vecteur unitaire de direction v, w orienté de v vers w.
- Pour chaque nœud $w \in V$ autre que v, une force de répulsion de la forme $\mathbf{F} = -k_e \frac{q^2}{\|vw\|}\mathbf{u}_{vw}$ où k_e et q sont des paramètres, $\|vw\|$ la distance entre les centres des nœuds v et w et $\mathbf{u}_{vw}$ le vecteur unitaire de direction v, w orienté de v vers w.

L'algorithme final est alors l'algorithme 1, en notant $\mathbf{V}_v(t)$ la vitesse du nœud v au temps t et $\mathbf{v}(t)$ la position du nœud v au temps t et en découpant le temps par intervalles de durée Δ_t.

```
1 Placer aléatoirement tous les nœuds;
2 Affecter une vitesse nulle à tous les nœuds;
3 while le système évolue do
4 |   foreach nœud v do
5 |   |   Calculer Σ(v);
6 |   |   Calculer la vitesse du nœud : V_v(t + Δ_t) = V(t) + Σ;
7 |   |   Sa nouvelle position est : v(t + Δ_t) = v(t) + +Δ_t V(t + Δ_t);
8 |   end
9 end
```

Algorithm 1: Algorithme de placement par forces

La complexité d'une itération est en $O(|V|^2)$ et il est considéré qu'il faut $O(|V|)$ itérations, donc un algorithme au total cubique en le nombre de nœuds. Il existe beaucoup d'extensions et d'améliorations à cette méthode comme, par exemple, des approximations pour ne pas calculer des forces entre des nœuds beaucoup trop éloignés pour influer les uns sur les autres [26]

ou pour permettre de tenir compte de la taille des nœuds [27].

4.2 Existant

Il existe déjà plusieurs méthodes pour représenter les graphes à différents niveaux. Dans [28], Eades propose un algorithme de dessin très général de graphes planaires associés à une décomposition hiérarchique (conservant la planarité à chaque niveau). Le graphe est initialement dessiné, puis la partition la plus profonde par dessus de manière à ce qu'une partie englobe les nœuds qu'elle contient et ainsi de suite. Le dessin est alors plusieurs couches successives présentant le graphe à différents niveaux. Seulement, ce type de dessin n'est pas toujours esthétique si le graphe n'est pas planaire. De plus, cela suppose un placement de tous les nœuds, ce qui est impossible avec des graphes de grandes tailles.

Un approche utilisant les communautés est décrite dans [29]. La méthode de décomposition en communauté est ici assez différente de ce qui a été présenté. Le nombre de communautés attendues p fait partie des paramètres. Le graphe est donc décomposé en p communautés, et le graphe entre ces p communautés est affiché par une méthode de dessin de graphe classique. On peut visualiser différents niveaux d'approximation en jouant sur le nombre de communautés. L'intérêt est d'avoir une vision simplifiée de graphe en contrôlant le niveau de simplification. En revanche, on ne peut pas explorer facilement une partie précise du graphe.

Une troisième approche décrite dans [30] utilise une approche séparative pour détecter des communautés. La représentation du graphe des communautés sert alors de carte pour naviguer dans le graphe, lui même représenté par un algorithme de dessins de graphe classique. La force des arêtes est corrélée au poids que leur donne l'algorithme de détection des communautés. Les arêtes entre communautés auront un poids plus faible, et donc pourront être plus longues et seront représentées par des couleurs moins intenses, de façon à visualiser plus facilement les regroupements.

4.3 Méthode employée

Il est illusoire de tenter de représenter tels quels les graphes que l'on considère : ils contiennent bien trop de nœuds et d'arêtes pour que des algorithmes de dessin courants arrivent en un temps raisonnable à calculer une représentation (mais cette limitation tend à disparaître, on arrive à dessiner des graphes de dizaines de milliers de nœuds) et une telle représentation serait totalement inexploitable. Un humain a bien du mal à visualiser ce qu'il se passe quand il y a trop d'objets affichés en même temps à l'écran.

Pour compenser cela, nous avons donc essayé de schématiser le graphe initial par sa décomposition en communauté.

On suppose avoir une fonction prenant un graphe en entrée et donnant en sortie les coordonnées de chaque nœud dans une visualisation où les nœuds sont des cercles, chacun d'une surface donnée en paramètre. Cette fonction sera initialement implémentée en utilisant *graphviz*, un outil très courant de visualisation de graphe.

Ensuite, l'application de visualisation fonctionne comme ceci (figure 10) :

- Elle calcule la décomposition en communautés du graphe $G = (V, E)$ donné en paramètre, et fait représenter le graphe des communautés avec des nœuds de tailles proportionnelles au nombre d'éléments que contient sa communauté associée.
- A chaque nœud n représenté à l'écran est associée une partie P_n de l'ensemble des nœuds du graphe initial. Il s'agit pour les premiers nœuds affichés d'une communauté

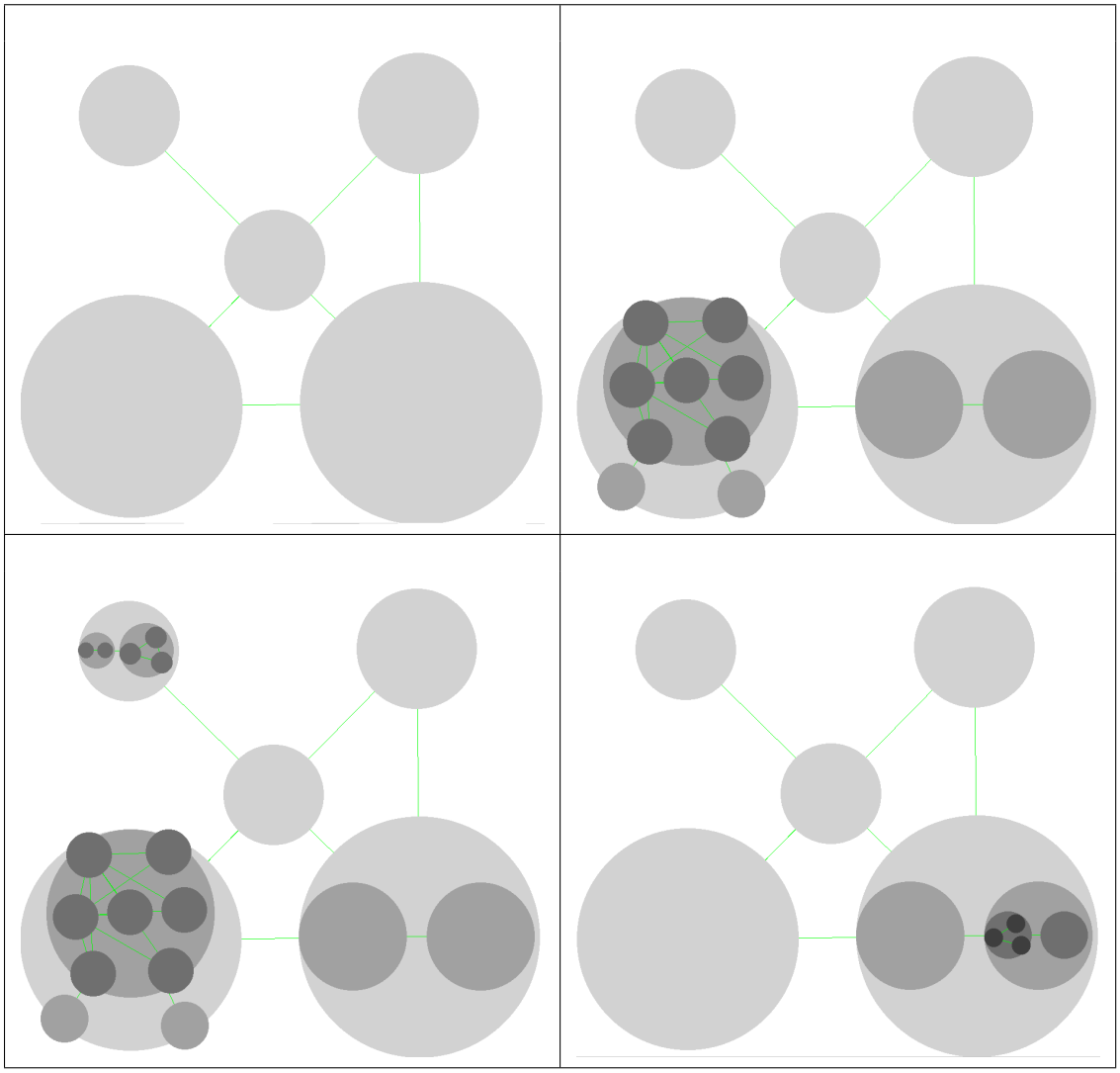

FIG. 10 – L'application de dessin de graphes, d'abord le rendu initial puis après l'exploration de certaines communautés. Plus le nœud est foncé, plus le graphe est un graphe entre communautés profondes.

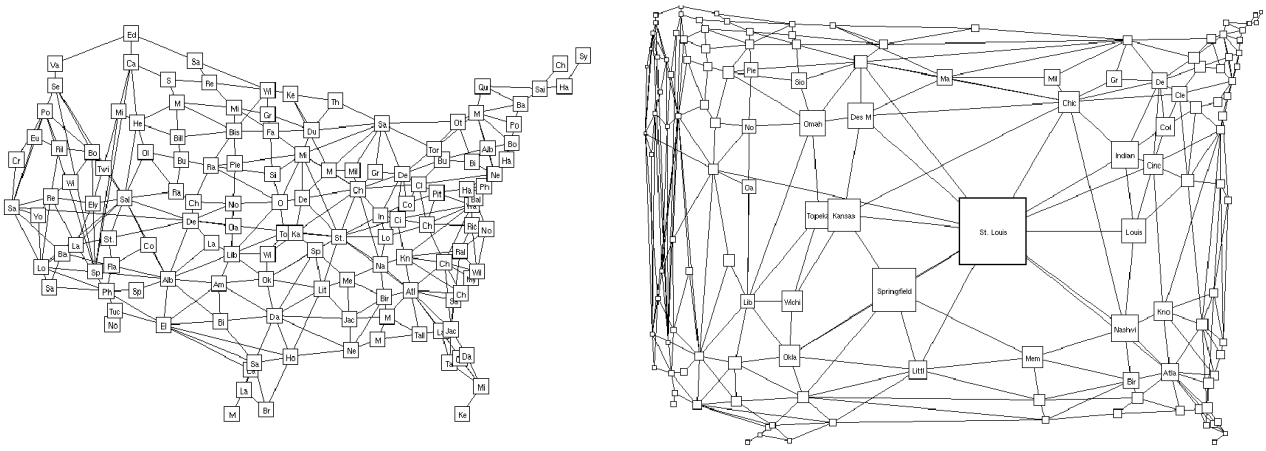

FIG. 11 – Rendu en fish eye du plan des USA tiré de [31]

du graphe initial.

– Quand l'utilisateur clique sur un nœud n dans la fenêtre, l'application extrait le sous graphe $G' = (P_n, E_n)$ du graphe initial, avec $E_n = \{(x, y) \in \text{tels que} x \in P_n, y \in P_n\}$, calcule les communautés dans G' et représente le graphe entre les communautés dans G' redimensionné pour être inclus dans l'image du nœud n. Chaque nouveau nœud n' représenté est associé à la partie de P_n correspondant à la communauté qu'il symbolise, et donc à une partie $P_{n'}$ de V.

4.4 Problèmes rencontrés

Nous n'avons hélas pas eu le temps d'aller plus loin et ce mode de représentation pose un certain nombre de problèmes.

Premièrement, la décomposition en communauté en maximisant la modularité semble encore diviser les graphes en trop de communautés pour être vraiment exploitable, comme on le voit sur la figure 12. Dessiner un graphe d'environ 300 sommets est possible techniquement (les algorithmes de placement terminent en un temps raisonnable) mais le graphe reste très peu lisible, il y a trop d'informations à l'écran : beaucoup d'arêtes se coupent, l'espace manque à l'écran pour représenter tous les nœuds. Cela oblige à représenter des nœuds de très faibles tailles, et donc empêche de zoomer efficacement sur les parties que l'on veut détailler. Une solution serait d'appliquer un algorithme de vision dite *fisheye* [31] lorsque l'on sélectionne une communauté à détailler Cette méthode permet de créer un effet d'optique similaire à celui d'une lentille, la zone d'intérêt étant fortement zoomée et le reste rapetissé, comme par exemple sur la figure 11.

L'utilisation d'un rendu "fisheye" améliorerait aussi beaucoup l'utilisation de l'espace. En effet, quand on zoome dans une communauté, seule la surface initialement consacrée à cette communauté est utilisée, et c'est en général une partie infime de la surface d'affichage. Même avec, par exemple, 4 communautés, chacune se voit affecter moins d'un quart de la surface disponible (il faut représenter les arêtes). N'utiliser qu'un quart de la surface disponible pour le centre d'intérêt n'est pas logique.

Deuxièmement, la procédure de dessin est trop simpliste. Dessiner les sous communautés indépendamment les unes des autres fait apparaître des propriétés fausses. Par exemple, dans

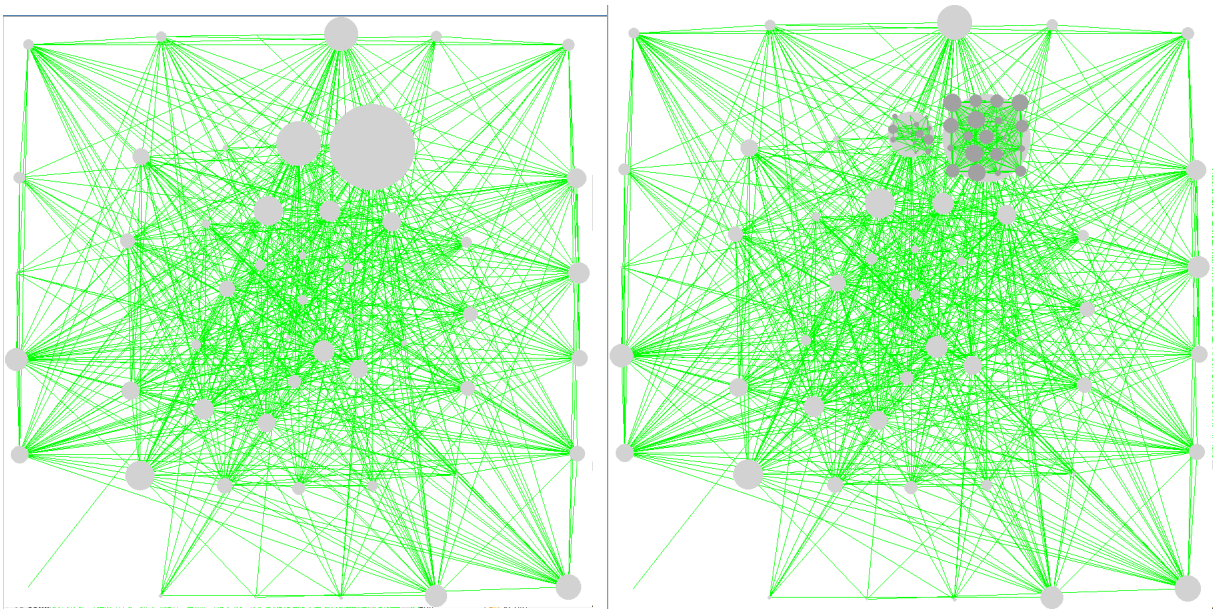

FIG. 12 – Rendus avec le graphe arxiv : il reste trop de communautés et le graphe est trop dense pour un rendu lisible. Sur la figure à droite on explore une communauté, il faudrait un zoom sur ce point d'intérêt pour voir quelque chose.

le cas d'un graphe "fil" à 4 sommets $1, 2, 3, 4$ avec comme liens $\{(1,2);(1,4);(3,4)\}$. Une décomposition en communautés pourrait être : $\{\{1,2\};\{3,4\}\}$ ce qui serait rendu comme sur le figure 13. Le dessin fait apparaître une proximité entre les nœuds 2 et 3 et un éloignement entre les nœuds 1 et 4 qui sont totalement faux. Après dénombrement, il y a entre 55 et 70% des nœuds qui sont reliés par au moins un lien à un nœud d'une autre communauté. L'influence des autres communautés risque donc d'être forte.

La méthode d'affichage suit l'arbre de séparation. On affiche les nœuds du premier niveau, et un clic signifie afficher les fils. Afin de connaître le nombre de nœuds qu'il faudra afficher en moyenne, nous avons calculé la distribution P_k du nombre de fils de chaque nœud dans l'arbre. Ainsi, P_k est la probabilité qu'un nœud de l'arbre ait k fils. Un résultat typique est représenté en échelle logarithmique sur la figure 14. On constate que les points sont disposés suivant une droite, ce qui est caractéristique d'une distribution en loi de puissance. Cela implique qu'il existe des nœuds ayant un très grand nombre de fils, et donc entraînant de dessiner des sous graphes ayant beaucoup de nœuds (la probabilité que cela arrive serait minime avec une loi à décroissance exponentielle). Les dessiner va être soit impossible, soit illisible.

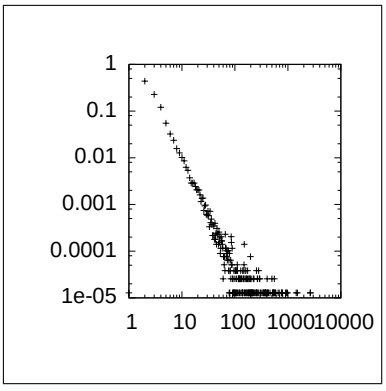

FIG. 14 – Distribution du nombre de fils sur Webndu

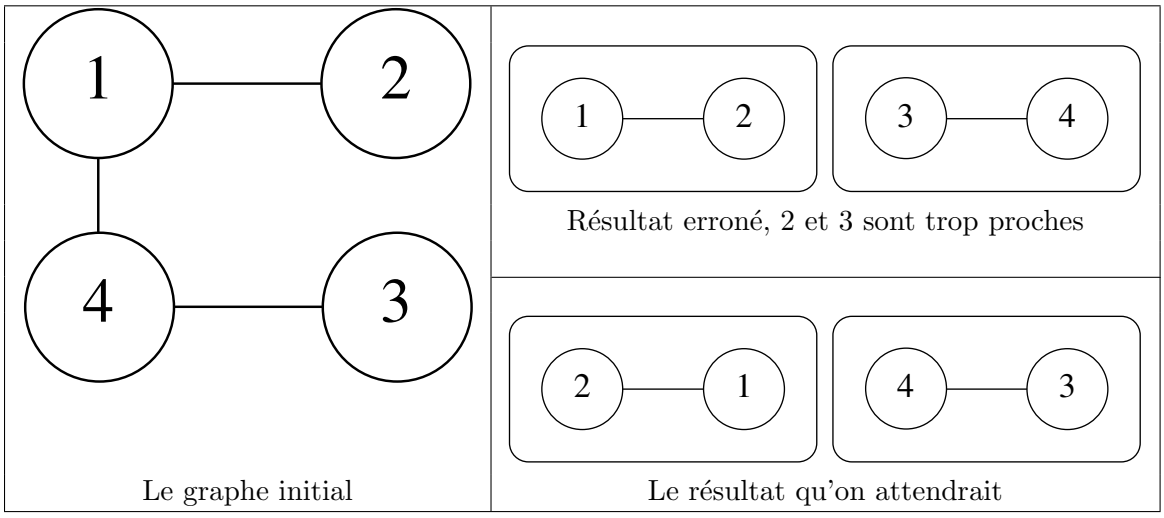

FIG. 13 – Exemple de placement des nœuds d'une communauté indépendamment de l'autre faisant apparaître des propriétés erronées

5 Conclusion

Nous avons étudié deux structures hiérarchiques utilisées pour définir une structure multi-échelle dans un graphe. Chaque niveau de ces arbres permettait d'extraire une simplification du graphe initial. Ensuite, la série de simplifications a été étudiée expérimentalement afin de comprendre l'évolution de certaines grandeurs classiques à chaque niveau. La première structure provient d'un arbre construit durant un calcul de communautés, et s'avère pertinente. Les grandeurs évoluent de manière relativement stable vers un graphe bien plus petit et dense. Les propriétés de petits mondes et de scale free sont conservées, et le clustering évolue régulièrement. Lorsque l'on étudie les sous graphes du graphe initial correspondant à une de ces communautés, on obtient un graphe ayant les mêmes caractéristiques que le graphe initial mais étant plus dense. La seconde décomposition, en revanche, en divisant sous graphe par sous graphe le graphe initial, semble moins intéressante d'un point de vue global. Les communautés trouvées ont une modularité bien inférieure, et les grandeurs usuelles évoluent différement suivant le graphe initial. Quand on extrait récursivement le sous graphe du graphe initial correspondant à la plus grande communauté, on constate néanmoins que ce sous graphe a lui même une décomposition de forte modularité. Cela peut être signe d'une structure multi-échelle où les communautés ne seraient pas un regroupement d'autres communautés plus petites, mais la séparation de celles ci en d'autres groupes totalement différents, ou bien de l'existence de sous communautés seulement dans certaines. Il est délicat de conclure en l'absence d'étiquetages des graphes permettant d'identifier les causes, au-delà de la maximisation de la modularité, de la séparation en communautés (un tel étiquetage pourrait être le laboratoire, l'équipe, le pays, ou la langue par exemple sur Arxiv).

Comme deuxième objectif, nous avons essayé de dessiner les graphes à l'aide de cette structure multi-échelle. Les nœuds du graphe initial sont regroupés en leur communauté obtenue par maximisation de la modularité, de façon à ne représenter qu'une schématisation du graphe. Ensuite, l'utilisateur peut explorer plus précisément certaines communautés. Le manque de temps ne nous a pas permis de finir ce travail, beaucoup de problèmes ne sont pas encore résolus : vouloir dessiner une communauté sans tenir compte de ce qui l'entoure,

d'une part, est trop naïf et fait apparaître des propriétés fausses et, d'une autre part, on doit dessiner des graphes très denses et potentiellement très gros. Il faudrait une modification de l'algorithme de placement de nœuds pour tenir compte de contraintes extérieures au sous graphe dessiné et implémenter des méthodes de dessins plus sophistiquées comme le rendu "fish-eye" afin de rendre l'outil exploitable. Les graphes entre communautés étant très denses, il faudrait sans doute en plus filtrer les arêtes en fonction de critères à définir.

Ce stage m'a permis d'étudier plus en détail la problématique de l'étude des réseaux au sens large (par opposition aux uniques réseaux d'ordinateurs). Ces sujets recouvrent des disciplines diverses telles que la physique, la sociologie et bien sûr l'informatique. Les origines multiples des problèmes soulevés et des solutions proposées rendent ce domaine vraiment passionnant, et je tiens à remercier mes encadrants qui ont su me donner des directions tout en me laissant une part de liberté et l'équipe du LIP6 que j'ai côtoyée pour son accueil. Ce fut un plaisir de travailler en leur compagnie.

Références

[1] R. Albert, *Statistical mechanics of complex networks*. PhD thesis, UNIVERSITY OF NOTRE DAME, 2001.

[2] M. Latapy and C. Magnien, "Measuring Fundamental Properties of Real-World Complex Networks," *ArXiv Computer Science e-prints*, Sept. 2006.

[3] S. Milgram, "The small world problem," *Psychology Today*, vol. 1, p. 61, 1967.

[4] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, June 1998.

[5] R. Albert, H. Jeong, and A.-L. Barabasi, "The internet's achilles' heel : Error and attack tolerance of complex networks," *Nature*, vol. 406, pp. 200–0, 2000.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking : Bringing order to the web," tech. rep., Stanford Digital Library Technologies Project, 1998.

[7] R. Pastor-Satorras and A. Vespignani, "Epidemic spreading in scale-free networks," *Phys. Rev. Lett.*, vol. 86, pp. 3200–3203, Apr 2001.

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

[9] R. Guimerà and L. A. Nunes Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, pp. 895–900, Feb. 2005.

[10] K. Astrup Eriksen, I. Simonsen, S. Maslov, and K. Sneppen, "Modularity and Extreme Edges of the Internet," *ArXiv Condensed Matter e-prints*, Nov. 2002.

[11] V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri, "Extending modularity definition for directed graphs with overlapping communities," *ArXiv e-prints*, vol. 801, Jan. 2008.

[12] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Science*, vol. 99, pp. 7821–7826, June 2002.

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

[14] S. Fortunato and C. Castellano, "Community structure in graphs," *Encyclopedia of Complexity and System Science*, 2008.

[15] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, p. 026113, Feb 2004.

[16] P. Pons and M. Latapy, "Computing communities in large networks using random walks (long version)," *ArXiv Physics e-prints*, Dec. 2005.

[17] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, vol. 74, no. 3, p. 036104, 2006.

[18] A. Clauset, C. Moore, and M. E. J. Newman, "Finding community structure in very large networks," *Physical Review E*, vol. 70, pp. 066111–+, Dec. 2004.

[19] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of community hierarchies in large networks," *ArXiv e-prints*, vol. 803, Mar. 2008.

[20] R. Albert, H. Jeong, and A.-L. Barabasi, "The diameter of the world wide web," *Nature*, vol. 401, p. 130, 1999.

[21] M. E. J. Newman, "The structure of scientific collaboration networks," Working Papers 00-07-037, Santa Fe Institute, July 2000.

[22] M. Latapy, F. Aidouni, and C. Magnien, "Capture à grande échelle de traffic edonkey," in *Algotel*, 2008.

[23] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *Journal of Statistical Mechanics : Theory and Experiment*, vol. 9, pp. 8–+, Sept. 2005.

[24] A. Lancichinetti, S. Fortunato, and J. Kertesz, "Detecting the overlapping and hierarchical community structure of complex networks," 2008.

[25] I. Herman, G. Melançon, and M. Marshall, "Graph visualization and navigation in information visualization : A survey," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 6, pp. 24–43, Jan-Mar 2000.

[26] A. Quigley and P. Eades, "Fade : Graph drawing, clustering, and visual abstraction," in *GD '00 : Proceedings of the 8th International Symposium on Graph Drawing*, (London, UK), pp. 197–210, Springer-Verlag, 2001.

[27] D. Harel and Y. Koren, "Drawing graphs with non-uniform vertices," in *Proceedings of Working Conference on Advanced Visual Interfaces (AVI'02)*, pp. 157–166, 2002.

[28] P. Eades and Q.-W. Feng, "Multilevel visualization of clustered graphs," in *Proc. Graph Drawing, GD*, no. 1190, (Berlin, Germany), pp. 101–112, Springer-Verlag, 18–20 1996.

[29] A. Y. Wu, M. Garland, and J. Han, "Mining scale-free networks using geodesic clustering," in *KDD '04 : Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, (New York, NY, USA), pp. 719–724, ACM, 2004.

[30] D. Auber, Y. Chiricota, F. Jourdan, and G. Melançon, "Multiscale visualization of small world networks," *infovis*, vol. 00, p. 10, 2003.

[31] M. Sarkar and M. H. Brown, "Graphical fisheye views of graphs," in *CHI '92 : Proceedings of the SIGCHI conference on Human factors in computing systems*, (New York, NY, USA), pp. 83–91, ACM, 1992.

Annexes

Graphe initial									
	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	
	9377	24107	0.0005	66	0.241	0.650	6.409	5.14	

Graphe entre communautés niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	60	653	0.335	54	0.61	0.67	1.64	21.77	0.812
2	108	1015	0.157	82	0.402	0.53	1.92	18.80	0.809
3	434	2545	0.022	179	0.15	0.32	2.68	11.73	0.77
4	2139	7964	0.0025	364	0.072	0.305	4.160	7.45	0.57

Graphe à l'intérieur de la plus grande communauté niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	
1	986	2808	0.0057	38	0.187	0.556	4.14	5.69	
2	986	2808	0.0057	38	0.187	0.556	4.14	5.69	
3	947	2702	0.0060	38	0.187	0.556	4.25	5.71	
4	403	1335	0.016	33	0.179	0.453	3.68	6.63	

Graphe extrait niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	60	653	0.335	54	0.610	0.668	1.73	21.767	0.813
2	579	3590	0.0180	81	0.173	0.245	3.51952	12.40	0.725
3	1905	8054	0.003	62	0.158	0.295	4.71087	8.46	0.533
4	4975	15338	0.0011	47	0.218	0.532	6.05389	6.17	0.257
5	8446	22437	0.00061	64	0.244	0.633	6.4876	5.31	0.044
6	9345	24040	0.00052	66	0.241	0.649	6.62363	5.145	0.001

Graphe extrait récursivement niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	986	2808	0.0058	38	0.187	0.556	4.430	5.696	0.64
2	79	165	0.053	19	0.293	0.626	3.67	4.177	0.5/
3	18	34	0.22	8	0.438	0.622	2.51	3.778	0.38
4	5	5	0.5	3	0.5	0.583	1.24	2	0.22

FIG. 15 – Résultats pour Arxiv

Graphe initial									
	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	
	226174	1.75e+06	6.86e-05	1434	0.32	0.587	5.602	15.52	

Graphe entre communautés niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	Modul- arité
1	351	2407	0.033	254	0.60	0.94	2.27	13.72	0.657787
2	362	2811	0.037	265	0.60	0.92	2.07	15.53	0.657776
3	581	8291	0.0458	473	0.37	0.83	2.15	28.54	0.657099
4	3753	51129	0.0067	2706	0.07	0.623	2.16	27.245	0.644294
5	28076	192887	0.0004	11904	0.009	0.47	2.90	13.74	0.58093

Graphe à l'intérieur de la plus grande communauté niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	
1	31605	295895	0.00059	603	0.186	0.568	4.19	18.72	
2	31605	295895	0.00059	603	0.186	0.56	4.03	18.72	
3	31605	295895	0.00059	603	0.186	0.56	4.18	18.72	
4	31540	295734	0.00059	603	0.186	0.56	4.02	18.75	
5	26864	272736	0.00076	599	0.185	0.55	3.91	20.30	

Graphe extrait niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	Modul- arité
1	351	2407	0.033	254	0.60	0.944	2.08	13.72	0.658
2	3093	115121	0.024	1417	0.300	0.54	2.83	74.44	0.587
3	15999	333378	0.002	2740	0.122	0.325	3.05	41.67	0.477
4	48629	593497	0.00047	1593	0.10	0.406	4.07	24.41	0.323
5	116366	1.09e+06	0.00015	1837	0.241	0.535	4.55	18.77	0.144
6	189999	1.52e+06	8.336e-05	1434	0.287	0.578	4.996	16.03	0.038
7	221379	1.72e+06	7.013e-05	1434	0.319	0.586	5.15	15.55	0.004
8	225927	1.75e+06	6.867e-05	1434	0.323	0.587	5.355	15.52	0.0001

Graphe extrait récursivement niveau par niveau									
ni- veau	nombre noeuds	nombre liens	densité	degré max	transi tivité	clus- tering	distance moyenne	degré moyen	Modul- arité
1	31605	295895	0.0006	603	0.185561	0.56	3.988	18.72	0.467
2	4399	50904	0.0053	382	0.204	0.648	3.121	23.1434	0.446
3	1007	8471	0.0167	243	0.263	0.777	2.87	16.82	0.42
4	148	825	0.0758	79	0.375	0.873	2.316	11.15	0.33
5	28	93	0.246	24	0.594	0.906	1.889	6.643	0.343
6	9	30	0.833	8	0.934	0.952	1.033=	6.667	0.02

FIG. 16 – Résultats pour Fichiers pair à pair

Graphe initial									
	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	
	439750	3.898e+06	4.031e-05	1052	0.374	0.765	5.876	8.863	

Graphe entre communautés niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	390	8483	0.107	316	0.858	0.926	2.088	21.751	0.759
2	414	10102	0.113	340	0.805	0.887	1.913	24.401	0.759
3	945	31127	0.068	854	0.379	0.664	1.968	32.939	0.757
4	7331	118768	0.004	5678	0.055	0.473	2.219	16.201	0.740
5	57619	382485	0.0002	22844	0.0048	0.531	3.054	6.638	0.645

Graphe à l'intérieur de la plus grande communauté niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	
1	59970	355451	0.00019	573	0.2789	0.747	4.70	5.93	
2	59924	355261	0.00019	572	0.2789	0.747	4.72	5.93	
3	56312	342441	0.00022	572	0.2795	0.747	4.64	6.08	
4	25660	185936	0.00056	424	0.2659	0.698	4.27	7.25	

Graphe extrait niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	390	8483	0.107	316	0.858	0.926	2.15	21.75	0.759
2	5566	187108	0.0118	2654	0.143	0.441	2.909	33.62	0.684
3	29956	480680	0.001	3763	0.105	0.370	3.857	16.05	0.536
4	110110	1.1e+06	0.00017	2275	0.146	0.627	4.539	9.991	0.334
5	269417	2.33e+06	6.26e-05	2188	0.233	0.744	5.001	8.630	0.135
6	396439	3.52e+06	4.46e-05	2206	0.343	0.7614	5.099	8.89	0.024
7	435554	3.87e+06	4.07e-05	1248	0.373	0.7640	5.301	8.877	0.002
8	439614	3.90e+06	4.03e-05	1052	0.374	0.7644	5.053	8.864	0.000
9	439748	3.90e+06	4.031e-05	1052	0.374	0.765	5.605	8.863	0.000

Graphe extrait récursivement niveau par niveau									
niveau	nombre noeuds	nombre liens	densité	degré max	transitivité	clustering	distance moyenne	degré moyen	Modularité
1	59970	355451	0.000197	573	0.279	0.747	4.47792	5.93	0.66
2	6514	40520	0.00191	298	0.35	0.797	3.97	6.22045	0.6337
3	1098	5199	0.0086	147	0.35	0.88	3.14	4.73	0.6332
4	177	746	0.048	55	0.60	0.91	2.54	4.21	0.476
5	45	162	0.16	25	0.79	0.878	2.44	3.6	0.28
6	15	80	0.76	14	0.97	0.979	1.04	5.33	0.03

FIG. 17 – Résultats pour Clients pair à pair