Un nouvel algorithme de propagation de labels avec barrages

Jean-Philippe Attal, Maria Malek

EISTI, Ecole Internationale des Sciences du Traitement de l'Information
Laboratoire Quartz, Cergy-France 95000, France
jal@eisti.eu,mma@eisti.eu

RÉSUMÉ. La propagation de labels est l'une des méthodes les plus rapides pour la détection de communautés, de complexité quasi-linéaire en termes d'arêtes. Il s'agit d'une méthode locale où chaque nœud possède son propre label qui change par interaction avec son voisinage. Malheureusement, cette méthode présente deux inconvénients majeurs. Le premier est qu'une mauvaise propagation peut mener à de trop grandes communautés (le problème des communautés géantes). Le second inconvénient est l'instabilité de la méthode, ne donnant que très rarement le même résultat après chaque lancement. Dans cet article, nous proposons des algorithmes et une étude portant sur la propagation de labels en plaçant des barrages sur certaines arêtes dans le but d'éviter de mauvaises propagations. Puis nous appliquons une méthode de détection de cœurs basée sur l'alimentation d'une matrice de fréquence de co-apparition dans le but de stabiliser la propagation de labels.

ABSTRACT. Label propagation is one of the fastest methods for community detection, with a near linear time complexity. It's a local method, where each node has its own label which changes by interaction with its neighbourhood. Unfortunately, this method has two major drawbacks. The first is a bad propagation which can lead to huge communities without sense (giant communities problem). The second is the instability of the method. Each trial of a label propagation algorithm gives rarely the same result. In this paper, we propose algorithms and a study on the label propagation by putting artificial dams on edges in order to avoid bad propagations. We then apply an ensemble learning clustering method based on a frequency matrix in order to stabilize the algorithm.

MOTS-CLÉS : détection de communautés, propagation de labels, barrages, ensemble learning, détection de cœurs.

KEYWORDS: community detection, label propagation, dams, ensemble learning, core detection.

DOI:10.3166/RIA.25.393-418

1. Introduction

La plupart des réseaux représentant des systèmes réels montrent des caractéristiques propres comme des groupes de nœuds fortement liés entre eux (que l'on appelle des communautés) et peu avec le reste du graphe. Un réseau de collaboration de chercheurs en est un exemple, où les groupes de nœuds sont des personnes travaillant sur des thèmes similaires. Une étude comparative a été effectuée par Fortunato (2010).

Dans cet article, nous exposons un nouvel algorithme de détection de communautés (et ses variantes), basé sur l'information globale du graphe dans le but d'aider une méthode locale à détecter les communautés. Nous verrons qu'en limitant la propagation de labels par des barrages, nous pourrons éviter l'obtention de trop grandes communautés. Une méthode basée sur la fréquence d'apparition des nœuds dans une même communauté permettra de détecter des cœurs (ensemble de nœuds fortement liés) et des communautés. Nous expliquons dans la section 2 la propagation de labels et la détection de cœurs, dans la section 3 notre approche liant barrages et stabilisation, puis nos expérimentations dans la section 4, et nous concluons dans la section 5 avec nos futures perspectives

2. L'approche par propagation de labels

2.1. La propagation de labels standard

La méthode de propagation de labels (Raghavan, 2007), est basée sur la transmission d'un label d'un nœud à ses voisins. Un état d'équilibre est atteint lorsque chaque nœud a son label égal à celui de la majorité de ses voisins. Soit un graphe $G = (V, E)$ avec V l'ensemble des sommets et E l'ensemble des arêtes $|E| = m$. À chaque étape, chaque nœud met à jour son label selon les labels de ses voisins, en utilisant un vote. Le label du nœud x prendra le label majoritaire de ses voisins. En notant c_x le label du nœud x, et par $N^l(x)$ l'ensemble du voisinage du nœud x avec le label l, l'affectation d'un label au nœud x est donnée par la formule suivante :

$$c_x = \text{argmax}_l \, |N^l(x)|$$

À la fin du processus, les nœuds ayant le même label représentent une communauté.

Cette méthode peut être effectuée de manière *synchrone* ou *asynchrone*. La méthode asynchrone signifie que la mise à jour d'un label d'un nœud est connue par tous les autres nœuds du graphe immédiatement. Son label est utilisé pour la mise à jour des labels des autres nœuds. Ce qui n'est pas le cas du mode asynchrone, où la mise à jour des labels utilise les labels des nœuds à la précédente propagation. Cet algorithme étant très instable, il serait souhaitable d'avoir une méthode de stabilisation. Pour ce faire, une méthode consiste à lancer plusieurs fois l'algorithme

non déterministe et à considérer les nœuds qui apparaissent le plus souvent ensemble dans une même communauté. On appelle ces nœuds dont la fréquence d'apparition est très forte, des *cœurs*.

Nous proposons d'utiliser la méthode de détection de cœurs (Seifi *et al.,* 2013). Elle consiste à utiliser une matrice de fréquence, spécifiant le nombre de fois que chaque paire de nœuds apparaît dans les mêmes communautés. Nous nommerons cette méthode le *stabilisateur*. Les auteurs de cette méthode l'ont appliquée en utilisant la méthode de Blondel et al. (2008). Il s'agit d'une méthode agglomérative, dont la contraction est effectuée par optimisation locale d'une fonction de qualité, la modularité (Newman et Girvan, 2004). Les auteurs ont montré qu'ils avaient réussi à stabiliser la méthode de Louvain, et à trouver des cœurs stables.

Soit $\mathcal{N}$ le nombre de fois que l'algorithme non déterministe est lancé. A chaque essai, nous notons chaque paire de nœuds qui apparait dans une même communauté. Il est ainsi possible de définir une matrice carré de taille n,$\mathrm{p}_{ij}^{\mathcal{N}}$, telle que l' élément p_{ij}représente la fréquence d'appartenance des nœuds i et j à une même communauté après les $\mathcal{N}$ essais. p_{ij} , étant une probabilité, pout tout $(i,j) \in V \times V$ a une valeur comprise entre 0 et 1. Un nombre proche de 1 signifie que les nœuds i et j sont souvent ensemble durant les $\mathcal{N}$ essais. Pour trouver les cœurs, on crée un nouveau graphe $G' = (V, E')$ où E' représente l'ensemble des arêtes créées à partir de la matrice de fréquence en utilisant un seuil $\alpha \in [0,1]$ permettant de faire apparaître des composantes connexes. Les composantes connexes sont ici les cœurs, qui correspondent à nos communautés. G' est appelé le graphe α-seuillé. α est un paramètre qui influence le fait de créer des connections dans le nouveau graphe G', et par conséquence, le nombre de composantes connexes. D'après les études menées par (Seifi *et al.,* 2013), de faibles valeurs de α conduisent à peu de composantes connexes alors que de fortes valeurs de α mènent à beaucoup de composantes connexes.

2.2. Autres approches à base de propagations de labels

Des méthodes basées sur la propagation de labels ont été proposées durant ces dernières années. Parmi celles-ci, on peut citer Leung *et al.* (2009), qui utilise un coefficient d'atténuation pour limiter la propagation de labels par rapport à la distance parcourue depuis leurs origines. Šubelj et Bajec (2011) ont continué dans cette voie, et ont proposé les méthodes offensives et défensives basées sur le fait que la propagation de labels ne peut pas dépasser une certaine distance, ou au contraire est paramétrée pour aller le plus loin possible. Liu *et al.* (2010), ainsi que Barber et Clark (2009) ont proposé une propagation de labels basée sur l'optimisation de la modularité.

Certains algorithmes de propagation de labels sont également basés sur un processus de consensus, de clustering d'ensembles et de stabilisation afin de rendre l'algorithme moins instable et d'éviter de mauvaises propagations. Zong-Wen *et al.* (2014) proposent une propagation de labels guidée par consensus. Les auteurs lancent plusieurs fois l'algorithme de propagation de labels pour obtenir différentes

partitions. Un nouveau graphe pondéré est alors créé où la pondération des arêtes représente le nombre de fois que chaque paire de nœuds est dans une même communauté. C'est alors qu'une propagation de labels avec consensus basée à la fois sur les poids des arêtes et la fréquence des labels permet d'assigner à un nœud un nouveau label. Le consensus s'effectue en faisant intervenir un paramètre $\lambda \in [0,1]$.

Les résultats ont montré que cette méthode produisait de meilleurs résultats que le LPA mais qu'une étude préalable pour chaque graphe devait être faite pour paramétrer. Cette méthode ne permet pas l'élaboration de dendrogramme.

Staudt et de Meyerhenke (2014) proposent une propagation de labels avec contraction de manière parallèle et distribuée sous openMP. Leur approche consiste à lancer plusieurs fois la propagation de labels et à effectuer une contraction du graphe sur les nœuds qui apparaissent systématiquement au sein d'une même communauté. Une contraction sur le graphe est effectuée et une nouvelle propagation de labels est lancée. Il ressort des expérimentations que leur méthode donne de meilleurs résultats que le LPA et est assez rapide. Cependant, la phase de contraction est fonction du nombre de propagations de labels en parallèle mais aussi sujette à quelques absences de contractions, notamment s'il y a certaines mauvaises propagations. Une étude sur le clustering d'ensemble pour les graphes a été effectuée par Ovelgönne et Geyer-Schulz (2012) basée sur les travaux de Topchy (2005).

Les méthodes présentées ci-dessus se fondent soit sur une modification de la propagation de labels, soit sur des algorithmes de consensus à paramétrer suivant certains objectifs. Les premières méthodes permettent de réduire le nombre de mauvaises propagations mais sont fragiles sur des graphes de terrains avec de grandes et de petites structures communautaires. Les méthodes à base de consensus demandent l'intervention de clauses conditionnelles ou d'une paramétrisation qui nécessite des études préalables et qui n'empêche pas l'obtention de la stabilisation. De plus, à l'exception de la méthode de Staudt et de Meyerhenke, aucune de ces méthodes ne permet de créer un dendrogramme. Nous proposons de ne pas modifier la propagation de labels mais d'associer plutôt cette méthode à la détection de cœurs, tout en modifiant la topologie du graphe pour mieux en révéler les structures communautaires, qui peuvent être de tailles différentes.

3. Approche proposée

La propagation de labels avec barrages a pour objectif de détecter des communautés, utilisant à la fois l'information globale topologique du graphe et une méthode locale pour trouver les communautés. La propagation de labels souffrant d'instabilité et de la production de trop grandes communautés, nous proposons une méthode pour résoudre ces deux problèmes en interdisant à certaines arêtes de propager un label tout en effectuant une méthode de détection de cœurs en lançant plusieurs fois l'algorithme non déterministe. Les mesures issues de l'analyse des réseaux sociaux peuvent se révéler très utiles pour connaître l'importance de

certaines arêtes ou nœuds au sein d'un graphe. Nous utilisons ici la mesure d'intermédiarité Girvan et Newman (2002) qui nous permettra de placer des barrages lors de l'exécution de la propagation de labels.

Soit $w \to \mathbb{R}$ la fonction de pondération sur les arêtes de G, avec E, ensemble d'arêtes du graphe G. Pour un graphe non pondéré, nous avons $w(e) = 1 \forall\, e \in E$ Soit un chemin entre deux sommet commençant en $s \in V$ et se terminant à $t \in V$. Notons σ_{st} le nombre de plus courts chemins entre les sommets s et t. La notion d'intermédiarité d'une arête repose sur le nombre de plus courts chemins qui passent à travers une certaine arête. La centralité d'intermédiarité pour les arêtes, que l'on note $CI(e)$ pour un lien e est donnée par $CI(e) = \sum_{s,t \in V, s \neq t} \frac{\sigma_{st}(e)}{\sigma_{st}}$ où $\sigma_{st}(e)$ représentant le nombre de plus courts chemins allant de s à t passant par e. La complexité pour calculer la centralité d'intermédiarité est en $O(n^4)$. Cependant, des recherches, Brandes (2001), Bader *et al.* (2007) et Geisberger *et al.* (2008) portant sur l'approximation de cette mesure ont pu réduire la complexité en $O(n^2)$ sur des graphes complexes.

Nous utilisons la centralité d'intermédiarité pour mettre nos barrages et éviter l'obtention de trop grandes communautés. La complexité de la centralité d'intermédiarité étant en $O(n^4)$ (Fortunato, 2010), la complexité de PLAB est en $O(n^4) + m - \beta \times m$, avec β le nombre de barrages. Nous exposons l'algorithme de propagation de labels avec barrages (PLAB, *Algorithme 1)* et un exemple d'application à la *figure 1*.

Algorithme 1. PLAB

1: ***Paramètre :*** *Un graphe G=(V,E), un réel β*
2: ***Sortie*** : *Une partition* $P = \{P_1, \dots, P_C\}$
3: Calcul de la centralité d'intermédiarité de G.
4: *Séle*ctionner β pourcentage des arêtes ayant les plus grandes valeurs d'intermédiarité et mettre des barrages.
5: Lancer la propagation de labels standard (communautés de G).
6 : **Retourner** La partition $P = \{P_1, \dots, P_C\}$

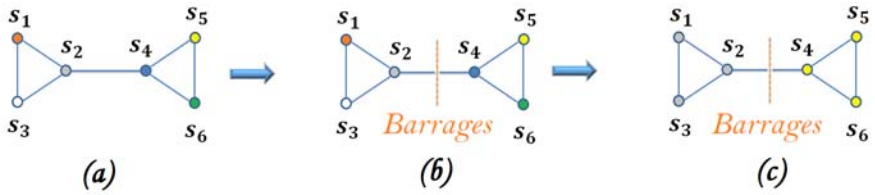

Figure 1. Exemples de propagation de labels avec barrages

Dans l'exemple de la figure 1, le graphe comprends 6 sommets. On calcule dans un premier temps le degré d'intermédiarité des arêtes. Un tri est effectué sur les arêtes. Une sélection permet de prendre l'arête ayant la plus grande valeur d'intermédiarité. On place alors un barrage artificiel qui sera utilisé lors de la propagation de labels. Ce barrage interdit à la propagation de s'effectuer. On trouve par la suite deux communautés.

Cependant, cet algorithme présente deux inconvénients, le choix du nombre de barrages, et l'instabilité réduite, mais toujours présente. Pour remédier à ces problèmes nous proposons deux algorithmes basés sur la détection de cœurs. Le premier algorithme (algorithme 2) est basé sur l'optimisation d'une fonction de qualité qui peut par exemple être la modularité ou la conductance. Nous effectuons plusieurs fois PLAB avec un certain β (pourcentage de barrages) pour alimenter une matrice de fréquence de coappartenance. Nous recommençons le procédé en changeant la valeur de β, β variant de 0 à 1 par un pas choisi par l'utilisateur Δ avec une nouvelle matrice de fréquence de coappartenance. Pour chaque matrice de fréquence, nous calculons les composantes connexes qui représentent les communautés et effectuons une évaluation avec une fonction de qualité. Le résultat ayant le score optimal permet de retourner une partition. La complexité de l'algorithme est en $O(n^2) + \frac{1}{\Delta} \times -\beta \times \mathcal{N} \times k \times (m - \beta \times m)$ où k est le nombre d'itérations de la propagation de labels. La multiple propagation de labels avec barrages et stabilisation avec optimisation d'une fonction de qualité (MPLBS) est exposée : algorithme 2.

Algorithme 2. MPLBS

1: ***Paramètre :*** *Un graphe G=(V,E), un réel β, un seuil α, Δ le pas, $\mathcal{N}$ le nombre d'essais*

2: ***Sortie*** : *Une partition $P = \{P_1, \dots, P_C\}$*
3: Calcul de la centralité d'intermédiarité de G.
4 : Liste_Partition = []
5 : **Pour** k = 0 à 1 avec un pas de Δ **Faire**
6: Mettre des barrages sur les $k \times |E|$ arêtes ayant les plus grandes valeurs d'intermédiarité
7 : Allouer une matrice vide $\mathrm{p}_{ij}^{\mathcal{N}}(k)$
8 : Alimenter la matrice $\mathrm{p}_{ij}^{\mathcal{N}}(k)$ en lançant $\mathcal{N}$ fois la propagation de labels sur le graphe avec barrages.
9 : Créer un nouveau graphe $G' = (V, E')$ en partant de $\mathrm{p}_{ij}^{\mathcal{N}}(k)$ avec des arêtes dont la pondération est égale ou supérieure à α
10 : Créer une partition P_k en considérant les C composantes connexes.
11 : Liste_Partition. Ajouter (P_k)
12 : $k = k + \Delta$
13 : **Fin Pour**
14 : **Retourner** la partition P_k de Liste_Partition avec le meilleur score de la fonction de qualité choisie par l'utilisateur: $p_k^* = \{p_1^k{}_1, \dots, p_C^*\}$

Le second algorithme fait varier le nombre de barrages en alimentant une seule matrice de coappartenance, puis détecte les composantes connexes. L'idée est qu'alimenter une matrice de fréquence de coappartenance par une même méthode qui ne produit pas toujours de bons résultats ne peut pas être satisfaisant. Cependant, alimenter une matrice, avec différentes méthodes, ayant différents niveaux de barrage pourrait améliorer la qualité des composantes connexes trouvées pour la détection de communauté. L'algorithme prend en paramètre, outre le seuil α et le nombre d'essais $\mathcal{N}$, l'intervalle considéré pour l'alimentation de notre matrice, $\Delta_{[x,y]}, x \in [0,1], y \in [x,1]$ et $x < y$, avec un pas pour parcourir cet intervalle Δ. L'algorithme de propagation de labels avec barrages utilisant la stabilisation (PLBS) est exposé : algorithme 3.

Algorithme 3. PLBS

1: ***Paramètre :*** *Un graphe G=(V,E), un réel β, un seuil α,* Δ le pas, $\boldsymbol{\mathcal{N}}$ le nombre d'essais

2: ***Sortie*** : *Une partition* $P = \{P_1, \dots, P_C\}$
3: Calcul de la centralité d'intermédiarité de G.
4 : Allocation d'une matrice de fréquence de coappartenance vide
5 : **Pour** k = 0 à 1 avec un pas de $\boldsymbol{\Delta}$ **Faire**
6 : Mettre des barrages sur les $k \times |E|$ arêtes ayant les plus grandes valeurs d'intermédiarité.
4: Lancer $\boldsymbol{\mathcal{N}}$ fois la propagation de labels avec un nombre différent de barrages.
5 : Remplir la matrice de fréquence de coappartenance avec les résultats des différentes propagations de labels.
6 : $k = k + \Delta$
7 : **Fin Pour**
8 : Créer un nouveau graphe $G' = (V, E')$ en partant de $\mathrm{p}_{ij}^{\mathcal{N}}(k)$ avec des arêtes dont la pondération est égale ou supérieure à α
9 : Créer une partition P en considérant les C composantes connexes.
10 : **Retourner** La partition $P = \{P_1, \dots, P_C\}$

La complexité de l'algorithme est en $(n^2) + \frac{\mathbf{1}}{\Delta} \times -\beta \times \mathcal{N} \times k \times (m - \beta \times m) \times (y - x)$, où k est le nombre d'itérations de la propagation de labels.

Pour effectuer une comparaison de ces dernières méthodes basées sur les barrages, nous proposons la détection de communautés par propagation de labels avec détection de cœurs (CDLP). L'intérêt est surtout expérimental pour effectuer une étude comparative avec et sans barrages. CDLP, l'algorithme 4, a une complexité algorithmique en $O((n + \mathcal{N}) \times k)$ où k est le nombre d'itérations de la propagation de labels.

Algorithme 4. CDLP

1: ***Paramètre :*** *Un graphe G=(V,E), un seuil* α, $\mathcal{N}$ le nombre d'essais

2: ***Sortie*** : Une partition $P = \{P_1, \dots, P_C\}$
3: Allocation d'une matrice de fréquence de coappartenance vide
4: Lancer $\mathcal{N}$ fois la propagation de labels avec un nombre différent de barrages.
5: Remplir la matrice de fréquence de coappartenance avec les résultats des différentes propagations de labels.
6 : Créer un nouveau graphe $G' = (V, E')$ en partant de $\mathrm{p}_{ij}^{\mathcal{N}}(k)$ avec des arêtes dont la pondération est égale ou supérieure à α
7 : Créer une partition P en considérant les C composantes connexes.
8 : **Retourner** La partition $P = \{P_1, \dots, P_C\}$

4. Expérimentations et analyse

Pour établir des comparaisons à partir de nos expériences, nous utilisons des mesures supervisées (lorsque nous connaissons les vraies communautés) telles que l'information mutuelle normalisée (NMI) Fred et Jain (2003), la pureté, l'index de Rand ajusté Rand (1971) (ARI) mais également non supervisées telles que la modularité Q (Newman et Girvan, 2004), et la conductance φ (Kannan *et al.,* 2004), (Shi et Malik, 2000). Nous expliquons les mesures supervisées et non supervisées que nous utilisons pour étudier la qualité de partitionnement de nos algorithmes.

4.1. Mesures supervisées et non supervisées

Les mesures supervisées permettent de comparer directement les résultats de l'algorithme proposé à la vraie partition. Dans certains, cas, elles peuvent également être utilisées pour observer la similarité des résultats entre plusieurs algorithmes.

L'indice de Rand ajusté (ARI) : Il s'agit d'une mesure permettant de comparer deux partitions P_1 et P_2, composées toutes les deux de parties disjointes. Cette mesure est basée sur une matrice de contingence pour savoir si des nœuds sont classés dans les mêmes communautés. Sa valeur est comprise entre 0 et 1. Elle prend la valeur 1 lorsque les deux partitions sont identiques. Si la valeur est proche de 0, les deux partitions sont très différentes.

L'information mutuelle normalisée (NMI) : C'est une mesure qui permet de représenter le degré de dépendance entre deux partitions P_1 et P_2. Elle est basée sur la théorie de l'information. Sa valeur peut varier entre 0 et 1. Plus la valeur sera proche de 1, plus il existera un lien de corrélation entre les deux partitions P_1 et P_2, et plus leur similarité sera grande.

Les mesures non supervisées permettent d'avoir une estimation de la qualité de partitionnement sans connaître les véritables partitions. Ces mesures sont soit basées sur des structures aléatoires respectant la topologie du graphe initial, soit basées sur

les densités et les liens entre communautés. Nous utilisons la modularité et la conductance.

La modularité : cette mesure est basée sur la génération d'un graphe aléatoire, nommé « le modèle nul », respectant la distribution des degrés des nœuds du graphe sur lequel est appliqué l'algorithme de détection de communautés. En projetant les communautés sur les deux graphes, par comparaison, si des structures avec de fortes densités apparaissent sur le graphe aléatoire, cela sous-entend que des structures communautaires sembleraient avoir été trouvées.

Considérons une partition $P = \{P_1, \dots, P_C\}$ en k parties représentant des groupes de nœuds distincts du graphe $G = (V, E)$. En rappelant la notation $|E| = m$, d_c le degré d'un ensemble de nœuds de la partie c de P représentant une communauté, l_c le nombre d'arêtes à l'intérieur de c, on donne la modularité comme étant :

$$Q_P = \sum_{c \in k} \frac{l_c}{m} - \frac{d_m}{2m}^2$$

La modularité peut avoir une valeur entre -1 et 1. Plus la valeur d'une partition est proche de 1, plus les structures détectées sont proches de structures communautaires. Cette mesure peut être appliquée comme fonction objective de certains algorithmes. Cependant, dans notre cas, elle nous aidera à avoir de l'information sur le partitionnement, surtout dans les cas où les véritables communautés de nos graphes de terrains ne sont pas connues.

La conductance : cette mesure est basée sur la densité des communautés et du nombre de liens sortant de ces dernières. Une structure communautaire est supposée avoir beaucoup de liens en son sein et un nombre faible de liens sortants. La conductance est basée sur le rapport entre le nombre de liens sortants l_{out}^c et le nombre total de liens (intérieur l_{int}^c et extérieur) pour une communauté c. En considérant une partition $P = \{P_1, \dots, P_C\}$ en k parties de noeuds disjoints, la conductance se définit comme suit:

$$\varphi = \frac{1}{n} \sum_{c=1}^{k} \frac{l_{out}^c}{2 \times l_{out}^c + l_{out}^c)}$$

La conductance peut avoir une valeur comprise entre 0 et 1. Plus cette valeur sera proche de 0, plus cela signifiera que les communautés ont une densité forte avec peu de liens pointant vers l'extérieur.

La pureté : cette mesure, pour une communauté $c_i \in P_1 = \{c_1, \dots, c_l\}$, par rapport à une autre partition P_2 (avec $P_2 = \{r_1, \dots, r_n\}$ se définit comme suit :

$$purete(c_i, P_2) = \max_{i:\to l} \frac{||c_i \cap r_i||}{||c_i||}$$

Cette fonction calcule le taux de recouvrement maximal entre la communauté c_i et les communautés se trouvant dans la partition P_2. On définit la pureté de la partition P_1 par rapport à la partition P_2 par la somme pondérée de la pureté des communautés de P_1par rapport à P_2, ce qui nous donne :

$$purete(P_1, P_2) = \sum_{i=1}^{m} w_{c_i} \times purete(c_i, P_2)$$

où $w_{c_i} = \frac{||c_i||}{\sum_{i=1}^{m} ||c_i||}$

4.2. Expérimentation sur des réseaux réels

Nous avons expérimenté nos algorithmes sur des réseaux connus de la littérature tableau 1. Dans la suite de cette section, nous utilisons la notation d pour signifier la densité du graphe, AT pour la transitivité moyenne du graphe et # pour le nombre de communautés. Les graphes sociaux sur lesquels nous travaillons sont :

– le club de karaté de Zachary (1977) (Zac) : un réseau de membres d'un club chez lesquels une dispute créa la formation de deux communautés autour du manager et de l'entraîneur.

– Un réseau footballistique (Girvan et Newman, 2002) (Foot) : un graphe exposant 11 différentes compétitions entre clubs de football américains.

– Un réseau de livres politiques américains (Krebs, 2004) (pol) : un graphe de co-achat exposant les livres politiques édités en 2004 et vendus sur Amazon.com. le graphe comprend 3 communautés, les démocrates, les républicains et les neutres.

– Un réseau de dauphins (Lusseau et al., 2003) (Dol) : un graphe de dauphins étudié à Doubtful Sound, en Nouvelle Zélande. Le graphe comprend 62 dauphins représentés en deux communautés, mâles et femelles.

– Un réseau de musiciens de jazz (Gleiser et Danon, 2003) (jazz). Chaque nœud représente un musicien et une arête dénote que deux musiciens ont joué ensemble dans un groupe. Les données furent collectées en 2003.

– Le réseau aéroportuaire américain de 1997 où les nœuds sont des aéroports et les liens sont des lignes directes (Batagelj et Mrvar, 2006).

– Un réseau d'auteurs scientifiques, (Newman, 2006), où les arêtes représentent le nombre de papiers co-écrits dans le domaine des réseaux sociaux.

Pour nos expérimentations, nous prenons $\mathcal{N}$=100 pour stabiliser la propagation de labels et alimenter les matrices de co-fréquences. Nous effectuons une étude comparative avec des algorithmes issus de la littérature, comme la propagation de labels (LPA) (Raghavan, 2007), des variantes comme Šubelj L., Bajec, (2011). DPA, Leung *et al.* (2009) (Hop), le modèle de spin (Ronhovde et Nussinov, 2010), la méthode spectrale (Spectral) avec la modularité (Newman, 2006), Infomap (Rosvall et Bergstrom, 2007), LICOD (Kanawati, 2011), Girvan Newman (GN)

(Girvan et Newman, 2002), Walk (Pons et Latapy, 2006), Louvain (Blondel *et al.*, 2008) et enfin la méthode de Seifi *et al.* (2013).

Tableau 1. Caractéristiques des réseaux de test pour notre étude

Réseaux	$\|V\| et \|E\|$	Diamètre	AT	#
Zachary	34-78	5	0,256	2
Football	115-615	4	0,094	12
Dauphins	62-159	8	0,309	2
Politique	105-441	7	0,348	3
Jazz	198-5 484	6	0	Inconnu
USAir97	332-2 126	6	-0,208	Inconnu
Netscience	1 589-2 742	17	0,693	Inconnu

Les méthodes citées ci-dessus présentent l'avantage d'offrir un spectre assez large, présentant des méthodes globales et locales. Le modèle de Spin est une méthode agglomérative, où le graphe va se contracter à partir de certains nœuds en optimisant une fonction, l'hamiltonien, en fonction de son voisinage ou d'un certain horizon. La méthode de Louvain consiste à optimiser de manière locale la modularité et à effectuer des contractions suivant cette optimisation. La méthode spectrale de Newman consiste à calculer les vecteurs propres de la matrice de modularité, à projeter les vecteurs dans un nuage de points, à appliquer un algorithme de clustering et à retransposer les clusters sur le graphe. La méthode Licod consiste à trouver certains nœuds qualifiés de leaders au sein du graphe, et à associer les autres nœuds (nommés suiveurs) pour former des communautés.

Pour chaque graphe de test, nous proposons de donner le MPLBS avec un pas faible. Les résultats du CDLP peuvent être obtenus par observation des figures avec un nombre de barrages nul. Les résultats du PLBS seront donnés dans un tableau avec un pas $\Delta = 0{,}025$. La ligne des abscisses représente le pourcentage de barrages alors que l'ordonnée représente le score avec la matrice de co-appartenance relative au nombre de barrages. Pour le PLBS, nous choisirons 3 intervalles, $\Delta_{[0,0;0,33]}$, $\Delta_{[0,33;0,66]}$ et $\Delta_{[0,66;1,0]}$, en utilisant un pas $\Delta = 0{,}025$.

4.2.1. Étude expérimentale portant sur le PLAB

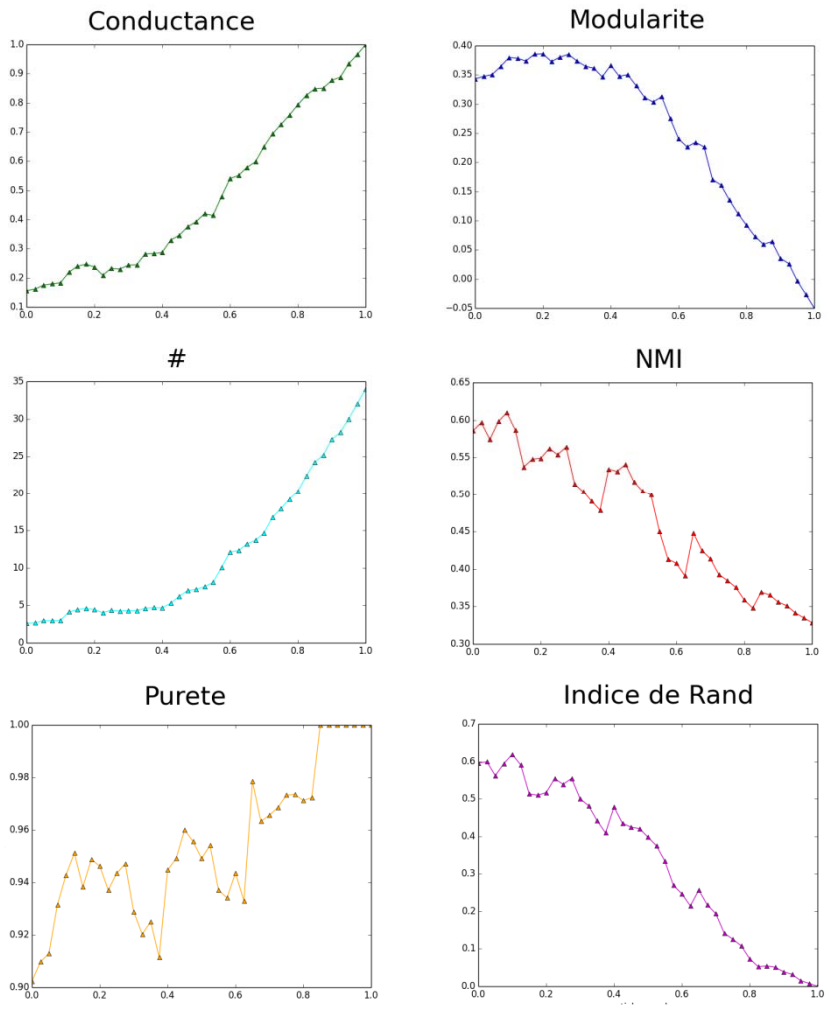

Figure 2. Mesures supervisées et non supervisées du PLAB sur le réseau Zachary. L'axe des abscisses représente le pourcentage de barrages

Notre première étude consiste à apprécier la mise en place de barrages afin d'observer une possible amélioration de la qualité par rapport à la propagation de labels standard. Nous proposons de faire varier le nombre de barrages sur certains réseaux classiques de la littérature dont nous connaissons les vraies structures communautaires, tout en calculant les mesures supervisées et non supervisées.

En utilisant PLAB sur le réseau de karaté, les valeurs des mesures supervisées et non supervisées s'améliorent pour atteindre un maximum autour de 12,5 % de barrages, ce que nous pouvons observer à la figure 2. La qualité se détériore par la suite avec une augmentation du nombre de communautés. Cependant, les résultats ne sont pas stables. On peut voir l'exemple sur le NMI, la présence de nombreux pics respectivement à 20 %, 40 % et 60 %. Cela est dû à l'instabilité de la détection de communautés qui ne produit que très rarement le même résultat d'un test à un autre. On obtient une décroissance non linéaire, avec des cassures.

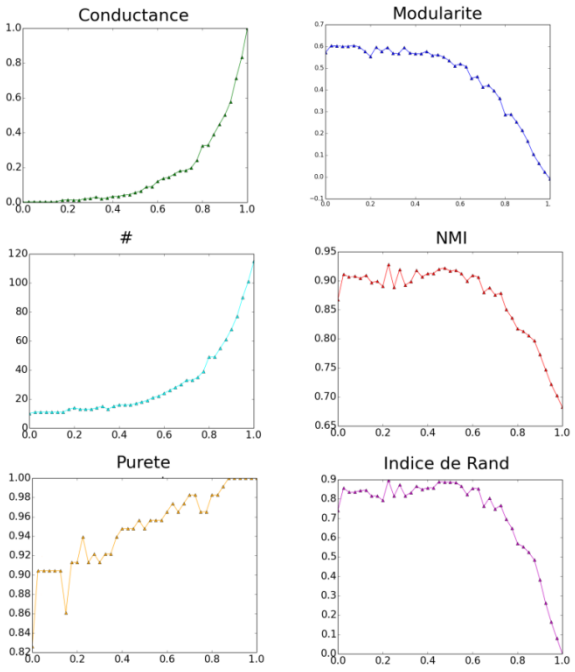

Figure 3. Mesures supervisées et non supervisées du PLAB sur le réseau footballistique. L'axe des abscisses représente le pourcentage de barrages

En considérant l'exemple des clubs de foot, où les communautés représentent des conférences, on voit que l'apport de barrages augmente ostensiblement la qualité de partitionnement. De 2,5 % à 55 % de barrages, les résultats sont meilleurs que ceux du LPA. Des pics récurrents apparaissent, signalant l'instabilité de la méthode.

4.3. Etude expérimentale portant sur le MLPBS

L'algorithme PLAB présente deux inconvénients : le choix du nombre de barrages, et l'instabilité réduite, mais toujours présente. Pour remédier à ces

problèmes nous proposons deux algorithmes basés sur la détection de cœurs. Le premier algorithme (algorithme 2), est basé sur l'optimisation d'une fonction de qualité qui peut par exemple être la modularité ou la conductance. Nous effectuons plusieurs fois PLAB avec un certain β (pourcentage de barrages) pour alimenter une matrice de fréquence de co-appartenance.

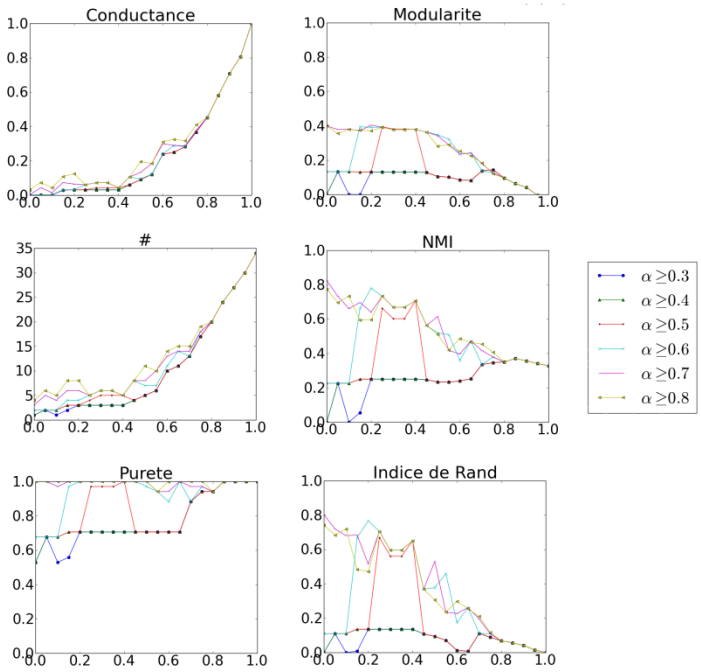

Figure 4. Résultat de la propagation de labels avec cœurs sur Zachary, avec différents pourcentages de barrages sur l'axe des abscisses

Pour Zachary, nous observons à la figure 4 que les résultats sont assez différents selon les valeurs du seuil α. Pour $\alpha \geq \{0{,}3; \ldots; 0{,}5\}$, la mise en place de barrages a une incidence sur la détection de communautés. De base, CDLP ne trouve qu'une grande communauté. C'est entre 20 % et 45 % de barrages que la méthode obtient de meilleurs résultats. Pour $\alpha \geq 0{,}6$, on obtient un pic du NMI de 0,78 à 20 % de barrages. Pour de très fortes valeurs de α, la présence des barrages ne permet pas d'améliorations notables de la qualité de partitionnement, ce qui est le cas pour $\alpha \geq \{0{,}7; 0{,}8\}$. Plus le nombre de barrages augmente, moins la propagation de labels peut s'effectuer. On voit que pour toutes les valeurs de α, une forte augmentation de la conductance a lieu après 40 % de barrages, avec une augmentation du nombre de communautés. Après 60 % de barrages, on voit la présence d'une convergence pour toutes les valeurs α, dûe au très grand nombre de barrages. Pour ce graphe, l'utilisation des barrages semble intéressante pour de

faibles valeurs de α. MLPBS retournera, par exemple avec $\alpha \geq 0{,}6$, le partitionnement à 20 % de barrages.

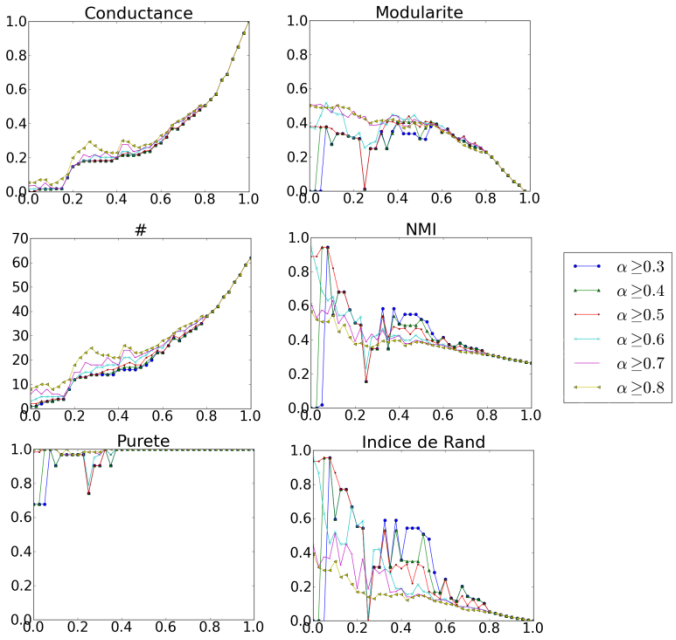

Figure 5. Résultat de la propagation de labels avec cœurs avec les dauphins, avec différents pourcentages de barrages sur l'axe des abscisses

Pour les dauphins, figure 5, les meilleurs résultats en termes de mesures supervisées sont obtenus pour $\alpha \geq \{0{,}3; 0{,}4\}$. La qualité de détériore après $\alpha \geq 0{,}6$. Pour $\alpha \geq 0{,}3$, jusqu'à 7,5% de barrages, aucune communauté n'est détectée à 10%, deux communautés sont détectées avec un NMI de 0,95. Pour de faibles valeurs de , la mise en place de barrages peut se révéler bénéfique et permet d'améliorer la qualité de la détection de communautés, ce qui n'est pas le cas pour des valeurs de α très élevées. Pour $\alpha \geq 0{,}8$, les conductances sont très élevées et l'utilisation de barrages devient inutile. MLPBS retournera, par exemple avec $\alpha \geq 0{,}6$, le partitionnement à 7,5 % de barrages.

Pour le club de football, figure 6, la qualité augmente jusqu'à 50 % de barrages. La matrice de co-fréquence stabilise correctement le LPA, sans présence de grands pics. Le nombre de communautés n'augmente que très faiblement jusqu'à 40 % et stagne jusqu'à 50%. Les arêtes où les barrages ont été mis sont sur les liens qui lient les communautés entres elles. Après 55 % de barrages, la qualité se détériore très rapidement. Selon nos observations, les barrages se mettent dans des zones fortement denses.

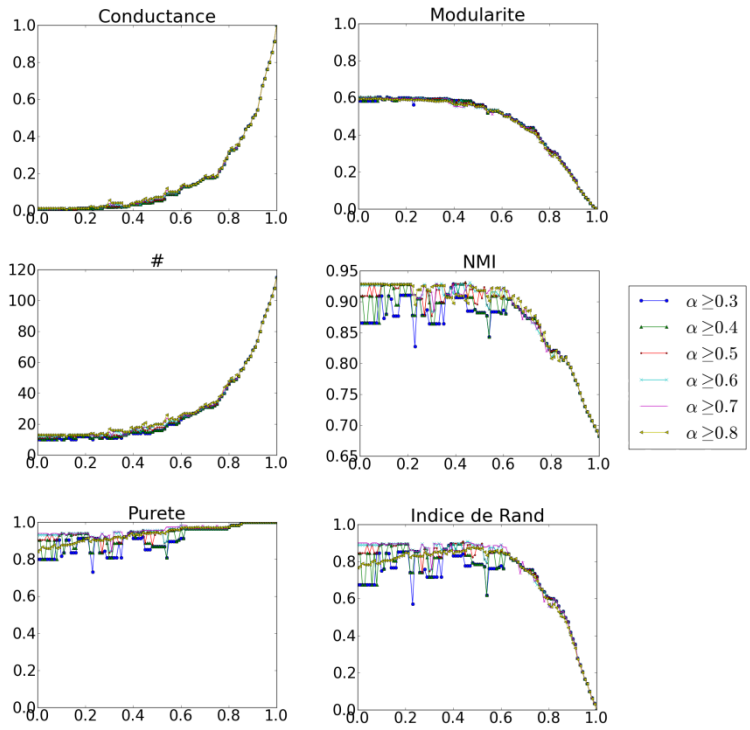

Figure 6. Résultat de la propagation de labels avec cœurs avec le réseau footballistique, avec différents pourcentages de barrages sur l'axe des abscisses

Pour les livres politiques dont les résultats sont exposés à la figure 7 jusqu'à 10 %, le nombre de communautés reste stable avec un bon NMI et un bon ARI. Puis ce nombre augmente de manière proportionnelle avec le nombre de barrages.

Le réseau de jazz est un graphe où les nœuds sont les musiciens et les liens le fait qu'ils aient joué ensemble ou pas. Il comporte 198 nœuds pour 5 484 arêtes. On observe dans ce cas que α joue un rôle important sur la stabilité et la qualité de l'algorithme, figure 8. Pour $\alpha \geq \{0{,}3; 0{,}4\}$, la modularité est faible avec un pic entre 15 % et 20 % de barrages puis à 65 %. Un faible α ne permet pas de trouver de bonnes structures communautaires. Les meilleurs résultats de la modularité sont atteints avec $\alpha \geq \{0{,}7; 0{,}8\}$, où il n'y a pas de présence de cassures. Les communautés étant de plus petites tailles, cela diminue le risque de mauvaises propagations. Les meilleurs résultats en termes de mesures supervisées sont atteints entre 15 % et 20 % de barrages. On voit d'ailleurs la conductance plus élevée avec $\alpha \geq \{0{,}7; 0{,}8\}$ que pour $\alpha \geq \{0{,}3; 0{,}4\}$.

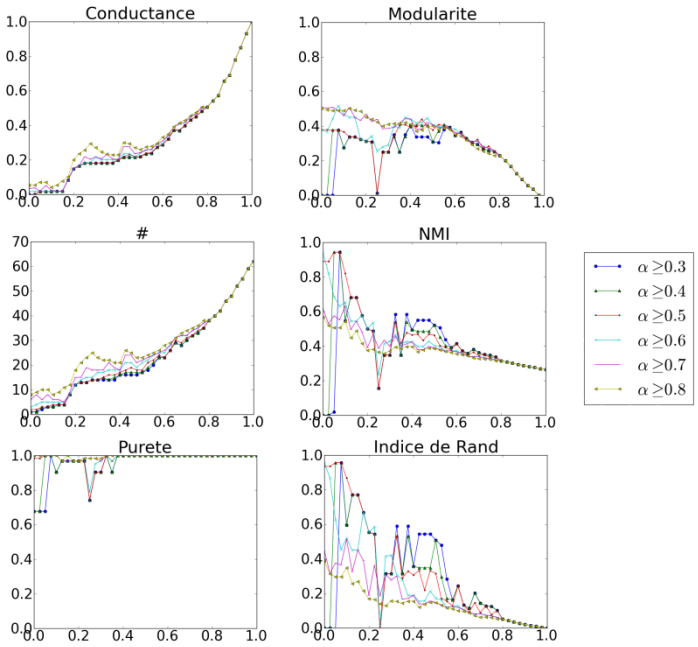

Figure 7. Résultat de la propagation de labels avec cœurs avec le réseau footballistique, avec différents pourcentages de barrages sur l'axe des abscisses

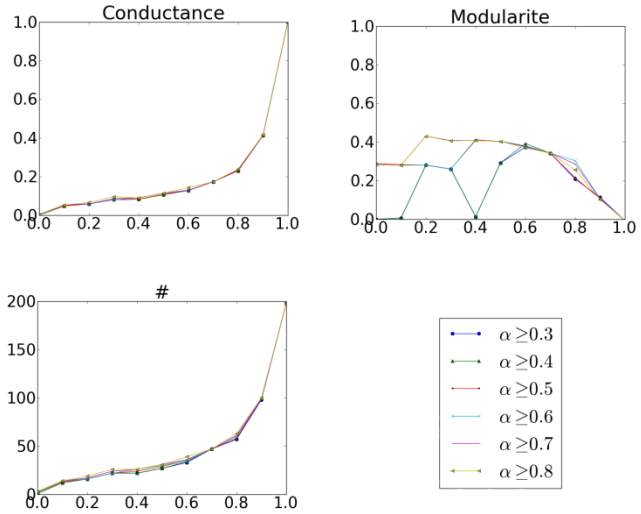

Figure 8. Résultat de la propagation de labels avec cœurs avec le réseau jazz, avec différents pourcentages de barrages sur l'axe des abscisses

Le réseau US-Air 97 est un graphe représentant des infrastructures pour le transport aérien entre le Canada, les États-Unis et le Mexique. Il comprend 332 nœuds et 2 126 arêtes. Les communautés représentent des infrastructures comme des aéroports, et les arêtes, leurs connexions. Cependant, elles y sont très rares, ce graphe ayant une distribution des degrés des nœuds assez uniforme. L'obtention de la modularité maximale requiert près de 60 % à 80 % de barrages, figure 9. Cependant les communautés obtenues après 80 % de barrages montrent une certaine proximité en termes de distance. Les petits aéroports proches des grands sont regroupés en communautés.

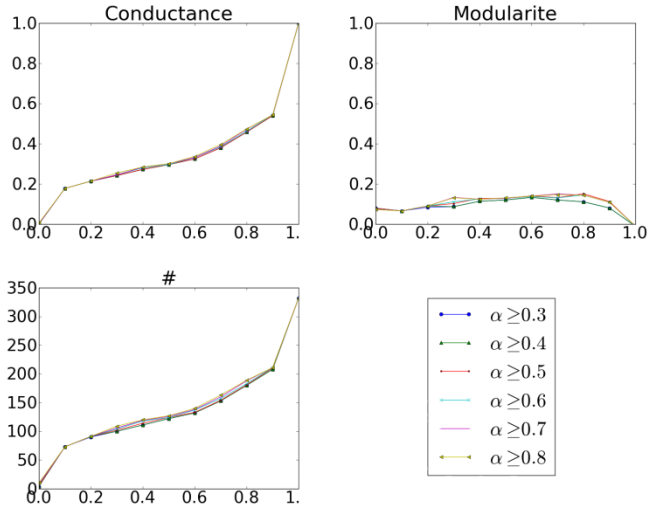

Figure 9. Résultat de la propagation de labels avec cœurs avec le réseau aérien US-Air 97, avec différents pourcentages de barrages sur l'axe des abscisses

Netscience est un cas d'étude où la mise en place de barrages ne permet pas l'amélioration de la qualité de partitionnement, figure 9. Les meilleurs résultats se situent avec CDLP, qui détecte selon les valeurs de α entre 420 et 500 communautés. Le paramètre influent est α, où avec $\alpha \geq \{0{,}8\}$, CDLP détecte pas moins de 500 communautés avec une modularité de 0,9 alors qu'avec $\alpha \geq 0{,}3$, CDLP détecte 420 communautés avec une modularité de 0,96. La taille des communautés est très faible. La mise en place de barrages ne permet pas l'obtention de meilleurs résultats en termes de qualité. D'après nos observations, l'ajout de barrages s'est effectué dans des zones densément connectées du graphe (souvent dans des sous-graphes complets), ce qui a pour conséquence de détériorer la qualité de partitionnement. Une convergence apparaît et les résultats pour tous les α sont les mêmes après 30 % de barrages.

4.4. Étude expérimentale portant sur le PLBS

Tableau 2. Résultats du PLBS

Experiences portant sur PLBS						
Algorithmes	Q	Φ	NMI	ARI	Pureté	#
Zac #2						
$\Delta_{[0,0;0,3]}$	0,131	0,031	0,228	0,091	0,676	3
$\Delta_{[0,3;0,6]}$	0,378	0,073	0,565	0,498	0,971	6
$\Delta_{[0,6;1,0]}$	0,073	0,550	0,377	0,068	1,0	23
Foot #11						
$\Delta_{[0,0;0,3]}$	0,599	0,012	0,911	0,852	0,913	12
$\Delta_{[0,3;0,6]}$	0,584	0,0419	0,931	0,907	0,956	16
$\Delta_{[0,6;1,0]}$	0,313	0,327	0,831	0,622	0,991	48
Dolphins #2						
$\Delta_{[0,0;0,3]}$	0,27	0,0684	0,547	0,597	0,903	3
$\Delta_{[0,3;0,6]}$	0,392	0,41	0,456	0,299	1,0	18
$\Delta_{[0,6;1,0]}$	0,281	0,694	0,329	0,071	1,0	35
Pol #3						
$\Delta_{[0,0;0,3]}$	0,496	0,029	0,565	0,671	0,857	6
$\Delta_{[0,3;0,6]}$	0,459	0,104	0,600	0,668	0,933	16
$\Delta_{[0,6;1,0]}$	0,212	0,313	0,434	0,194	0,962	45

A partir du tableau 2 les meilleurs résultats pour les réseaux Zac, Pol et Foot en termes de mesures supervisées sont en prenant l'intervalle $[0{,}3; 0{,}6]$ (en faisant donc varier le nombre de barrages de 30 à 60 %.), permettant l'alimentation de la matrice de co-appartenance. Pour Dol, il s'agit de l'intervalle $[0{,}0; 0{,}3]$. Nous notons que plus les bornes de l'intervalle ont des valeurs importantes (avec un pourcentage de barrages assez fort), plus le nombre de communautés est important. Cela s'explique du fait que la propagation ne peut plus s'effectuer dans certaines régions du graphe. Par voie de conséquence, plus le nombre de communautés est important, plus les densités de ces dernières augmentent, ce qui entraîne une augmentation de la conductance. On observe que la qualité des communautés se dégrade, avec un NMI et un ARI tendant vers 0.

4.4.1 Étude sur le temps d'exécution

Les spécificités de notre machine sont un disque dur de 140 GO, une Ram 64 542 (48 giga) (BI processeurs avec 4 cœurs chacun), 23 processeurs Intel 2 GHT et un Cpu de 1 200 MHZ. Nous utilisons python et la librairie *igraph*. Le temps d'exécution du PLBS se décompose en 3 éléments :

– le temps de calcul de la centralité d'intermédiarité} Δ_{CI}

– le temps pour mettre en place les barrages, l'allocation d'une matrice de fréquence et son alimentation par les différentes propagations de labels $\Delta_{Matrice(LPA)}$

– le temps pour la détection de composantes connexes correspondant à nos communautés Δ_{CC}

Cela peut se schématiser par la formule suivante : $\Delta_{T_{PLBS}} = \Delta_{CI} + \Delta_{Matrice(LPA)} + \Delta_{CC}$

Les parties de lecture et d'écriture n'ont pas été mises dans la mesure elles ne concernent pas le corps de l'algorithme.

Pour l'algorithme PLBS, nous prenons $\mathcal{N} = 100$, un intervalle de $\Delta_{[x,y]} = \Delta_{[0,0;0,3]}$ avec un pas $\Delta = 0{,}025$.

Tableau 3. Résultats sur le temps d'exécution du PLBS

Temps d'éxecution en secondes					
Réseau	$\|V\|$ et $\|E\|$	Δ_{CI}	$\Delta_{Matrice(LPA)}$	Δ_{CC}	$\Delta_{T_{PLBS}}$
Zac	34 \ 78	0,0004	0,280	0,003	0,284
Dol	62 \ 159	0,002	0,587	0,019	0,608
Pol	105 \ 441	0,007	4,158	0,217	4,382
Foot	115 \ 615	0,010	4,309	0,276	4,596
Jazz	198 \ 5 484	0,125	15,432	0,077	15,634
US-Air 97	332 \2 126	0,0947	16,935	0,226	17,256
Netscience	1589 \2 742	0,0614	91,717	1,217	93,098

Nous observons sur le tableau 3 que la partie prenant le plus de temps concerne l'alimentation de la matrice par les $\mathcal{N}$ propagations de labels. Pour Netscience, le temps commence à devenir conséquent avec près de 93 secondes. Le calcul de la centralité d'intermédiarité est assez rapide sur *igraph*, bien que sa complexité soit élevée.

MPLBS utilise quant à elle K matrices différentes avec différents niveaux de barrages. Le temps d'exécution du MPLBS se décompose comme suit :

– le temps de calcul de la centralité d'intermédiarité Δ_{CI}

– le temps pour mettre les barrages, l'allocation des K matrices de fréquence et leurs alimentations par les différentes propagations de labels $\Delta_{Matrice(LPA)}$

– le temps pour la détection de composantes connexes correspondant à nos communautés Δ_{CC}

– le temps de calcul de la fonction de qualité choisie, la modularité Δ_Q ou la conductance Δ_φ

Cela peut se schématiser par la formule : $\Delta_{T_{MPLBS}} = \Delta_{CI} + \Delta_{Matrice(LPA)} + \Delta_{CC} + \Delta_Q \wedge \Delta_\varphi$. Pour l'algorithme MPLBS, nous prenons un intervalle $\Delta = 0{,}025$ et $\mathcal{N} = 100$.

Tableau 4. Résultats sur le temps d'exécution du PLBS

Temps d'exécution en secondes				
Réseau	$\|V\|$ et $\|E\|$	Δ_{CI}	$\Delta_{Matrice(LPA)}$	Δ_{CC}
Zac	34 \ 78	0,00048	0,6423	0,0006
Dol	62 \ 159	0,0840	1,7110	0,0011
Pol	105 \ 441	0,01815	4,3550	0,0021
Foot	115 \ 615	0,0222	3,6145	0,0017
Jazz	198 \ 5484	0,1475	21,6762	0,0057
US-Air 97	332 \ 2 126	0,0947	98,3806	0,04204
Netscience	1 589 \ 2 742	0,0614	376,1667	0,02
Réseau	Δ_Q	Δ_Φ	$\Delta_{T_{MPLBS}}$	
Zac	0,0005	0,0680	0,712	
Dol	0,0005	0,0025	1,799	
Pol	0,0007	0,0022	4,378	
Foot	0,0009	0,9577	4,597	
Jazz	0,00110	0,6392	22,47	
US-Air 97	2,6306	0,4845	101,632	
Netscience	0,0068	73,9808	450,236	

Pour le calcul du temps global, nous sommerons le temps de calcul de la modularité et celui de la conductance. En observant les tableaux 3 et 4, nous voyons que le temps d'exécution du MPLBS est beaucoup plus important que celui du PLBS. Cela vient du fait qu'il y a K matrices de fréquence à alimenter. C'est également cette alimentation qui prend le plus de temps par rapport aux autres traitements, qui prennent beaucoup moins de temps. Le temps de calcul de la conductance et de la modularité ainsi que le temps de calcul des composantes connexes sont très rapides. La taille de la matrice à alimenter est fonction de la taille du graphe considéré. Plus le graphe aura de nœuds, plus le temps d'alimentation sera important.

4.5. Étude comparative

Pour notre étude comparative nous avons posé le nombre de propagation de labels à $\mathcal{N} = 100$. Pour le CDLP, nous prenons $\alpha \geq 0{,}5$. La présence du symbole "*" signifie que le résultat est une moyenne sur 100 lancements car instable, pour les algorithmes non déterministes. Le symbole # signifie le nombre de communautés.

Le tableau 7 expose la meilleure paramétrisation possible. Le symbole « * » signifie qu'il s'agit du résultat retourné par l'algorithme donnant la modularité la plus élevée. Les observations montrent que les algorithmes MPLBS, PLAB et CDLP proposés sont très compétitifs par rapport à ceux issus de la littérature, donnant de très bonnes valeurs pour le NMI et le ARI, ainsi que notre version finale PLBS. En effet, PLBS pour le réseau de football obtient un NMI de 0,931 et un ARI de 0,907.

Tableau 5. Étude comparative (Première partie)

Etude comparative avec d'autres algorithmes (partie 1)						
Algorithmes	*Q*	Φ	NMI	ARI	Pureté	#
Zac #2						
Louvain	0,4188	0,2739	0,49	0,3922	0,9411	4
Seifi	0,4188	0,0048	0,49	0,3922	0,9412	4
GN	0,4012	0,3175	0,4851	0,3915	0,9411	5
Spin	**0,4197**	0,2740	0,5878	0,4645	0,9705	4
Spectral	0,3934	0,335	0,579	0,435	0,970	4
Walk	0,3532	0,4034	0,4899	0,3207	0,9118	5
Hop	0,399	0,1784	0,3285	0,5151	0,9705	3
LPA	0,3368	**0,1513**	0,5749	0,5708	0,8952	2,61*
DPA	0,420		0,660			
Infomap	0,4020	0,1805	0,5683	0,5905	0,941	3
LICOD	0,24		0,60	0,62		3
PLBS	0,378	0,0728	0,5653	0,498	0,9706	6
PLAB	0,3991	0,1784	**0,6912**	**0,6841**	0,9706	3
MPLBS	0,402	0,1806	0,5684	0,5906	0,9412	3
CDLP	0,3761	0,242	0,6154	0.5932	0,9706	5
Dol #2						
Louvain	0,5185	0,2451	0,5109	0,3274	0,9677	5
Seifi	0,5185	0,0022	0,5109	0,3274	0,9677	5
GN	0,5193	0,2001	0,5541	0,3949	0,9838	5
Spin	0,5285	0,2339	0,5864	0,3734	1,0	5
Spectral	0,4912	0,2836	0,4489	0,2830	0,9516	5
Walk	0,4888	0,1798	0,5372	0,4167	0,9516	4
Hop	0,5189	0,2708	0,4811	0,2908	0,9677	6
LPA	0,4831	0,1720	0,6226	0,5024	0,9790	3,85*
DPA	**0,529**		0,774			
Infomap	0,5189	0,2708	0,4811	0,2908	0,9677	6
LICOD	0,35		0,41	0,32		2
PLBS	0,2749	0,0684	0,5466	0,5968	0,9032	3
PLAB	0,3761	0,0549	**0,9429**	**0,9563**	1,0	3
MPLBS	0,4569	0,0431	0,5979	0,6671	0,8476	2
CDLP	0,5176	0,1973	0,6313	0,4539	1,0	5

Tableau 6. Étude comparative (seconde partie)

Expériences avec d'autres algorithmes (partie 2)						
Algorithmes	Q	Φ	NMI	ARI	Pureté	#
Foot #11						
Louvain	0,6021	0,2778	0,855	0,7277	0,9217	9
Seifi	0,6021	0,0019	0,855	0,7277	0,8	9
GN	0,6005	0,290	0,8788	0,7781	0,8347	10
Spin	**0,6052**	0,293	0,8922	0,8165	0,86956	10
Spectral	0,4877	0,3608	0,6952	0,8909	0,6260	8
Walk	0,6038	0,2951	0,8874	0,8154	0,9217	10
Hop	0,5689	0,353	0,900	0,8361	0,9217	13
LPA	0,5946	0,2974	0,8865	0,7787	0,8544	10,39*
DPA	0,606		0,897			
Infomap	0,5790	0,335	0,9001	0,8527	0,9130	12
LICOD	0,49		0,83	0,69		16
PLBS	0,5844	**0,0419**	**0,9311**	**0,9066**	0.9565	16
PLAB	0,5985	0,3167	0,9289	0,9004	0,9391	13
MPLBS	0,6019	0,3102	0,9269	0,8893	0,9304	12
CDLP	0,3761	0,242	0,6154	0,5932	0,9706	5
Pol #3						
Louvain	0,5205	0,1587	0,5121	0,558	0,7238	4
Seifi	0,5205	0,0005	0,5121	0,558	0,8476	4
GN	0,5168	0,1098	0,5584	0,6823	0,8571	5
Spin	**0,5255**	0,2020	0,4655	0,4655	0,4388	6
Spectral	0,4671	0,2367	0,5201	0,5466	0,847	4
Walk	0,507	0,184	0,543	0,653	0,8476	4
Hop	0,4984	0,276	0,3478	0,3934	0,8666	7
LPA	0,4921	0,0863	0,5545	0,6507	0,8452	3,38*
DPA	0,527		**0,805**			
Infomap	0,5228	0,1642	0,493	0,5360	0,8476	6
LICOD	0,42		0,68	0,67		6
PLBS	0,4594	0,104	0,6006	0,6684	0,9333	16
PLAB	0,4949	0,0775	0,5734	**0,675**	0,8476	4
MPLBS	0,4946	**0,0692**	0,5861	0,6893	0,8476	3
CDLP	0,5062	0,1204	0,5406	0,6517	0,8571	7

Tableau 7. Paramétrisation de nos algorithmes

Foot #11	PLAB	PLBS	MPLBS
$\alpha \geq$	$\{0,6\}$	$\{0,5\}$	$\{0,5\}*$
$\beta =$	$\{0,05\}$	$\Delta_{[0,3;0,6]}$	$\{0,3\}*$
Zac #2	PLAB	PLBS	MPLBS
$\alpha \geq$	$\{0,6\}$	$\{0,5\}$	$\{0,5\}*$
$\beta =$	$\{0,1\}$	$\Delta_{[0,3;0,6]}$	$\{0,5\}*$
Dol #2	PLAB	PLBS	MPLBS
$\alpha \geq$	$\{0,6\}$	$\{0,5\}$	$\{0,3-0,5\}*$
$\beta =$	$\{0,05\}$	$\Delta_{[0,0;0,3]}$	$\{0,3\}*$
Pol #3	PLAB	PLBS	MPLBS
$\alpha \geq$	$\{0,6\}$	$\{0,5\}$	$\{0,3-0,5\}*$
$\beta =$	$\{0,3\}$	$\Delta_{[0,3;0,6]}$	$\{0,0\}*$

Nos expérimentations ont également montré qu'alimenter une matrice d'appartenance avec différents barrages pouvait donner de meilleurs résultats en terme de mesures supervisées dans la majeure partie des cas comme le montre PLBS par rapport au LPA et au CDLP. Imposer des barrages limite la propagation de labels et évite le fait de tomber dans de trop grandes communautés. C'est ce que montrent les résultats de PLAB qui nécessitent cependant de tester β. Pour les méthodes robustes, les expérimentations montrent qu'augmenter le nombre de barrages réduit la taille des communautés. Cela se traduit par une augmentation des valeurs de certaines paires de nœuds au sein de la matrice de co-fréquence. Lorsque le pourcentage de barrages est trop élevé, chaque nœud représente sa propre communauté, ce qui signifie de très fortes valeurs des éléments de la diagonale dans la matrice de co-fréquence. Ce phénomène permet de faire tendre le nombre de mauvaises propagations vers 0, et a également pour conséquence une augmentation de la conductance. Pour la paramétrisation, nous avons choisi $\alpha \geq 0{,}5$ pour tous les réseaux de PLBS. Un α trop fort donnerait de trop nombreuses communautés, et trop faible, risquerait de ne donner qu'une communauté. Le MPLBS utilisant la modularité, nous retourne la paramétrisation maximisant cette dernière, avec la partition résultante. Une dernière observation montre que PLBS et MPLBS donnent des conductances assez faibles en comparaison des autres algorithmes de la littérature, ceci s'expliquant par le fort nombre de barrages.

5. Conclusion

Dans cet article, nous avons exposé de nouveaux algorithmes basés sur la propagation de labels. Nos expérimentations ont montré que notre méthode hybride, liant propagation de labels et barrages issus de l'intermédiarité des arêtes permettait l'obtention de résultats satisfaisants. Nous avons également observé qu'alimenter une matrice de fréquences de co-apparition en faisant varier le nombre de barrages permettait d'augmenter la qualité des communautés obtenues avec une conductance faible. Notre analyse a montré que sur des réseaux réels, le pourcentage de barrages nécessaire pour l'obtention des meilleurs résultats était entre 20 % et 40 %. Nous adaptons actuellement ces algorithmes sur les grands graphes ayant plusieurs milliards d'arêtes. C'est en ce sens que nous développons une solution Hadoop (Dean et Ghemawat, 2008) et Spark (Zaharia *et al.,* 2010), qui nous permettra le développement de la propagation de labels en parallèle. Dans le cas de PLAB, une étude de l'estimation de α et β sera approfondie.

Bibliographie

Ana L., Jain A. K. (2003). Robust data clustering. *In Computer vision and pattern recognition,2003. proceedings. 2003 ieee computer society conference on*, vol. 2, p. II–128.

Bader D. A., Kintali S., Madduri K., Mihail M. (2007). Approximating betweenness centrality. *In Algorithms and models for the web-graph*, p. 124–137. Springer.

Barber M. J., Clark J. W. (2009). Fast unfolding of communities in large networks. *J. Stat. Mech*, p. P10008.

Batagelj V., Mrvar A. (2006). Pajek datasets.

Blondel V., Guillaume J., Lambiotte R., Mech E. (2008). Detecting network communities by propagating labels under constraints. *Physical Review E,* vol. 80, n° 2, p. 026129.

Brandes U. (2001). A faster algorithm for betweenness centrality. *Physical Review E, Journal of mathematical sociology,* vol. 25, n° 2, p. 163–177.

Dean J., Ghemawat S. (2008). Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, vol. 51, n° 1, p. 107–113.

Fortunato S. (2010). Community detection in graphs. *Communications of the ACM*, Physics Reports, vol. 486, n° 3, p. 75–174.

Geisberger R., Sanders P., Schultes D. (2008). Community detection in graphs., *In Proceedings of the meeting on algorithm engineering & expermiment*s, p. 90–100.

Girvan M., Newman M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences,* vol. 6, p. 565.

Gleiser P., Danon L. (2003). Community structure in jazz. *Advances in Complex Systems,* vol. 99, n° 12, p. 7821-7826.

Kannan R., Vempala S., Vetta A. (2004). On clusterings: Good, bad and spectral. *Journal of the ACM (JACM*), vol. 51, n° 3, p. 497–515.

Kanawati R. (2011). Licod: Leaders identification for community detection in complex networks, *ieee third international conference on social computing (socialcom)*, p. 577–582.

Krebs V. (2004). Books about US politics : *http://www.orgnet.com/*.

Leung I. X., Hui P., Lio P., Crowcroft J. (2009). Towards real-time community detection inlarge networks. *Physical Review E*, vol. 79, n° 6, p. 066107.

Liu X., Murata T. (2010). Advanced modularity-specialized label propagation algorithm for detecting communities in networks. *Physica A: Statistical Mechanics and its Applications,* vol. 389, n° 7, p. 1493–1500.

Lusseau D., Schneider K., Boisseau O. J., Haase P., Slooten E., Dawson S. M. (2003). The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, vol. 54, n° 4, p. 396–405.

Newman M. E. J. (2006). Finding community structure in networks using the eigenvectors of matrices. *Physical review E,* vol. 74, n° 3.

Newman M. E. J., Girvan M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E,* vol. 69, n° 2, p. 026113.

Ovelgönne M., Geyer-Schulz A. (2012). Le plaisir de l'interaction entre l'usager et les objets TIC numériques. An ensemble learning strategy for graph clustering. *Graph Partitioning and Graph Clustering,* vol. 588, p. 187.

Pons P., Latapy M. (2006). Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications*, vol. 10, n° 2, p. 191–218.

Raghavan U. N., Albert R., Kumara S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E,* vol. 76, n° 3, p. 036106.

Rand W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association,* vol. 66, n° 336, p. 846–850.

Ronhovde P., Nussinov Z. (2010). Local resolution-limit-free potts model for community detection. *Phys. Rev. E,* vol. 81, p. 046114.

Rosvall M., Bergstrom C. (2007). Maps of information flow reveal community structure in complex networks. *Technical report*.

Seifi M., Junier I., Rouquier J.-B., Iskrov S., Guillaume J.-L. (2013). Stable community cores in complex networks. *In Complex networks,* vol. 1, p. 87–98. Springer.

Shi J., Malik J. (2000). Normalized cuts and image segmentation. Pattern Analysis and Machine Intelligence, *IEEE Transactions on,* vol. 22, n° 8, p. 888–905.

Staudt,C,L. & Meyerhenke,H. Engineering High-Performance Community Detection Heuristics for Massive Graphs. *In the42nd International Conference on Parallel Processing IEEE*, (pp. 180-189)

Šubelj L., Bajec M. (2011). Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction. *Physical Review E*, vol. 83, n° 3, p. 036103.

Topchy A., Jain A. K., PunchW. (2005). Clustering ensembles: Models of consensus and weak partitions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. vol. 27, n° 12, p. 1866–1881

ZacharyW. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research,* vol. 33, p. 452-473.

Zaharia M., Chowdhury M., Franklin M. J., Shenker S., Stoica I. (2010). Spark: cluster computing with working sets. *In Proceedings of the 2nd usenix conference on hot topics in cloud computing,* p. 10–10.

Zong-Wen L., Jian-Ping L., Fan Y., Petropulu A. (2014). Detecting community structure using label propagation with consensus weight in complex network.*Chinese Physics B,* vol. 23, n° 9, p. 098902.