Déplier la structure communautaire d'un réseau en mesurant la proximité aux représentants de communauté

Maximilien Danisch, Jean-Loup Guillaume, Bénédicte Le Grand

Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 4 place Jussieu 75005 Paris.
L3I, Université de La Rochelle, Av. Michel Crepeau, 17042 La Rochelle, France
CRI, Université Paris 1 Panthéon-Sorbonne, 90 rue de Tolbiac, 75013 Paris, France

RÉSUMÉ. *Nous proposons un algorithme pour déplier la structure communautaire des grands graphes de terrain. L'algorithme est basé sur la détection de la communauté de chaque représentant communautaire : nœud contenu dans une seule communauté et important en son sein. Cette détection est faite avec une approche à base de mesure de proximité développée récemment. Par comparaison avec d'autres méthodes de l'état de l'art nous montrons que notre algorithme a des performances équivalentes voire meilleures et est capable de traiter les plus grands graphes de terrain.*

ABSTRACT. *How to find all overlapping communities in a complex network? That is, how to find all relevant groups of nodes in a linked dataset? No entirely satisfying solution to that important problem exists, having a criterion to decide which group is relevant and finding quickly these groups in large networks are bottlenecks. We found that in many networks the number of these groups is limited and that there exist, for each group, at least one node that can characterize it by itself: a node belonging only to that group and important within it. We call such a node a* community representative. *We develop an algorithm to find these overlapping communities. The community detection is done through measuring the* proximities *of all nodes to the representatives and then finding irregularities in the decrease of these values reflecting the presence of relevant groups. We show that our approach handles very large real-world networks and have comparable or even better performances compared to state of the art methods.*

MOTS-CLÉS : *communautés recouvrantes, communautés locales, mesures de proximité*

KEYWORDS: *overlapping communities, local communities, proximity measure*

1. Introduction

Parmi les problèmes liés à la détection de communautés, le problème le plus général et le plus difficile reste la détection de tous les groupes de nœuds pertinents dans un grand réseau réel, c'est-à-dire la détection de toutes les communautés recouvrantes. En effet, bien que des solutions à ce problème aient été proposées (Xie *et al*., 2011), par exemple la percolation de k-cliques (Palla *et al*., 2005; Kumpula *et al*., 2008; Reid *et al*., 2012), BigClam (Yang, Leskovec, 2013), la partition de liens (Ahn *et al*., 2010), etc., aucune solution ne fait l'unanimité et n'est entièrement satisfaisante. La conception d'un critère pour décider si un groupe de nœuds est pertinent et la détection rapide de ces -potentiellement très nombreux- groupes sont les principaux goulots d'étranglement scientifiques. Nous proposons ici une méthode de détection de communautés recouvrantes faisant face à ces deux problèmes.

Nous montrons dans la section 2 qu'en général et contrairement à ce que l'on pourrait penser, le nombre de groupes pertinents est du même ordre de grandeur, voire plus petit que le nombre de nœuds dans le réseau. Ainsi, si l'on peut détecter chacun de ces groupes en un temps constant (ne dépendant pas de la taille du réseau), alors la conception d'un algorithme rapide (linéaire) pouvant traiter les réseaux réels les plus grands est possible.

Dans la section 3 nous montrons qu'en général, dans les réseaux réels, une communauté a un représentant, c'est-à-dire un nœud qui appartient uniquement à cette communauté et qui est important en son sein. Nous montrons également, dans cette même section, comment on peut détecter rapidement la communauté d'un représentant avec une "approche mesure de proximité" qui consiste à mesurer la proximité de tous les nœuds du réseau au représentant puis à chercher une irrégularité dans la décroissance de ces valeurs de proximité afin de détecter la limite de la communauté. Cette "approche mesure de proximité" requiert effectivement un temps ne dépendant pas de la taille du réseau.

Un algorithme basé sur la détection de la communauté ego-centrée des représentants et linéaire en fonction de la taille du réseau est alors concevable. Ces communautés détectées sont alors étiquetables avec l'étiquette de leur représentant. Nous présentons cet algorithme dans la section 4. Nous y présentons également une procédure de nettoyage permettant d'éliminer les communautés trouvées plusieurs fois, dont la complexité est également linéaire dans les réseaux réels. La méthode de nettoyage consiste, lorsque deux communautés sont jugées trop similaires, à enlever la communauté évaluée comme la moins pertinente. La condition pour avoir un temps linéaire pour cette procédure de nettoyage est que le nombre de communautés partageant un nœud donné ne dépende pas de la taille du réseau (nous faisons le calcul seulement pour les paires de communautés partageant au moins un nœud). Nous montrons que cette condition est effectivement satisfaite dans les réseaux réels. Notre algorithme s'inscrit donc dans le cadre général des méthodes consistant à passer d'une structure communautaire locale à une structure communautaire globale (Kanawati, 2014).

Dans la section 5, nous comparons la performance de notre méthode à celles de l'état de l'art sur de grands réseaux réels. Nous discutons les différents critères de comparaison entre des communautés recouvrantes détectées et des communautés recouvrantes "vérité de terrain" et choisissons ceux qui nous paraissent les plus pertinents. Selon ces critères, notre méthode obtient des résultats comparables, voire meilleurs que les méthodes de l'état de l'art sur ces grands réseaux réels. Nous montrons également que notre méthode est capable de traiter les plus grands réseaux contrairement aux autres méthodes auxquelles nous l'avons comparée.

Finalement, nous concluons et présentons les perspectives d'amélioration de notre méthode dans la section 6. Une voie d'amélioration certaine consiste en l'examen de communautés bi-ego-centrées et pas seulement ego-centrées afin de détecter des communautés sans représentant unique. Le code source de notre algorithme est disponible à l'url suivante : http://bit.ly/maxdan94.

2. Discussion sur le nombre de communautés dans un réseau

Le nombre maximum de groupes dans un réseau de n nœuds est de $2^n - 1$. Il est donc *a priori* possible que le nombre de communautés recouvrantes dans un réseau soit également de cet ordre de grandeur. Si tel était le cas, tout algorithme cherchant à détecter toutes les communautés recouvrantes d'un graphe serait voué à l'échec : (i) le temps de calcul serait trop important, (ii) le stockage des résultats serait impossible et (iii) la structure découverte serait complètement inutile. Remarquons que, même si ce nombre de communautés était en $\Omega(n^2)$, les trois propriétés précédentes seraient également vraies pour de très grands graphes. Avant d'aller plus loin, nous devons donc essayer de nous convaincre que cela n'est pas le cas !

Une communauté étant identifiée par des régions denses (en termes de liens) avec des régions moins denses autour, il semble difficile d'atteindre ce nombre de $2^n - 1$. En effet, parvenir à dessiner des régions denses et des régions moins denses entre tous ces groupes semble difficile. Une meilleure borne supérieure serait peut-être de considérer le nombre maximum de cliques maximales dans un réseau de n nœuds : les cliques représentant les régions denses à l'extrême et l'absence de liens, les régions moins denses. Ce nombre est de l'ordre de $3^{\frac{n}{3}}$ (Moon, Moser, 1965) qui est plus petit que $2^n - 1$, mais reste encore bien trop élevé. Ajoutons que les réseaux réels sont creux et ont une dégénérescence [1] notée d peu élevée. Une autre borne supérieure du nombre maximum de cliques maximales en fonction de la dégénérescence est $(n - d)3^{\frac{d}{3}}$ (Eppstein *et al.*, 2010) : nous approchons ici du nombre de cliques maximales, linéaire en fonction du nombre de nœuds. Remarquons cependant que ce modèle n'autorise pas des communautés hiérarchiques et cette valeur peut donc difficilement être qualifiée de borne supérieure.

Pour aller plus loin, nous proposons au lecteur de se situer comme un nœud dans son propre réseau social de connaissances et de réfléchir autour de quelques formules. Notons C le nombre de communautés dans un réseau, on a clairement $C < n.c$ où c est le nombre moyen de communautés auxquelles un nœud appartient car chaque communauté contient plusieurs nœuds. Nous demandons à présent au lecteur d'évaluer à combien de communautés il appartient (amis, famille, collègues, etc.). Cette question peut sembler floue et difficile sans définition formelle de ce qu'est une communauté, mais le but est seulement d'avoir un ordre de grandeur de c et de se convaincre que ce nombre est petit devant le nombre de nœuds dans le réseau (c'est-à-dire, ici le nombre de personnes sur la Terre, de l'ordre de 7 milliards en 2015). 10, 30 ou même 100 nous semblent toutes être des réponses légitimes. Maintenant, en se considérant comme un nœud moyen du réseau, on obtient donc que C est de l'ordre de n.

On peut aller encore plus loin à l'aide de la formule exacte suivante : $C = n\frac{c}{t}$. Où t dénote la taille moyenne d'une communauté. En effet $\frac{c}{t}$ est le nombre moyen de communautés

1. Aussi appelée nombre de k-core. La dégénérescence d'un graphe G est le nombre maximum d tel qu'il existe un sous-graphe de G ne contenant que des nœuds de degré au moins d.

associées à un nœud et cette formule est évidente puisque, par définition, on a $c = \frac{Ct}{n}$. Nous invitons maintenant le lecteur à réfléchir encore une fois pour évaluer la taille moyenne de ses communautés. C'est encore une question difficile, car une communauté n'est pas une clique et donc les communautés d'une personne ne contiennent pas que des contacts directs (elles contiennent a priori aussi des gens que la personne ne connaît pas), mais encore une fois seul un ordre de grandeur importe. Faire le ratio entre son nombre moyen de communautés et la taille moyenne de ses communautés donnant une approximation de $\frac{c}{t}$. Nous avons l'intuition que c et t sont tous deux petits devant la taille du réseau et semblent également être du même ordre de grandeur, avec possiblement t plus grand que c. Ainsi le nombre de communautés dans un réseau semble être de l'ordre du nombre de nœuds dans le réseau : $C = O(n)$.

Afin de justifier plus avant que ce nombre de communautés est de l'ordre du nombre de nœuds dans le réseau, voire plus petit, nous avons effectué des statistiques sur différents réseaux. Ces résultats sont exposés dans le tableau 1 où nous avons comparé le nombre de nœuds, le nombre de liens et le nombre de communautés "vérité de terrain" pour différents réseaux réels (Yang, Leskovec, 2015). Nous voyons ici que le nombre de communautés est toujours petit devant le nombre de liens ou de nœuds, sauf pour le réseau Orkut où le nombre de communautés est de l'ordre de deux fois le nombre de nœuds.

Tableau 1. Nombre de nœuds, nombre de liens, le nombre de communautés, taille moyenne des communautés et nombre moyen de nœud par communauté "vérité de terrain" pour différents réseaux réels.

Réseau	n	e	C	t	c	$C.t = c.n$
LiveJournal	3 997 962	34 681 189	287 512	22.63	1.60	6 414 555
Friendster	65 608 366	1 806 067 135	957 154	23.14	0.34	22 151 543
Orkut	3 072 441	117 185 083	6 288 363	14.16	28.98	89 053 454
Youtube	1 134 890	2 987 624	8 385	13.50	0.10	113 200
DBLP	317 080	1 049 866	13 477	53.41	2.27	719 820
Amazon	334 863	925 872	151 037	19.38	8.74	2 927 342

Il semble donc, par construction du réseau ou par propriété intrinsèque des communautés, que le nombre de celles-ci soit de l'ordre du nombre de nœuds dans le graphe, tout en étant inférieur à ce dernier. Il serait donc possible d'arriver à un algorithme qui, dans un réseau issu du terrain, retournerait en temps linéaire toutes les communautés du réseau (à condition de pouvoir les trouver en un temps constant).

3. Détection de la communauté d'un représentant

Comme montré dans (Danisch *et al.*, 2012), si un nœud d'intérêt n'appartient qu'à une seule communauté et est important en son sein, c'est-à-dire si le nœud est un représentant de communauté, alors sa communauté peut être identifiée en utilisant une "approche mesure de proximité". Cette approche consiste en trois étapes : (i) calculer la proximité de chaque nœud du réseau au nœud d'intérêt en utilisant une mesure de proximité ad hoc, (ii) trier ces valeurs par ordre décroissant, (iii) chercher une structure en "plateau / décroissance / plateau" et identifier les nœuds avant la forte décroissance comme correspondant à la communauté du nœud d'intérêt.

En pratique, il suffit de chercher la plus grande pente dans la courbe proximité VS classement et de sélectionner les nœuds avant cette plus grande pente comme la communauté du nœud d'intérêt. La valeur de la plus grande pente peut permettre d'estimer dans quelle mesure la communauté est claire et à quel point le nœud d'intérêt en est un bon représentant.

Nous avons à disposition un grand choix de mesures de proximité comme le montre les articles d'état de l'art (Cohen *et al.*, 2012; Wilson *et al.*, 2013). Cependant, comme nous allons le voir, nous aurons besoin de répéter l'expérience pour chaque nœud du réseau. Nous avons donc besoin ici d'une mesure de proximité très rapide à calculer et dont le support (le nombre de valeurs non nulles) des valeurs est faible. Ainsi, nous choisissons l'approximation du pagerank enraciné (sur le nœud d'intérêt) et normalisé par les degrés proposé dans (Andersen *et al.*, 2006). Nous avons remarqué que, jusqu'à une certaine limite (dépendant du réseau) plus la probabilité α de revenir au nœud de départ est petite, meilleure sont les résultats, mais plus le temps de calcul augment. Nous avons choisi la valeur par défaut $\alpha = 0.01$.

Dans les réseaux réels, les communautés semblent en général avoir un représentant, cependant prouver cette assertion ou tout du moins évaluer son niveau de véracité est difficile. Pour cela, nous avons travaillé avec le réseau *Wikipédia 2012* (Danisch *et al.*, n.d.), sélectionné des catégories qui correspondent à nos communautés vérité de terrain et essayé de trouver pour chaque communauté des nœuds importants.

La figure 1 montre ces résultats obtenu pour le nœud "Tayo Koki" (resp. "Cruiserweight", "Magnus Carlsen") et sa communauté naturelle "Sumo" (resp. "Boxing", resp. "Chess"). Comme nous pouvons le voir pour chacun de ces exemples, la plus grande pente semble coïncider avec la fin de la communauté (en suivant le classement en fonction de la proximité). Elle montre Ã©galement le résultat obtenu avec le nœud "Chess Boxing" et les communautés "Chess" et "Boxing". Dans ce cas, la méthode ne marche pas car le nœud "Chess Boxing" n'est pas central à une communauté et appartient à plusieurs communautés.

Le tableau 2 montre les résultats associés aux figures 1 ainsi que d'autres résultats pour ce même réseau *Wikipédia 2012* : nous avons comparé l'ensemble des nœuds situés avant la plus grande pente avec l'ensemble des nœuds dans la catégorie du nœud d'intérêt en termes de score F_1. Pour deux communautés A et B, on a $F_1(A, B) = 2\frac{|A \cap B|}{|A|+|B|}$ [2]. La méthode est efficace lorsqu'un représentant (c'est-à-dire, un nœud uniquement dans la communauté en question et important en son sein) est choisi, mais ne marche pas lorsque le nœud candidat n'est pas un représentant (dernier exemple à chaque fois). Nous avons également renseigné la taille et le score F_1 obtenus par le nœud d'intérêt et l'ensemble de ses voisins. Remarquons également que la valeur de la plus grande pente semble indiquer dans quelle mesure la communauté est bien dessinée et à quel point le nœud d'intérêt est central à la communauté. Nous voyons que pour les trois communautés sélectionnées nous avons pu trouver un représentant et même plusieurs. Cela nous permet donc d'envisager un algorithme pour calculer toutes les communautés recouvrantes d'un graphe que nous présentons maintenant.

2. Ce score peut être vu comme la moyenne harmonique de la précision et du rappel : $F_1(A, B) = \frac{2precision(A,B)rappel(A,B)}{precision(A,B)+rappel(A,B)}$ avec $precision(A, B) = \frac{|A \cap B|}{|B|}$ et $rappel(A, B) = \frac{|A \cap B|}{|A|}$.

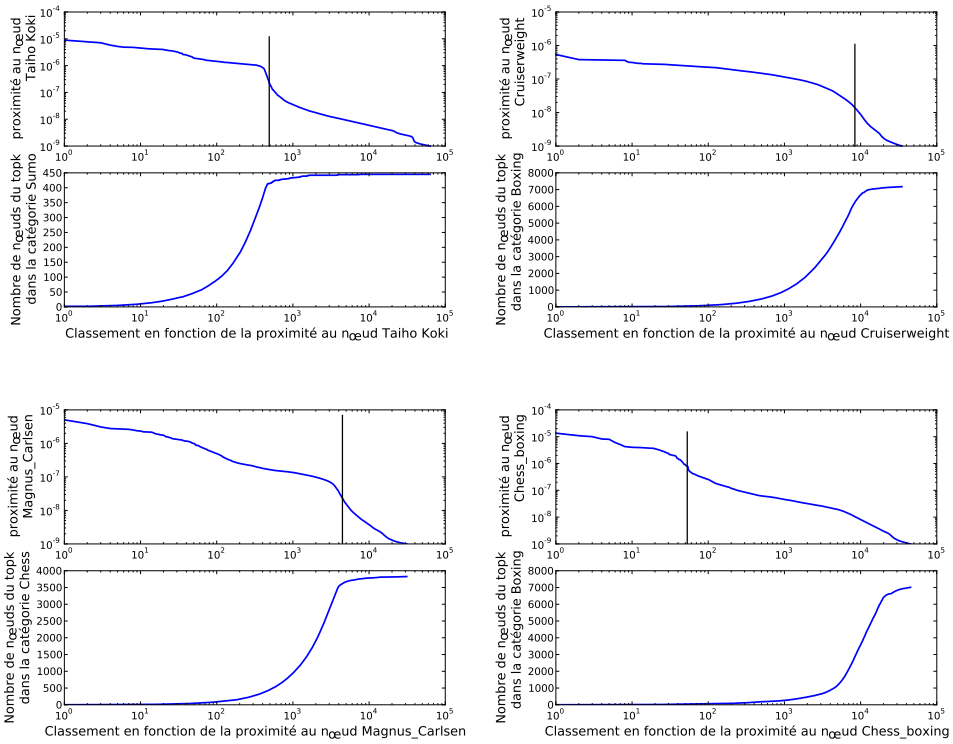

FIGURE *1. Résultats obtenus dans le réseau Wikipédia 2012. La barre verticale identifie la position de la plus grande pente : pour les trois premiers exemples, elle correspond à la transition "dans la communauté/hors de la communauté"*

Tableau 2. Résultats obtenus sur le réseau Wikipédia 2012.

Catégorie	candidat représentant	pente maximale	nombre de noeuds	F_1 score
Chess (3850 nœuds)	Chess	3.46	4179	0.89
	Magnus Calsen	3.69	4023	0.90
	Queens Gambit	4.37	4051	0.91
	World Chess Championship	4.04	4107	0.90
	Chess Boxing	1.35	48	0.00
Sumo (445 nœuds)	Sumo	3.03	508	0.88
	Taiho Koki	5.97	440	0.91
	Yokozuna	3.07	437	0.88
	Lyoto Machida	1.94	6458	0.08
Boxing (7289 nœuds)	Boxing	2.42	7572	0.79
	Cruiserweight	2.21	7572	0.78
	Vitali Klitschko	1.66	176	0.02
	Chess Boxing	1.34	48	0.01

4. Algorithme

4.1. Calcul des groupes

Notre algorithme prend en entrée un graphe et une mesure de proximité et retourne en sortie un ensemble de groupes de nœuds avec, pour chacun, un représentant (son étiquette) et un score évaluant sa pertinence en tant que communauté.

Pour chaque nœud du graphe, notre algorithme : (i) calcule la proximité à tous les autres nœuds du graphe, (ii) trie ces valeurs de proximité par ordre décroissant, (iii) calcule la plus grande pente en échelle logarithmique, (iv) met de côté l'ensemble de nœuds situés avant la plus grande pente (la communauté) en lui associant le sommet avec lequel il a été trouvé (son représentant) et la valeur de la plus grande pente (évaluant sa pertinence en tant que communauté).

Comme nous l'avons vu précédemment, en utilisant le pagerank enraciné approximé (Andersen *et al.*, 2006), le calcul de la proximité d'un nœud à tous les nœuds se fait en temps constant (ne dépendant pas de la taille du graphe) et donc le support de ces valeurs ne dépend également pas de la taille du graphe. Ainsi le temps de cet algorithme est bien linéaire en fonction de la taille du graphe.

4.2. Nettoyage des groupes

Il est très probable que plusieurs nœuds aient donné lieu à des groupes de nœuds très similaires. Cela se produit, par exemple, si deux nœuds sont des représentants de la même communauté. Il est donc important de nettoyer ces groupes avant de retourner les communautés finales. Notre solution à ce problème est la suivante : si deux groupes sont trop similaires, nous proposons d'enlever le moins pertinent (celui dont la plus grande pente a été la plus faible). Pour cela, nous évaluons la similarité entre deux groupes avec le score de F_1.

Si ce score est supérieur à un certain seuil δ, alors nous enlevons la communauté avec la pente la plus faible. $\delta = 0.8$ ou 0.9 sont des valeurs judicieuses, l'idée étant d'enlever simplement des groupes trop similaires.

Pour garder un algorithme efficace, on ne peut pas se permettre de calculer la similarité entre tous les groupes deux à deux. En effet, étant donné qu'il y a un nombre linéaire de groupes, cela donnerait un algorithme en temps quadratique qui ne pourrait pas traiter les plus grands graphes. Afin de pallier ce problème, nous proposons de calculer la proximité seulement entre deux groupes partageant au moins un sommet. Le calcul pour trouver toutes les paires de groupes partageant au moins un sommet se fait en temps linéaire. En effet, il suffit d'initialiser un tableau avec une entrée pour chaque nœud du graphe contenant une liste chaînée vide, puis de balayer les sommets de chaque groupe et d'ajouter à la liste de l'entrée correspondante au nœud courant un pointeur vers la communauté courante. On obtient ainsi pour une entrée du tableau un pointeur pour chaque communauté contenant le nœud en question. Remarquons qu'une paire de communautés partageant au moins un nœud peut apparaître plusieurs fois (en fait elle apparaît un nombre de fois égal au nombre de nœuds que les deux groupes partagent), mais ceci ne pose pas de problème : le temps pour former cette structure reste essentiellement linéaire en $O(cn)$ avec n le nombre de nœuds et c le nombre moyen de communautés qu'a un nœud, qui est petit devant la taille du réseau comme nous l'avons vu.

Calculer le score F_1 entre deux communautés se fait en temps proportionnel à la somme du nombre de nœuds dans ces communautés, sachant que l'on n'est pas obligé de calculer ce score entre deux communautés si on l'a déjà fait auparavant (on peut s'en rappeler en utilisant une table de hachage).

Le temps total de cet algorithme de nettoyage est donc en $O(tc^2n)$ où t est la taille moyenne d'une communauté et c^2 désigne ici la moyenne du carré du nombre de communautés qu'a un nœud. En pratique t et c^2 sont très petits devant la taille du réseau et leur produit reste petit devant lui.

4.3. Améliorations algorithmiques

Remarquons que pour le calcul des proximités, la boucle sur les nœuds est entièrement parallélisable sans aucune perte de performance, i.e., utiliser n processeurs/cœurs divise le temps par n.

Remarquons également qu'il est inutile de calculer cette mesure de proximité pour un nœud très proche d'un nœud pour lequel les proximités ont déjà été calculées. En effet, si deux nœuds sont très proches, leurs proximités à tous les nœuds du réseau auront des valeurs similaires et la détection de la plus grande pente résultera en la détection de deux groupes de nœuds similaires. Ainsi une fois que nous avons calculé la proximité d'un nœud donné à tous les nœuds du réseau, nous pouvons passer le calcul pour les k nœuds les plus proches, k étant un paramètre. Pour une faible valeur de k, cette technique permet grossièrement de diviser le temps par k, le gain diminue ensuite. Si k est trop grand, on peut passer à côté de certaines communautés.

La construction de la structure pour le nettoyage (liste des communautés incorporant chaque nœud du graphe) est parallélisable (une légère perte en temps peut cependant être observée puisque l'on peut potentiellement écrire dans les mêmes entrées du tableau). Le calcul des communautés trop similaires à l'aide de cette structure est là encore parallélisable.

Le fait que des communautés apparaissent plusieurs fois (moyennant un peu de bruit) est aussi un indicateur de la pertinence d'une communauté, cette valeur correspond en fait au nombre de représentant qu'a la communauté. On remarque en pratique qu'enlever les communautés n'apparaissant qu'une seule fois améliore les performances de la méthode : ces communautés sont souvent des faux positifs.

5. Validation

5.1. Métriques

Afin de comparer la performance de différentes méthodes les unes par rapport aux autres, il existe plusieurs métriques pour évaluer la similarité de l'ensemble des communautés trouvées à celle d'un ensemble de communautés "vérité de terrain". Par exemple, la distance d'édition correspond au nombre minimum d'opérations élémentaires nécessaire pour transformer la structure communautaire trouvée en la structure communautaire "vérité de terrain". Les opérations élémentaires étant : (i) l'ajout d'un nœud à une communauté, (ii) la suppression d'un nœud d'une communauté, (iii) la création d'une communauté avec un nœud, (iv) la suppression d'une communauté avec un nœud.

Cependant cette mesure donne des résultats peu intuitifs : avoir trouvé (resp. ne pas avoir trouvé) une grosse communauté qui n'existe pas (resp. qui existe) est bien plus pénalisant qu'en avoir trouvé (resp. ne pas en avoir trouvé) une petite qui n'existe pas (resp. qui existe). Il existe beaucoup d'autres métriques telles que l'Information Mutuelle Normalisée ou l'indice de Ohméga comme détaillé dans (Malliaros, Vazirgiannis, 2013). Le calcul de ces métriques afin de passer à l'échelle n'est souvent pas trivial et peu également poser problème.

Nous nous orientons donc plutôt vers d'autres métriques plus intuitives et moins sensibles à ce genre de problèmes. Nous allons en présenter deux autres puis les combiner pour conduire à une métrique qui nous semble très pertinente. Notons $T = \{T_1, T_2, ..., T_t\}$ l'ensemble des communautés trouvées et $V = \{V_1, V_2, ..., V_v\}$ l'ensemble des communautés vérité de terrain. La métrique la plus immédiate est certainement la moyenne du score F_1 des communautés trouvées en comparant chacune d'elles avec la communauté la plus similaire parmi les communautés vérité de terrain : $\mathcal{F}_1(T, V) = \frac{1}{t} \sum_{i=1}^{t} \max_{j \in [\![1,v]\!]} F_1(T_i, V_j)$.

Cette valeur illustre combien chaque communauté trouvée est similaire à au moins une communauté vérité de terrain. Remarquons que si une seule communauté a été trouvée et qu'elle est exactement égale à une communauté vérité de terrain, alors la valeur de ce score est de 1 (valeur maximale) et ceci même si il y a beaucoup de communautés vérité de terrain (des communautés qui ne sont pourtant pas trouvées dans notre exemple). Ainsi, il est important d'associer cette métrique à une autre de manière à avoir un critère d'évaluation plus fiable.

Une métrique, similaire à la première (que l'on peut voir comme son symétrique), est la moyenne du score F_1 des communautés vérité de terrain en comparant chacune d'entre elles avec la communauté la plus similaire parmi les communautés trouvées : $\mathcal{F}_1(V, T) = \frac{1}{v} \sum_{i=1}^{v} \max_{j \in [\![1,t]\!]} F_1(V_i, T_j)$.

Cette valeur illustre combien chaque communauté vérité de terrain est similaire à au moins une communauté trouvée. Ici, inversement à la première métrique, si l'on a un très grand nombre de communautés trouvées, alors on aura plus de chances d'avoir un score élevé. En particulier un ensemble composé de tous les groupes possibles (les $2^n - 1$ groupes) aura un score de 1 (score maximal). Remarquons également que pour la métrique $\mathcal{F}_1(T, V)$ (resp. $\mathcal{F}_1(V, T)$), des communautés vérité de terrain (resp. trouvées) peuvent être associées à plusieurs communautés trouvées (resp. vérité de terrain). Il est possible de pallier ce problème en utilisant un algorithme de mariage pour maximiser la valeur finale et non pas chaque valeur indépendamment, cependant le temps de cet algorithme risque d'être supra-linéaire (en fonction du nombre de communautés totales) et de ne pas passer à l'échelle. Nous verrons que ce problème (utiliser plusieurs fois la même communauté) n'en n'est pas vraiment un si nous combinons les deux scores.

Afin d'avoir une métrique plus juste, il est possible de combiner ces deux métriques ; ceci a été proposé dans (Yang, Leskovec, 2013) où les auteurs proposent d'en faire la moyenne. Ceci ne nous paraît pas pertinent, car les deux ensembles "extrêmes" cités précédemment (un ensemble d'une seule communauté exactement égale à une communauté vérité de terrain et un ensemble composé de tous les groupes possibles) auront tous deux un score d'un peu plus de 0.5 alors qu'ils devraient tous deux avoir un score proche de zéro. Nous proposons donc de faire plutôt la moyenne harmonique (plutôt que la moyenne arithmétique) afin de satisfaire ce critère. Ceci est d'ailleurs la technique généralement adoptée pour d'autres scores, comme le score F_1 lui-même, si l'on fait l'amalgame entre $\mathcal{F}_1(T, V)$ et la sensi-

bilité et $\mathcal{F}_1(V,T)$ et la précision. Nous proposons donc d'utiliser la métrique suivante : $\mathcal{S}(T,V) = \frac{2\ \mathcal{F}_1(T,V)\ \mathcal{F}_1(V,T)}{\mathcal{F}_1(T,V)+\mathcal{F}_1(V,T)}$.

5.2. Comparaison à d'autres méthodes de détection de communautés recouvrantes

Nous comparons ici notre méthode à trois méthodes de l'état de l'art pour la détection de communautés recouvrantes à savoir : (i) la percolation de k-cliques (Palla *et al.*, 2005; Kumpula *et al.*, 2008; Reid *et al.*, 2012), (ii) BigClam (Yang, Leskovec, 2013) et (iii) la partition de liens (Ahn *et al.*, 2010). Nous effectuons cette comparaison sur quatre grands réseaux réels pour lesquels on dispose d'une structure en communautés vérité de terrain. Ces réseaux sont détaillés en annexe : *Wikipédia 2008*, *Wikipédia 2012*, *Google scholar* et *DBLP*. Le tableau 3 montre ces résultats. Pour la percolation de k-cliques nous avons choisi le k donnant les meilleurs résultats en termes de $\mathcal{S}(T,V)$. Pour notre méthode, nous avons choisi $k = 10$ (ceci permet de ne pas calculer les proximités pour les k nœuds les plus proches d'un nœud pour lequel les proximités ont déjà été calculées) et $\delta = 0.8$ (pour ne pas prendre en compte une communauté similaire à plus de 80% à une communauté détectée selon le score F_1). Nous avons également sélectionné les communautés trouvées au moins deux fois et donc supprimé celles trouvées une seule fois.

Tableau 3. Tableau montrant les scores $\mathcal{F}_1(T,V)$, $\mathcal{F}_1(V,T)$ et $\mathcal{S}(T,V)$ ainsi que le nombre de communautés trouvées pour les différentes méthodes.

Réseau et nombre de communautés	k-clique percolation			BigClam			partition de liens			approche proximité		
scholar	0.14	0.16	0.15	0.07	0.00	0.01	0.16	0.12	0.14	0.13	0.13	0.13
16773	26318 (k=3)			100			11133			18845		
DBLP	0.00	0.03	0.00	0.05	0.10	0.07	0.00	0.03	0.00	0.32	0.19	0.24
24	3113 (k=5)			50			12871			15		
wiki08										0.67	0.03	0.04
79365										355		
wiki12										0.72	0.03	0.05
730752										2046		

Pour le réseau *Google scholar*, notre méthode donne des résultats similaires aux autres. On peut l'expliquer car d'une part, les communautés "vérité de terrain" sont de piètre qualité du fait de leur définition (annotation manuelle par les chercheurs de leur domaine de recherche), d'autre part le réseau lui-même souffre du même problème. C'est pourquoi une méthode médiocre et une méthode parfaite pourraient avoir des résultats similaires. Nous voyons que pour le réseau *DBLP* notre méthode est bien au-dessus des autres pour les 3 mesures. Soulignons que, de par leur définition, ce réseau et ses communautés sont de meilleure qualité que dans le réseau précédent.

Pour les réseaux *Wikipédia 2008* et *Wikipédia 2012*, notre méthode donne de très bons résultats en termes de $\mathcal{F}_1(T,V)$, mais pas vraiment en termes de $\mathcal{F}_1(V,T)$, c'est-à-dire que les communautés que l'on trouve sont très similaires à des communautés "vérité de terrain", mais l'inverse n'est pas vrai. En fait, notre méthode retourne un faible nombre de communautés par rapport au nombre réel de communautés, ainsi beaucoup de communautés "vérité de terrain" ne trouvent pas leur équivalent dans les communautés trouvées et donc le score $\mathcal{F}_1(V,T)$ est faible. Ceci peut s'expliquer car nous avons considéré toute catégorie (agrégée

avec ses sous-catégories) comme étant une communauté, ce qui n'est pas le cas en pratique : beaucoup de catégories agrégées ne sont pas des communautés.

Les trois autres méthodes n'ont pas été capables de traiter ces réseaux Wikipédia : 64 Go de mémoire n'ont pas été suffisants pour la percolation de k-clique et la partition de liens. Quant à BigClam, il n'a pas terminé après deux semaines de calcul avec 4 processeurs. Ce problème de BigClam a d'ailleurs déjà été constaté dans d'autres articles (Whang *et al.*, 2013).

6. Conclusion et perspectives

Nous avons montré que contrairement à ce que l'on pourrait penser, le nombre de communautés pertinentes dans un réseau est faible (de l'ordre du nombre de nœuds, voire plus petit) et la plupart de ces communautés semblent avoir un représentant, c'est-à-dire un nœud appartenant seulement à cette communauté et important en son sein. En prenant en compte ces observations, nous avons mis au point un algorithme permettant de détecter ces groupes pertinents à l'aide de la détection de la communauté ego-centrée de ces représentants. Notre approche exploite une mesure de proximité et un nettoyage intelligent de ces communautés ego-centréees. L'algorithme est plus rapide que ceux de l'état de l'art et donne de meilleurs résultats sur certains jeux de données.

Notre algorithme de nettoyage des communautés trouvées consistant à calculer la similarité des communautés partageant au moins un nœud est suffisamment rapide pour traiter en quelques jours des réseaux contenant des centaines de millions de liens comme le réseau *Wikipédia 2012* (ce que les autres algorithmes de l'état de l'art auxquels nous avons comparé notre méthode ne peuvent pas faire). Cependant notre algorithme de nettoyage (goulot d'étranglement en termes de vitesse d'exécution de notre algorithme) pourrait sûrement être amélioré en utilisant un algorithme de *Locality sensitive hashing* (Leskovec *et al.*, 2014) dont l'idée principale est d'utiliser une famille de fonctions de hachage choisies telles que des points proches dans l'espace d'origine aient une forte probabilité d'avoir la même valeur de hachage. Ainsi des ensembles de communautés similaires pourraient être détectés plus rapidement (car elles auraient la même valeur de hachage) et le nettoyage gagnerait en vitesse par rapport à notre heuristique courante (consistant à comparer toutes les communautés partageant au moins un nœud).

Bien qu'il semble que beaucoup de communautés aient un représentant (comme par exemple les grandes catégories agrégées de Wikipédia comme "Boxing" ou "Chess" qui en ont au moins un), ce fait est sûrement faux pour certains réseaux ou certains types de communautés. Par exemple il semble que dans un réseau social chaque nœud appartienne à plusieurs communautés (groupe d'amis, famille, collègues). Pour ce type de réseaux une adaptation de la méthode est possible, par exemple en considérant des paires de nœuds comme graine et en cherchant des communautés bi-égocentrées (communauté définie par deux nœuds) en s'inspirant des travaux effectués dans (Danisch *et al.*, 2012). Cependant un accroissement du temps de calcul serait observé et une sélection intelligente des paires de nœuds devrait être effectuée. Nous laissons cette généralisation pour des travaux futurs.

Remerciements

Ce travail a été partiellement soutenu par le projet CODDDE ANR-13-CORD-0017-01. Les auteurs remercient Noé Gaumon et Damien Noguez qui ont apporté de précieuses suggestions pour l'algorithme de nettoyage.

Références

Ahn Y.-Y., Bagrow J. P., Lehmann S. (2010). Link communities reveal multiscale complexity in networks. *Nature*, vol. 466, n° 7307, p. 761–764.

Andersen R., Chung F., Lang K. (2006). Local graph partitioning using pagerank vectors. In *Foundations of computer science, 2006. focs'06. 47th annual ieee symposium on*, p. 475–486.

Cohen S., Kimelfeld B., Koutrika G. (2012). A survey on proximity measures for social networks. In *Search computing*, p. 191–206. Springer.

Danisch M., Guillaume J.-L., Le Grand B. *et al*. (n.d.). Learning a proximity measure to complete a community. In *The 2014 international conference on data science and advanced analytics (dsaa2014)*.

Danisch M., Guillaume J.-L., Le Grand B. *et al*. (2012). Towards multi-ego-centered communities: a node similarity approach. *Int. J. of Web Based Communities*.

Eppstein D., Löffler M., Strash D. (2010). *Listing all maximal cliques in sparse graphs in near-optimal time*. Springer.

Kanawati R. (2014). Seed-centric approaches for community detection in complex networks. In G. Meiselwitz (Ed.), *Social computing and social media*, vol. 8531, p. 197-208. Springer.

Kumpula J. M., Kivelä M., Kaski K., Saramäki J. (2008). Sequential algorithm for fast clique percolation. *Physical Review E*, vol. 78, n° 2, p. 026109.

Leskovec J., Rajaraman A., Ullman J. D. (2014). *Mining of massive datasets*. Cambridge University Press.

Malliaros F. D., Vazirgiannis M. (2013). Clustering and community detection in directed networks: A survey. *Physics Reports*, vol. 533, n° 4, p. 95–142.

Moon J. W., Moser L. (1965). On cliques in graphs. *Israel journal of Mathematics*, vol. 3, n° 1, p. 23–28.

Palla G., Derényi I., Farkas I., Vicsek T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, vol. 435, n° 7043, p. 814–818.

Reid F., McDaid A., Hurley N. (2012). Percolation computation in complex networks. In *Proceedings of the 2012 international conference on advances in social networks analysis and mining (asonam 2012)*, p. 274–281.

Whang J. J., Gleich D. F., Dhillon I. S. (2013). Overlapping community detection using seed set expansion. In *Proceedings of the 22nd acm international conference on conference on information & knowledge management*, p. 2099–2108.

Wilson K. A., Green N. D., Agrawal L., Gao X., Madhusoodanan D., Riley B. *et al*. (2013). Graph-based proximity measures. *Practical Graph Mining with R*, p. 135.

Xie J., Kelley S., Szymanski B. K. (2011). Overlapping community detection in networks: the state of the art and comparative study. *arXiv preprint arXiv:1110.5813*.

Yang J., Leskovec J. (2013). Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth acm international conference on web search and data mining*, p. 587–596.

Yang J., Leskovec J. (2015). Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, vol. 42, n° 1, p. 181–213.