Une nouvelle approche topologique pour la recommandation de tags dans les folksonomies

Manel Hmimida

CONSERVATOIRE NATIONAL DES ARTS ET MÉTIERS | le cnam

École Doctorale du Conservatoire National des Arts et Métiers

DICEN & A^3

THÈSE DE DOCTORAT

présentée par : **Manel HMIMIDA**

soutenue le : **03 mars 2015**

pour obtenir le grade de : **Docteur du Conservatoire National des Arts et Métiers**

Discipline / Spécialité : **Sciences de l'information et de la communication**

Une nouvelle approche topologique pour la recommandation de tags dans les folksonomies

THÈSE DIRIGÉE PAR

M. Zacklad Manuel	*Professeur, CNAM-Paris.*
M. Kanawati Rushed	*Maître de conférences, LIPN-Villetaneuse.*

RAPPORTEURS

M. Ben Yahia Sadok	*Professeur, Faculté des sciences-Tunis.*
M. Laurent Dominique	*Professeur, Université de Cergy-Pontoise.*

EXAMINATEURS

M. Charnois Thierry	*Professeur, LIPN-Villetaneuse.*
M. Barkaoui Kamel	*Professeur, CNAM-Paris.*

Dédicace

Cette thèse est dédiée:

- **À** *l'âme de ma mère qui a toujours rêvé de partager avec moi les moments de bonheur, pour l'énorme sacrifice qu'elle avait fait pendant sa vie. Que ce travail soit pour elle l'expression de mon amour éternel.*

- **À** *mon cher père Ismail source de mon inspiration dans la vie qui n'a jamais douté de moi ni reculé devant aucun sacrifice à chaque fois qu'il était question de mon éducation, de mon avenir et de mon bonheur.*

- **À** *mon grand amour et cher mari Akram pour ses encouragements, ses sacrifices, sa présence, son écoute et son soutien aux moments les plus difficiles. Sans toi, cette thèse n'aurait été ni commencée ni terminée. Je te remercie aussi pour tout l'amour, et pour m'avoir toujours poussé en avant, faisant fin de mes doutes et mes objections. Je ne vais pas ajouter plus car tous les mots de toutes les langues restent incapables de me servir pour exprimer mes remerciements et mes gratitudes envers toi.*

- **À** *mon petit poussin qui a supporté des moments difficiles de stress et d'angoisse durant la grossesse. Pardon mon amour pour ce que je t'ai fait subir.*

- **À** *mes adorables sœurs Ines, Imen, Ibtissem, Amani, mes petites nièces Zeineb et Mariem et mes beaux frères Fadhel et Issam qui ont été une source de courage.*

- **À** *toute ma famille, ma belle famille et à tous mes ami(e)s.*

Remerciements

Cette thèse doit beaucoup aux nombreuses personnes qui m'ont encouragée, soutenues et confortées à l'élaboration de ce mémoire de thèse. Qu'elles trouvent dans ce travail l'expression de mes plus sincères remerciements.
Je tiens à remercier vivement mon directeur de thèse **Manuel Zacklad** de m'avoir donné l'occasion de travailler sur ce sujet riche, actuel et passionnant. Je tiens à le remercier pour toutes les remarques et discussions constructives que l'on a pu avoir et qui m'ont permis de progresser.
Je dois beaucoup à mon encadrant **Rushed Kanawati**, de m'avoir encadré et pour la confiance qu'il m'a accordée, pour ses conseils constructifs et sa constante disponibilité et sans qui cette thèse n'aurait pas vu le jour. Je tiens ici à lui témoigner mon admiration, ma gratitude et mon profond respect.
Je tiens à exprimer toute ma gratitude aux professeurs **Sadok ben Yahia** et **Dominique Laurent** d'avoir accepté d'être les rapporteurs de ce travail.
Je remercie également les professeurs **Thierry Charnois** et **Kamel Barkaoui** d'avoir accepté d'examiner cette thèse.
Je souhaite aussi remercier tous les gens avec qui j'ai collaboré pendant la réalisation de cette thèse, en particulier Manisha Pujari et Issam Falih.
J'adresse autant de remerciements à tous les membres de Dicen et de Lipn, notamment Younès Bennani et Céline Rouveirol, pour leur sympathie, toutes les discussions et tous les conseils qui m'ont été très précieux.
Un grand merci à Lamjed Ben Jabeur, Nizar Chatti, Nathalie Leborgne et Ahmed Ben Salah qui m'ont soutenu et aidé à réaliser ce travail.
Enfin, je remercie tous mes amis qui étaient à mes côtés dans les moments difficiles notamment Marie-Hélène, Houda, Manel et Ilhem.

Résumé

Nous nous intéressons dans cette thèse à la problématique de recommandation de tags dans les systèmes de partage et de classification sociale des ressources, dits *folksonomies*. Les utilisateurs annotent les ressources à partager par des *tags* librement choisis. Ces tags servent alors d'index pour faciliter l'accès rapide aux ressources partagées. Or, la liberté de choix de tags les rend *ambigus*. Nous proposons une nouvelle approche *topologique* nommée TLTR (Two Level Tag Recommendation) pour la recommandation de tags. Les approches topologiques s'appuient exclusivement sur l'analyse de la structure du graphe (ou hypergraphe) modélisant une folksonomie pour le calcul des tags à recommander à un utilisateur pour annoter une ressource. Ces approches présentent l'avantage de ne pas recourir à l'analyse des contenus des ressources ni à l'élaboration des profils utilisateurs. Ceci les rend génériques et applicables pour différents types de folksonomies. Or, la grande taille des graphes représentant les folksonomies actuelles rend l'application de ces approches problématique. TLTR est basée sur une approche originale de compression des graphes. Le graphe d'une folksonomie est compressé en appliquant une méthode de clustering sur chacune des trois composantes d'une folksonomie, à savoir: l'ensemble des utilisateurs, l'ensemble des ressources et l'ensemble des tags. Nous proposons également une méthode de clustering topologique basée sur une approche centrée graine pour la détection des communautés dans les graphes multiplexes. Une approche topologique classique, en l'occurrence la méthode *Folkrank*, est appliquée sur le graphe réduit afin de sélectionner les clusters de tags les plus appropriés. Ces clusters sont ensuite utilisés pour construire un autre graphe contextuel extrait du graphe original représentant la folksonomie. La méthode *Folkrank* est à nouveau appliquée afin de calculer la liste de tags à recommander. Des expérimentations sur de grandes folksonomies, notamment, des jeux de données extraits du système de partage des références bibliographiques *Bibsonomy* montrent la pertinence de notre approche.

Mots clés: Recommandation de tags, Réseaux complexes, Réseaux multiplexes, TLTR, Détection des communautés.

Abstract

We focus in this thesis on the problem of tag recommendation in social sharing and classification systems called *folksonomies*. Users of a folksonomy annotate their resources with *freely* tags chosen. These tags are then used as an index to provide fast access to shared resources. However, the freedom in the tag selection makes them *ambiguous*. Tags recommender systems aim to help users in selecting the most appropriate tags for annotating a resource. Different approaches are proposed in the literature. We propose here a new *topological* approach for tag recommendation. Topological approaches are based exclusively on the analysis of the structure of the graph (or hypergraph) modeling a folksonomy. These approaches have the advantage of not depending on resources content analysis neither on construction user profiles. This makes them generic and applicable to different types of folksonomies. However, the large size of graphs representing current folksonomies makes the application of these approaches difficult. We propose a new approach, called TLTR (Two Level Tag Recommendation), which is based on an original approach of graph compression. The graph of a folksonomy is compressed by clustering each of the three components, namely the set of users, resources and tags. A topological clustering method based on a seed-centered approach for community detection in multiplex graphs (i.e. multi-relational) is proposed. A classical topological approach, namely *Folkrank*, is applied to the reduced graph to select the most appropriate clusters of tags. These clusters are then used to build another contextual graph extracted from the original graph representing the folksonomy. *Folkrank* method is applied again to compute the list of tags to recommend. Experiments on large folksonomy, including, data extracted from references system *Bibsonomy* show the relevance of our approach.

Keywords: Tags Recommendation, Complex network, Multiplex network, TLTR, Community detection.

Table des matières

Liste des tableaux

Table des figures

Première partie

Introduction Générale

Chapitre 1

Introduction

1.1 Contexte

Le terme folksonomie a été forgé par Thomas Vander Wal en combinant les deux termes Folk (les usagers) [1] et taxonomie (règles de classification). Une folksonomie est un système de classification de ressources permettant d'annoter et de catégoriser le contenu des documents numériques [Peters 2009]. Un des éléments principaux d'une folksonomie est le tag (ou étiquette). Ce terme désigne des symboles notamment des mots clés pouvant être associés à des ressources en vue de regrouper et de retrouver des informations ayant les mêmes tags. C'est pour cela que les tags sont choisis et attribués librement par les utilisateurs en fonction de leurs besoins.

Une folksonomie (voir figure 1.1) est constituée de trois entités fondamentales: les utilisateurs, les tags et les ressources. Contrairement aux taxonomies où les tags sont structurés en hiérarchie, les folksonomies sont caractérisées par des tags ayant une structure plate. Une définition formelle a été introduite par [Hotho *et al.* 2006] comme suit:

Définition 1 *Une folksonomie est un ensemble de tuples* $F := (U, T, R, Y)$ *où* U, T *et* R *sont des ensembles finis dont les éléments désignent respectivement les utilisateurs, les tags et les ressources.* $Y \subseteq U \times T \times R$ *est une relation ternaire où chaque élément* $y \in Y$ *peut être représenté par un triplet:* $y = \{(u, t, r) | u \in U, t \in T, r \in R\}$ *décrivant le fait qu'un utilisateur* u *a annoté la ressource* r *avec le tag* t.

1. Folksonomy, Vander Wal Thomas online posting, 2007

La simplicité des folksonomies réside dans le fait qu'elles n'exigent aucun consensus, contrairement aux taxonomies qui exigent que les termes soient prédéfinis. En effet, la démarche méthodologique de conception des folksonomies repose sur un certain nombre d'étapes qu'on pourrait résumer comme suit: d'abord, les utilisateurs créent librement leurs propres tags pour annoter des ressources telles que: des pages web, des photos, des vidéos. Par la suite, ces tags sont utilisés pour gérer et classer le contenu des corpus dynamiques en ligne. La collection de toutes les annotations d'un l'utilisateur est appelée personomie.

Définition 2 *Une personomie P_u d'un utilisateur $u \in U$ est la restriction de F à u, c'est-à-dire $P_u := (T_u, R_u, I_u)$ avec $I_u := \{(t,r) \in T \times R | (u,t,r) \in Y\}$. Nous définissons $T_u := \pi_1(I_u)$, $R_u := \pi_2(I_u)$ où π_i indique la projection sur la i-ème dimension.*

L'utilisateur peut explorer sa personomie ainsi que l'ensemble de la folksonomie dans toutes les dimensions. De plus, en sélectionnant une ressource, l'utilisateur peut voir les autres utilisateurs qui l'ont partagée ainsi que les tags qu'ils lui ont affectés. Un utilisateur peut également rechercher et trouver ses propres ressources ou les ressources des autres utilisateurs au sein d'un système de tagging et de partage collaboratif.

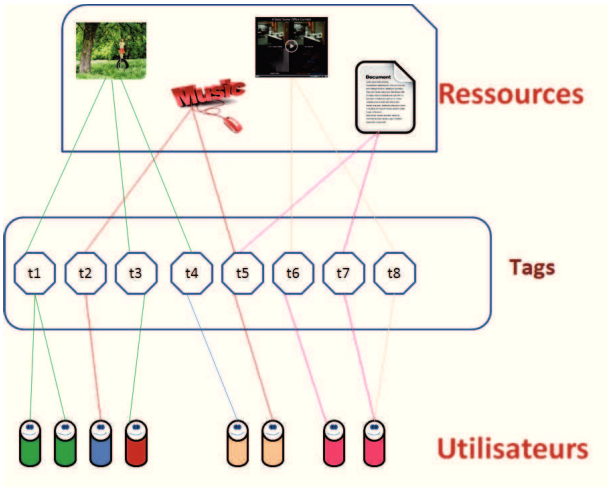

FIGURE 1.1 – Les composantes d'une folksonomie

Différents types de folksonomies: les folksonomies ont été présentées dans [Vander Wal 2005]. Vander distingue deux types de folksonomies: les étroites (narrow) et

les larges (broad) illustrés dans la figure 1.2.

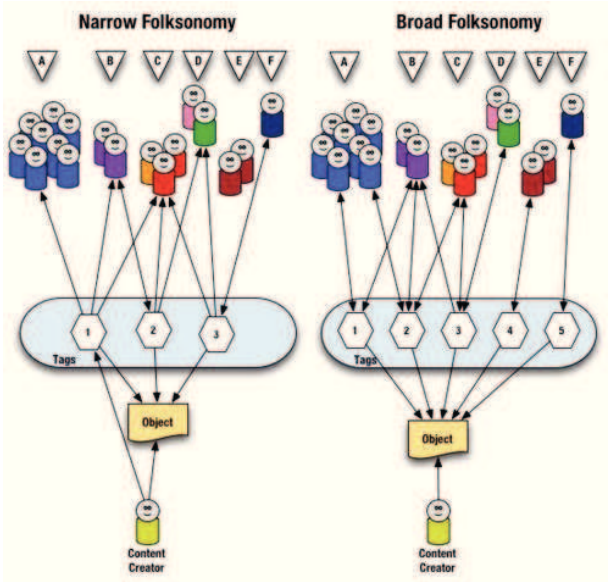

FIGURE 1.2 – Les deux types de folksonomies: large (Broad) et étroite (Narrow) [Vander Wal 2005]

.

– ***Les folksonomies étroites (narrow):*** les ressources dans ce type de folksonomie sont principalement annotées par l'utilisateur l'ayant ajouté. Elles sont utilisées dans un objectif individuel comme le montre la figure 1.3. Flickr [2] et Youtube [3] sont deux exemples de folksonomies étroite. Ils consistent respectivement à partager des photos et des vidéos où l'utilisateur est le seul en mesure d'assigner ses tags à ses ressources.

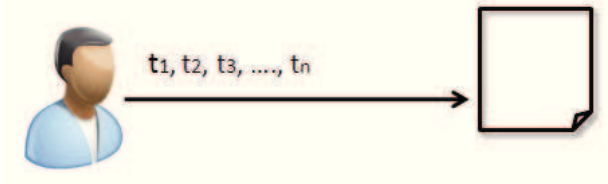

FIGURE 1.3 – Tagging simple

– ***Les folksonomies larges (broad):*** ce type de folksonomie favorise l'aspect collectif et collaboratif du partage de l'information. De nombreuses personnes différentes

2. http://www.flickr.com/
3. http://www.youtube.com/

peuvent annoter le même document comme le montre la figure 1.4. Elles décrivent le contenu du document de leur point de vue avec un effet sur le choix des tags qui peuvent être similaires ou complètement différents.

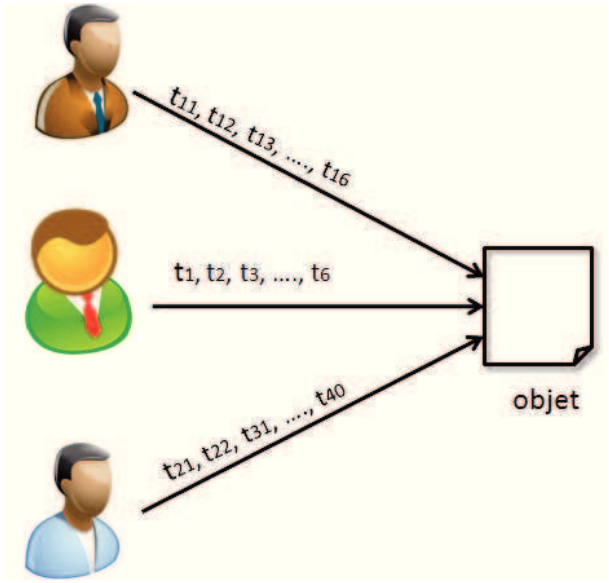

FIGURE 1.4 – Tagging collaboratif

Parmi les folksonomies larges les plus connues, nous citons Delicious [4] qui est un outil pour le partage des signets web. Les sites de tagging social présentent des nuages de tags. Ces nuages représentent explicitement l'ensemble des tags les plus populaires sur le site et dont la taille de police est variable. En effet, cette taille est proportionnelle à la fréquence d'utilisation des tags pour annoter les ressources. Plus la taille d'un tag est grande, plus ce dernier est très utilisé par les utilisateurs. La figure 1.5 montre un nuage de tags sur Delicious, il indique que les tags *blog*, *books* font partie des tags les plus populaires sur le site.

FIGURE 1.5 – Delicious: nuage de tags

4. https://delicious.com/

Le fait que les utilisateurs soient libres de générer leurs propres tags dans l'annotation des ressources engendre des confusions et des polysémies. En effet, le problème de vocabulaire incontrôlé conduit à un certain nombre de limitations. Notamment, le problème d'ambiguïté de tags qui peut être décrit par les deux cas suivants:

- Les utilisateurs utilisent le même tag pour annoter des ressources qui sont sémantiquement différentes.
- Les utilisateurs utilisent des tags différents pour annoter la même ressource.

La figure 1.6 montre une capture du site Flickr où nous constatons la présence de plusieurs tags pour annoter une seule image.

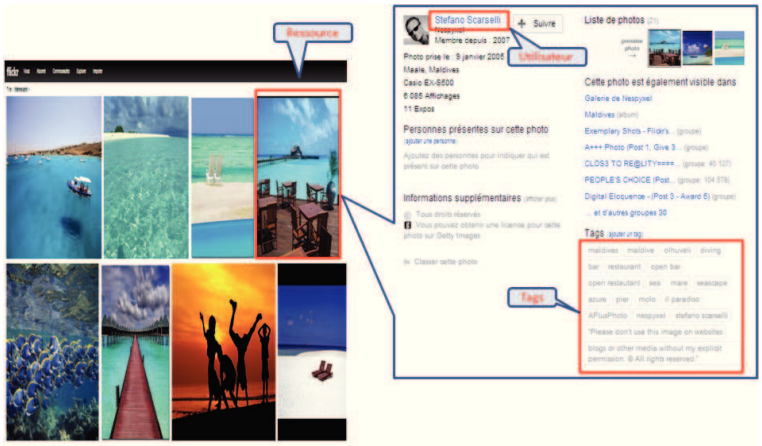

FIGURE 1.6 – Une capture de site Flickr

À titre d'exemple, les tags "US", "U.S" et "U.S.A" sont tous utilisés pour désigner les États Unis (United States of America). La plupart des systèmes de tagging autorisent l'emploi des mots simples. Par exemple, Delicious n'autorise pas les espaces dans les tags contrairement à certains systèmes comme Flickr. Dans certains cas, un tag est composé de plusieurs mots sans espaces. Néanmoins, les utilisateurs peuvent décrire une hiérarchie par un seul tag composé ou bien une catégorie par plusieurs termes tels que conception/Merise sur Delicious.

Les systèmes de recommandation de tags (SRT) tentent de faire face au problème d'ambiguïté de tags en aidant l'utilisateur à sélectionner ceux qui sont les plus appropriés pour annoter une ressource donnée. Plusieurs travaux ont été proposés dans la littérature

scientifique pour remédier à ce problème ([Jäschke *et al.* 2007], [Jäschke *et al.* 2008], [Lee et Chun 2007], [Lipczak *et al.* 2009]).

Définition 3 *Un système de recommandation (SR) est un système capable de fournir des recommandations personnalisées permettant de guider l'utilisateur vers des ressources utiles au sein d'un espace de données de taille importante [Burke 2002]. Une définition plus formelle de la recommandation est donnée par [Adomavicius et Tuzhilin 2005]; Soient* $U = \{u_1, ..., u_M\}$ *un ensemble d'utilisateurs,* $I = \{i_1, ..., i_N\}$ *l'ensemble de tous les items qui peuvent être recommandés et* $g : U \times I \rightarrow R$ *avec* R *un ensemble ordonné et* $g(u_m, i_n)$ *une fonction qui mesure le degré d'intérêt que portera l'utilisateur* u_m *à l'item* i_n*. Alors, pour chaque utilisateur* $u \in U$*, le système de recommandation sélectionne l'item* $i^{max,u} \in I$ *qui maximise l'intérêt de* u*:*

$$\forall u \in U, i^{max,u} = \arg\max_{i \in I} g(u, i) \tag{1.1}$$

Nous nous intéressons dans cette thèse au problème de recommandation de tags. En effet, en considérant un utilisateur donné $u \in U$ et une ressource $r \in$R, un ensemble $\hat{T}(u, r)$ de tags est retourné comme recommandation. Souvent, $\hat{T}(u, r)$ représente les n premiers tags calculés en fonction d'un critère de pertinence. Un tag peut être jugé pertinent pour indexer une ressource donnée, en s'appuyant soit sur les avis des utilisateurs voisins de la même communauté, soit en prenant en compte les avis des experts de domaine, soit sur la base du profil personnel de l'utilisateur.

$$\hat{T}(u, r) := \underset{t \in T}{\text{argmax}}^{n} \sum_{v \in \mathcal{N}_n^k} sim(x_u, x_v)\delta(v, t, r) \tag{1.2}$$

avec $\delta(v, t, r) = 1$ si l'utilisateur v a annoté une ressource r avec le tag t, 0 sinon.

1.2 Contribution

Ce travail de thèse s'inscrit dans le cadre des approches topologiques de recommandation de tags. Nous proposons une approche qui opère à double niveau. Le premier niveau permet de travailler sur la recommandation au niveau général. Concrètement, cette étape permet d'identifier des groupes de tags, sémantiquement proches, qui sont potentiellement

liés avec des groupes d'utilisateurs, partageant les mêmes centres d'intérêt, et des groupes de ressources, portant sur les mêmes thématiques. Le deuxième niveau permet d'exploiter les résultats du premier niveau afin de limiter le périmètre de recommandation de tags. Les principales contributions de ce travail peuvent être résumées comme suit:

1. Nous avons développé une approche qui consiste à déterminer les tags pertinents à recommander à l'utilisateur tout en rajoutant une couche supplémentaire de contraction du graphe initial de la folksonomie. Cette couche assure le filtrage des tags en éliminant dès le départ les tags non intéressants par rapport au contexte de l'utilisateur. Par conséquent, nous réduisons considérablement la quantité de données de départ. Tout d'abord, nous appliquons l'algorithme de recommandation Folkrank [Hotho *et al.* 2006] sur le graphe réduit afin de sélectionner les clusters de tags les plus appropriés. Ces clusters sont ensuite utilisés pour construire un autre graphe contextuel de taille beaucoup plus réduite extrait du graphe initial représentant la folksonomie. La méthode Folkrank est à nouveau appliquée afin de calculer la liste de tags à recommander.

2. Afin de générer le graphe réduit, il était nécessaire de généraliser les concepts en utilisant un algorithme de détection de communautés. Nous proposons une nouvelle approche pour la détection de communautés dans les réseaux multiplexes. En effet, la caractéristique multidimensionnelle de ce type de réseaux permet d'exploiter un maximum d'informations. Notre approche est une généralisation directe de l'algorithme Licod [Yakoubi et Kanawati 2014], proposé initialement pour traiter les graphes monoplexes, au cas d'un multiplexe. Le principe de notre approche repose sur le fait que les communautés se forment autour des nœuds appelés Leaders. C'est pour cela que nous commençons par calculer ces derniers puis nous appliquons une méthode d'expansion autour de ces valeurs calculées dans le but d'identifier les communautés dans le réseau.

3. Nous présentons les résultats de notre contribution sur des graphes réels de taille moyenne obtenus à partir de différents types de données comme les données bibliographiques de Dblp et du système de partage des signets Bibsonomy.

1.3 Liste des publications

- **Revue internationale**

  1. **Hmimida M.**, Kanawati R., Community detection in multiplex networks: a seed-centric approach, *Networks and Heterogeneous Media: Special Issue on New trends, models and applications in Complex and Multiplex Networks*, Volume 10, Issue 1, pp71-85, March 2015

- **Chapitre de livre**

  1. **Hmimida M.**, Kanawati R., Système de recommandation par niveaux pour la classification de facettes pour la recherche documentaire. *Chapitre dans Systèmes de recommandation*, Ghislaine Chartron, Imad Saleh, Gérald Kembellec (éditeurs). Hermès, 2014.
  2. **Hmimida M.**, Kanawati R., A Two-level Recommendation Approach for Document Search. *Chapter in Recommender systems*, Ghislaine Chartron, Imad Saleh, Gérald Kembellec (editors). ISTE 2014, pp119-132.

- **Conférences**

  1. Falih I., **Hmimida M.**, Kanawati R., Une approche centrée graine pour la détection de communautés dans les réseaux multiplexes. *15ème conférence internationale sur l'extraction et la gestion des connaissances EGC2015*, (papier court), mars 18-20 Paris 2015.
  2. **Hmimida M.**, Kanawati R., A seed-centric algorithm for community detection in multiplex networks, *First European Social Networks Conference*, July 1-4 2014 Barcelona.
  3. Ankoud M., **Hmimida M.**, Étude de l'évaluation des ECMs, *ISKO-Maghreb 2013*, 8-9 novembre 2013, Marrakech-Maroc, IEEE pp.1-5.
  4. **Hmimida M.**, Kanawati R., Ankoud M., Nouveau modèle de recommandation pour la classification à facettes, *15ème édition du Colloque international sur le Document Électronique CIDE 15 Tunis*, 1-3 novembre 2012 Tunisie, Europia pp.145-157.

5. **Hmimida M.**, Ankoud M., Recommendation level in faceted classification for documentary classification *ICEELI'2012*, 1-3 juillet 2012 Sousse Tunisia, IEEE pp.1-5.
6. Ankoud M., **Hmimida M.**, Un modèle d'évaluation d'un SOC dans l'environnement d'ECM Prototype: HyperTaging, *ISKO-Maghreb'2012* : Concepts et Outils pour le Management de la Connaissance, 3-4 novembre 2012 Hammamet Tunisie.
7. Salzano G., Ankoud M., **Hmimida M.**, Zacklad M., Gestion des évolutions dans un SOC d'entreprise, multidimensionnel et distribué *29 édition d'INFORSID* , 24-26 mai 2011 Lille.

– **Ateliers**

1. Falih I., **Hmimida M.**, Kanawati R., Community detection in multiplex network: a comparative study. *European conference on complex systems (ECCS'14)*, Satellite workshop on multiplex networks 24 september 2014 Lucca.
2. Falih I., **Hmimida M.**, Kanawati R., Détection de communautés dans les réseaux multiplexes: étude comparative. *5ième Journée thématique: Fouille de grands graphes (JFGG'14)*, 15-17 octobre 2014 Paris.
3. **Hmimida M.**, Approche de recommandation de tags par niveau, *3ième journée de fouille de grands graphes JFGG*, 17-18 octobre 2012 Villetaneuse.

1.4 Plan général

Outre l'introduction générale, ce rapport est composé de cinq chapitres. Dans le chapitre 2 nous présentons une étude de l'état de l'art des systèmes de recommandation de tags. Nous exposons les quatre classes d'approches de recommandation de tags: le filtrage collaboratif, les approches basées sur le contenu, les approches hybrides et les approches topologiques. À la fin de ce chapitre, nous concluons par une analyse critique de ces différentes approches. Dans le chapitre 3, nous présentons un état de l'art sur les différents travaux qui concernent les algorithmes de détection de communautés dans les réseaux multiplexes. Une classification des approches de détection de communautés en deux catégories

est proposée. La première catégorie concerne les approches qui visent à ramener le problème de détection de communautés dans un graphe multiplexe en un graphe simple (monoplexe). En revanche, la deuxième catégorie concerne les approches qui visent à garder la structure multiplexe du graphe. Ces approches consistent à généraliser une méthode de détection de communautés d'un graphe simple pour qu'elle traite simultanément les différentes couches du graphe multiplexe.

Dans le chapitre 4, nous introduisons notre approche de détection de communautés dans les réseaux multiplexes appelée *Mux-Licod*. Les nouvelles mesures topologiques utilisées sont également définies et permettent ainsi de travailler sur les graphes multiplexes tels que: la mesure de voisinage, le plus court chemin et le calcul de degré d'un nœud dans le cadre multiplexe. Enfin, une étude comparative entre notre approche et les travaux existants étudiés dans l'état de l'art est établie. Cette étude est menée sur différentes bases de données et repose sur deux indicateurs principaux: la mesure de redondance et la mesure de la modularité. Les expérimentations montrent que notre approche Mux-Licod est nettement supérieure aux autres travaux en fonction de ces indicateurs topologiques.

Le chapitre 5 est consacré à la présentation de notre approche de recommandation de tags par niveau TLTR[5]. Celle-ci permet d'améliorer la qualité de la recommandation en réduisant le graphe initial de la folksonomie. D'abord, nous présentons le modèle de notre système. Ensuite, nous montrons toutes les étapes permettant d'assurer la contraction du graphe, en passant par les projections de graphe afin d'obtenir la structure unipartite de différentes couches de notre réseau multiplexe décrivant notre graphe. Les expérimentations ont été menées sur les données de Bibsonomy afin d'évaluer la performance du système TLTR en terme de précision de la recommandation des tags. Dans le chapitre 6, nous concluons ce rapport et nous proposons les principales perspectives.

5. Two Level Tags Recommendation

Deuxième partie

État de l'art

Chapitre 2

Systèmes de recommandation de tags

2.1 Introduction

L'accroissement constant de la quantité de l'information dans les folksonomies ainsi que la liberté offerte à l'utilisateur pour créer ses tags complexifient la tâche de navigation dans une telle structure notamment pour chercher et repérer les ressources (image, vidéo, article, etc). Cette difficulté est traduite par des problèmes d'ambiguïté de tags.

Aujourd'hui, les folksonomies incorporent des systèmes de recommandation de tags afin de limiter l'effet de ces problèmes. En effet, ces systèmes proposent à l'utilisateur les tags les plus pertinents pour annoter ses ressources. Ils peuvent intervenir au niveau de deux tâches:

- La tâche d'indexation en aidant l'utilisateur à sélectionner les tags les plus appropriés pour annoter ses ressources et/ou
- La tâche de recherche des informations au sein des répertoires en faisant une recherche par tags.

Nous introduisons des exemples des travaux de recommandation de tags qui supportent de différents types de données. Les ressources peuvent correspondre à différents types de données tels que des films ([Szomszor *et al.* 2007] et [Said *et al.* 2010]), de la musique [Eck *et al.* 2007], des messages dans les blogs [Mishne 2006], des articles scientifiques ([Jäschke *et al.* 2007] et [Song *et al.* 2008b]) et des photos ([Garg et Weber 2008], [Sigurbjörnsson et Van Zwol 2008] et [Viana *et al.* 2013]). Nous rappelons que le principe général des

systèmes de recommandation consiste à chercher/recommander les tags les plus pertinents pour une ressource et un utilisateur donnés.
Ce chapitre suit le plan suivant: nous commençons par présenter une classification des différentes approches de recommandation de tags en particulier les quatre catégories suivantes: les approches basées sur le contenu, le filtrage collaboratif, les approches topologiques et les approches hybrides (basées à la fois sur le contenu et sur la topologie). Dans les sections qui suivent, nous détaillons chacune de ces approches et à la section 2.3, nous présentons une étude critique de ces travaux. Ce chapitre se termine par une conclusion.

2.2 Classification d'approches de recommandation de tags

Historiquement, les systèmes de recommandation sont classifiés en deux grandes familles: *les approches basées sur le contenu* et *le filtrage collaboratif*. Dans le cas où les folksonomies sont représentées sous forme de graphe, une troisième famille d'approche *topologique* peut être définie. En plus, l'utilisation conjointe de filtrage collaboratif et des approches basées sur le contenu génère un autre type d'approche nommé *les approches hybrides*. Dans cette section, nous présentons quatre classes d'approches de recommandation de tags: le filtrage collaboratif, les approches basées sur le contenu, les approches topologiques et les approches hybrides.

- Le filtrage collaboratif: il dépend des opinions et des évaluations des autres groupes utilisateurs [Marinho *et al.* 2011], [Parra et Brusilovsky 2009].
- Les systèmes basés sur le contenu: ils sont fondés seulement sur l'analyse de contenu textuel lié à la ressource [Song *et al.* 2008a], [Musto *et al.* 2010].
- Les approches topologiques: ces approches sont à base de graphes où les systèmes de recommandation exploitent les relations entre les utilisateurs, les ressources et les tags représentés dans le graphe de la folksonomie pour générer les recommandations [Marinho *et al.* 2011].
- Les approches hybrides: consistent à fusionner à la fois plusieurs types d'approches de recommandation de tags comme dans le cas d'une approche collaborative basée sur le contenu [Gemmell *et al.* 2010].

Dans les sections qui suivent, nous décrivons chacune de ces approches.

2.2.1 Le filtrage collaboratif

Le filtrage collaboratif (FC) est la catégorie la plus populaire des algorithmes de recommandation. Le principe de ce type d'algorithme est de recommander à un utilisateur donné, les items évalués positivement par des utilisateurs similaires [Shardanand et Maes 1995]. Le terme FC a été utilisé pour la première fois dans [Goldberg *et al.* 1992] pour décrire le système de filtrage *Tapestry*. Il s'agit d'un système de recherche et de recommandation permettant aux utilisateurs d'accéder aux documents et aux messages électroniques à travers les requêtes et les appréciations des autres utilisateurs similaires. Parmi les premiers travaux, on trouve le système de GroupLens permettant la recommandation des articles de presse [Resnick *et al.* 1994] et le système RINGO pour la recommandation de musique [Shardanand et Maes 1995].

En général, il en existe deux types de méthodes: soit basée sur *les utilisateurs*, soit basée sur *les items*. Ces deux méthodes peuvent également être fusionnées ([Wang *et al.* 2006], [Tso-Sutter *et al.* 2008]).

Filtrage basé sur les utilisateurs: l'algorithme FC basé sur l'identification des utilisateurs similaires (FCU) consiste à prédire les intérêts de l'utilisateur pour une ressource donnée en exploitant un ensemble de profils utilisateurs. Prenons comme exemple, un système de recommandation qui prédit les évaluations sur des ressources ou suggère une liste de nouvelles ressources les plus pertinentes pour un utilisateur. Traditionnellement, pour m utilisateurs et n ressources, les profils utilisateurs sont représentés par une matrice utilisateur-ressource $X \in \mathbb{R}^{m \times n} \cup \{.\}$ avec $\{.\}$ désigne les valeurs manquantes. La matrice peut être décomposée en vecteurs lignes:

$$X := [X_1, ..., X_m]^T \text{ avec } X_u := [x_{u,1}, ..., x_{u,n}], \text{ pour chaque } u := 1, ..., m. \tag{2.1}$$

avec $x_{u,r}$ indique que l'utilisateur u a évalué la ressource r par la note $x_{u,r} \in \mathbb{R}$. Chaque vecteur ligne X_u correspond au profil utilisateur en représentant les évaluations des ressources par l'utilisateur u. Cette décomposition permet de déterminer la valeur de similarité utilisateur-utilisateur [Resnick *et al.* 1994].

Filtrage basé sur les items: l'algorithme FC basé sur les items (FCI) consiste à calculer la similarité entre les items. Le but étant de recommander à l'utilisateur les items les plus similaires à son profil. La matrice est alternativement représentée par des vecteurs colonnes:

$$X := [X_1, ..., X_n] \text{ avec } X_r := [x_{1,r}, ..., x_{m,r}]^T, \text{ pour chaque } r := 1, ..., n. \tag{2.2}$$

dans lequel chaque vecteur colonne $x_{m,r}$ correspond aux notes données par tous les m utilisateurs à une ressource spécifique. Cette représentation assure le calcul de la similarité item-item [Deshpande et Karypis 2004].

À cause de la nature des relations ternaires des folksonomies, le filtrage collaboratif classique ne peut pas être appliqué directement. Cependant, le principe de base de l'application de filtrage collaboratif (FC) dans les folksonomies pour la recommandation des tags est très similaire à celui de filtrage collaboratif classique. En effet, il consiste à recommander des tags en s'appuyant sur l'hypothèse que les utilisateurs similaires ont des comportements et des goûts équivalents. L'idée est de proposer de nouveaux objets ou de prédire un certain nombre d'objets en fonction des avis positifs (évaluations) des utilisateurs [Sarwar *et al.* 2001]. Afin de pouvoir appliquer le FC sur des folksonomies, [Marinho *et al.* 2011] proposent une approche permettant de réduire la relation ternaire Y. À cette fin, il a considéré X la matrice présentant alternativement les deux projections en deux dimensions $\pi_{UR}Y \in \{0,1\}^{|U|\times|R|}$ avec $(\pi_{UR}Y)_{u,r} = 1$ s'il existe $t \in T$ avec $(u,t,r) \in Y$ sinon la valeur sera 0 et $\pi_{UT}Y \in \{0,1\}^{|U|\times|T|}$ avec $(\pi_{UT}Y)_{u,t} = 1$ s'il existe $r \in R$ avec $(u,t,r) \in Y$ sinon 0 (voir figure 2.1). Ces projections conservent les informations sur les utilisateurs et conduisent à un système de recommandation fondé sur la présence ou l'absence des occurrences entre les utilisateurs respectivement, avec les ressources ou les tags. Notons qu'il y a deux possibilités pour calculer les k plus proches voisins N_k^u d'un utilisateur u, en tenant compte soit des ressources ou bien des tags comme des objets. Après avoir défini la matrice X et choisir parmi ces deux projections $\pi_{UR}Y$ et $\pi_{UT}Y$, [Marinho *et al.* 2011] obtiennent la configuration requise pour pouvoir appliquer le filtrage collaboratif standard. Tout d'abord, ils calculent l'ensemble N_k^u (voir l'équation 2.3) des k utilisateurs les plus similaires à l'utilisateur u.

$$N_k^u := \arg \max_{v \in U \setminus \{u\}}^{k} sim(X_u, X_v) \tag{2.3}$$

où la fonction argmax désigne le nombre $k \in \mathbb{N}$ des voisins à retourner et sim est une mesure de similarité telle que la mesure de cosinus. [Parra et Brusilovsky 2009] proposent

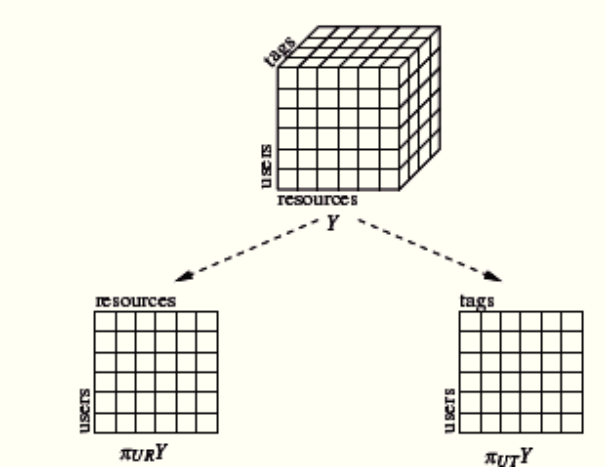

FIGURE 2.1 – La projection de Y en deux matrices utilisateur-ressource et tag-utilisateur. source[Marinho *et al.* 2011].

une approche basée sur l'algorithme de voisinage pondéré *"neighbor-weighted collaborative filtering"* qui présente une extension de modèle classique de filtrage collaboratif [Schafer *et al.* 2007]. D'abord, cette approche commence par déterminer l'ensemble des voisins $\mathcal{N}_u$ pour proposer des recommandations à l'utilisateur u, avec $\mathcal{N}_u$ comprend tous les utilisateurs qui partagent au moins un article ou des tags en communs avec l'utilisateur u. Par la suite, la similarité entre les utilisateurs notée $Sim\,(u, v)$ est calculée comme suit:

$$Sim(u,v) = \frac{\sum_{i \subset CR_{u,v}} (r_{ui} - \overline{r_u})\,(r_{ni} - \overline{r_n})}{\sqrt{\sum_{i \subset CR_{u,v}} (rui - \overline{ru})^2}\sqrt{\sum_{i \subset CR_{u,n}} (r_{ni} - \overline{r_n})^2}} \tag{2.4}$$

Cette formule présente la corrélation de Pearson [Asuero *et al.* 2006] entre un utilisateur u et son voisin v, avec $CR_{u,v}$ désigne l'ensemble des items co-évalués par les utilisateurs u et v. Une fois la similarité calculée, les dix utilisateurs les plus similaires sont sélectionnés. Par la suite, les items appartenant à ces derniers seront classés en utilisant une mesure de pertinence, afin d'être recommandés à l'utilisateur u. La formule de prédiction utilisée est présentée dans 2.5.

$$pred(u,i) = \bar{r}_u + \frac{\sum\limits_{n \in \mathcal{N}_u} Sim(u,n) \times (r_{ni} - \bar{r}_n)}{\sum\limits_{n \in \mathcal{N}_u} Sim(u,n)} \tag{2.5}$$

Cette mesure est déterminée en comparant la totalité des notes données aux items par l'utilisateur cible et de ses voisins.

Une idée similaire a été proposée également dans [Tso-Sutter *et al.* 2008]. Les auteurs ont intégré l'utilisation des informations sur les tags pour améliorer la recommandation. En effet, les auteurs ont transformé la relation ternaire entre les utilisateurs, les items et les tags en trois relations binaires: utilisateur-item, utilisateur-tag et tag-item (voir figure 2.2). Le but de cette projection est de permettre l'intégration des tags dans un filtrage classique. La matrice R présentée dans la figure 2.2 est dérivée du graphe tripartite et elle permet de visualiser les nouveaux items et par qui ils ont été ajoutés. L'agrégation de la dimension tag notée D produit une matrice utilisateur-item, contenant un ensemble de tags pour chaque paire utilisateur-item. En normalisant les valeurs d'utilisateur-item binairement, les auteurs obtiennent la matrice R. Notons que de cette façon, le profil de l'utilisateur est automatiquement enrichi avec les tags. Un algorithme de fusion est ensuite proposé pour combiner les prédictions issues du filtrage collaboratif basé sur l'utilisateur (FCU) et le filtrage collaboratif basé sur les items (FCI) à partir de la matrice étendue. Dans le cas de FCU, la matrice utilisateur-item a été enrichie par la matrice utilisateur-tag. Alors que, dans le cas de FCI, la matrice item-utilisateur a été étendue par la matrice tag-item. [Bogers et Van den Bosch 2008] s'appuient à la fois sur le filtrage collaboratif standard des utilisateurs et des items dans le bookmarking social de partage des documents scientifiques Citeulike [1] pour la recommandation des documents. En conclusion, ils ont trouvé que la recommandation dans Citeulike basé sur les utilisateurs est plus pertinente que celle basée sur les items.

[Mishne 2006] propose le système AutoTag qui est un outil permettant de recommander des tags aux utilisateurs afin d'annoter les messages dans les blogs à l'aide des méthodes de filtrage collaboratif. Plus précisément, les tags sont suggérés dans le but d'annoter les nouveaux messages postés sur weblog à travers les tags associés aux messages similaires. En AutoTag, les messages de weblog prennent eux-mêmes le rôle des utilisateurs. De la même façon que dans la plupart des systèmes de recommandation, AutoTag se base sur l'hypothèse que les utilisateurs sont similaires du moment où ils achètent des produits

1. http://www.citeulike.org/

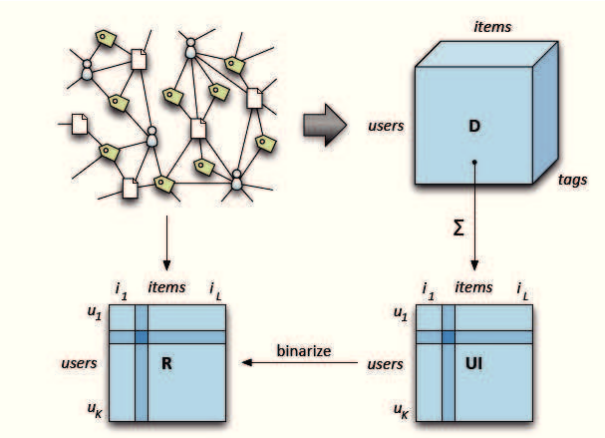

FIGURE 2.2 – Représentation du graphe tripartite d'une folksonomie en matrice 3D. Source: [Tso-Sutter *et al.* 2008]

semblables. En effet, le processus de recommandation est réalisé en trois étapes:

1. Le calcul des messages similaires: AutoTag utilise la technique de recherche d'information (RI) afin d'estimer la similarité entre les messages de weblog. Le moteur de RI permet d'indexer la collection de messages ainsi que de générer la requête à partir de message initial. Les messages les plus similaires sont considérés comme les plus pertinents. Dans les expérimentations, Mishne a testé la génération des requêtes de différentes manières. Au départ, il considère le texte entier en tant que requête. Par la suite il utilise les liens présents dans le texte pour repérer les co-citations. Le meilleur résultat a été obtenu à travers les termes les plus distinctifs permettant de décrire la requête.

2. Le calcul des tags: pour ordonner la liste des tags, AutoTag utilise un classement très simple basé sur la fréquence des tags dans les premiers résultats.

3. Filtrage et ordonnancement: la seule source d'information sur un blogueur est l'ensemble des tags qu'il a précédemment utilisé pour annoter ses messages. Par conséquent, si l'un de ses tags utilisés apparaît dans la liste de classement, AutoTag stimule le score par un facteur constant.

L'évaluation d'AutoTag sur une grande collection de messages montre une bonne précision. En effet, il permet de simplifier le processus de tagging en améliorant sa qualité. Les 10 premiers tags dans la liste seront proposés à l'utilisateur qui choisira par la suite les

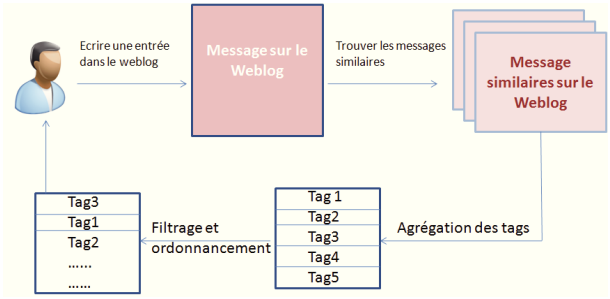

FIGURE 2.3 – Processus de circulation de l'information dans AutoTag

tags les plus convenables pour annoter son message. Dans cette même catégorie, l'approche TagAssist [Sood *et al.* 2007] propose une extension de l'approche AutoTag en rajoutant une étape de prétraitements afin d'améliorer la qualité des recommandations. En effet, cette étape est composée de deux sous étapes principales: la première étape, appelée la phase de normalisation des mots clés, consiste à réduire la forme de chaque mot dans le système à sa forme racine. La deuxième étape, dite la phase de validation, vise à valider chaque groupe issu de la phase de normalisation afin de s'assurer que le système n'a pas regroupé des tags avec des sens différents sous la même racine normalisée. Le système utilise un ensemble des paramètres pour évaluer la qualité de la recommandation de tags. Le cœur de cette approche est fondé sur un moteur de suggestion de tags qui s'appuie sur les tags avec lesquels les anciens messages sont annotés pour pouvoir recommander des tags appropriés pour les nouveaux contenus.

2.2.2 Les approches basées sur le contenu

Ces algorithmes consistent à analyser le contenu ou les métadonnées de ces ressources afin de déterminer quels sont les tags qui peuvent intéresser les utilisateurs. Le principe est très similaire aux techniques utilisées dans le domaine de recherche d'information. La différence réside notamment dans l'absence de requête utilisateur explicite. Pour recommander des tags, deux éléments doivent être constitués: le profil de l'utilisateur et les profils des items. Ces deux éléments sont présentés par des listes de tags pondérés. La manière la plus simple pour définir le profil utilisateur u_m est un vecteur $u_m = (u_{m,1}, ..., u_{m,l})$ avec $u_{m,l} = |\{(u_m, t_l, i) \in A | i \in I\}|$ est le nombre de fois que l'utilisateur a annoté des items par le tag t_l. Parallèlement, le profil des items i_n est défini par $i_n = (i_{n,1}, ..., i_{n,l})$ avec

$i_{n,l} = |\{(u, t_l, i_n) \in A | u \in U\}|$ qui présente le nombre des items annotés par le tag t_l. Afin de mieux comprendre ces deux profils, nous introduisons dans le tableau (2.1) la définition de différents modèles utilisés à la fois pour définir les profils utilisateurs et les profils des items utilisés dans les systèmes de recommandation. En général, les utilisateurs annotent

Éléments	Description
Profil utilisateur basé sur la fréquence des tags	$tf_{u_m}(t_l)$ nombre de fois que l'utilisateur u_m a annoté des items avec le tag t_l
Profil item basé sur la fréquence des tags	$tf_{i_n}(t_l)$ nombre de fois où l'item i_n a été annoté par le tag t_l
Profil utilisateur basé sur la fréquence inverse des tags	$iuf(t_l) = \log \frac{M}{n_u(t_l)'}$ avec $n_u(t_l) = \|\{n_u(t_l) \in \mathbb{U} \| u_{m,l} > 0\}\|$
Profil item basé sur la fréquence inverse de tags	$iif(t_l) = \log \frac{N}{n_i(t_l)'}$ avec $n_i(t_l) = \|\{i_n \in \mathbb{I} \| i_{n,l} > 0\}\|$
Taille de profil utilisateur	$\|u_m\| = \sum_{l=1}^{L} u_{m,l}$
Taille de profil des items	$\|i_n\| = \sum_{l=1}^{L} i_{n,l}$

TABLE 2.1 – Exemple des profils utilisateurs et des items

des items qui leur semblent pertinents. Donc, les tags utilisés pour l'annotation décrivent mieux leurs intérêts, leurs goûts et leurs besoins. Par ailleurs, ce type d'approche peut également supposer que plus un tag est utilisé par un utilisateur, plus ce tag est considéré très important pour lui. Par analogie, les tags affectés aux items expriment un point de vue de la communauté des utilisateurs qui annotent la ressource. Plus des utilisateurs annotent un item par un tag particulier, plus ce tag décrit mieux son contenu. La représentation de contenu est ensuite comparée au profil utilisateur afin de trouver les items les plus pertinents pour cet utilisateur. Comme le filtrage collaboratif, la représentation des profils utilisateurs est fondée sur des modèles à long terme où la mise à jour est effectuée automatiquement à chaque apparition des nouvelles informations.

Selon [Xu *et al.* 2008], les hypothèses présentées précédemment doivent être soigneusement prises en compte parce que les tags utilisés très souvent par les utilisateurs pour annoter de nombreux items ne peuvent pas être utiles pour découvrir leurs préférences et les caractéristiques des items. [Adomavicius et Tuzhilin 2005] ont présenté une définition plus

formelle de la recommandation basée sur le contenu (voir l'équation 1.1) avec:

$$g(u_m, i_n) = sim\left(ProfilUtilisateur(u_m), Contenu\left(i_n\right)\right) \in \mathcal{R} \tag{2.6}$$

où $ProfilUtilisateur(u_m) = (u_{m,1}, ..., u_{m,k}) \in \mathcal{R}_k$ présente les préférences c'est-à-dire les appréciations portées sur les contenus déjà consultés par l'utilisateur u_m et $Contenu\left(i_n\right) = i_n = (i_{n,1}, ..., i_{n,k}) \in \mathcal{R}_k$ correspond à l'ensemble des contenus qui caractérise l'item i_n. Ces caractéristiques sont généralement représentées par des vecteurs composés de nombres réels (poids) dans lequels chaque composante mesure "l'importance" de l'élément correspondant dans les représentations de l'utilisateur et de l'item. La fonction *sim* mesure la similarité entre le profil utilisateur et le profil des items dans l'espace des attributs de contenu. Parmi les premiers travaux de recommandation basés sur le contenu, nous citons l'approche présentée dans [Lee et Chun 2007]. Ce système recommande des tags extraits à partir des blogs en utilisant un réseau de neurones artificiel. Ce réseau est formé par des données statiques, en particulier la fréquence des mots (voir 2.7) dans les blogs, conjointement avec des informations lexicales concernant la sémantique du mot extraite de WordNet.

$$\mathcal{T}F/IDF(mot) = TermFreq(mot) \times \log\left(\frac{|Corpus|}{Docfreq(mot)}\right) \tag{2.7}$$

avec:

- $\mathcal{T}ermFreq(mot)$ désigne le nombre d'occurrences de ce mot dans le blog. Elle se calcule comme suit:

$$\mathcal{T}ermFreq(mot) = \frac{n_i}{\sum_k n_k} \tag{2.8}$$

- $|Corpus|$ indique le nombre total de documents pour chacun des utilisateurs.
- $Docfreq(mot)$ indique la fréquence d'un mot dans un corpus.

[Song *et al.* 2008a] considèrent la tâche de recommandation de tags comme un problème de classification *multi label*. Le modèle Gaussian est un processus stochastique basé sur une collection des variables aléatoires x, qui forme une distribution Gaussian multivariée par une fonction $\mu(x)$ et une fonction de covariance $k(x, x')$. Ce processus est utilisé afin de classer le contenu des ressources web (le titre et les descriptions de taille réduite). Chaque classe correspond à un sujet, il est représenté sous la forme d'un profil composé de tags. Par la suite, les tags provenant de différents profils sont combinés pour créer la recommandation

finale. Contrairement aux systèmes de recommandation de tags basés sur les graphes, ces méthodes ne sont pas limitées uniquement aux ressources. Toutefois, leur pratique est toujours en question puisqu'elles reposent sur des algorithmes d'apprentissage automatique de calcul intensif . En outre, ces méthodes ne sont pas en mesure de réutiliser les tags qui sont déjà présents dans le système de tagging, donc des modèles de classification peuvent être construits pour résoudre ce problème. D'autre part, l'avantage principal de cette approche est sa généralité. En effet, la plupart des systèmes extraient l'ensemble d'information seulement à partir de contenu textuel des ressources, alors que ces méthodes ne sont pas limitées en particulier à ce type de contenu. Par exemple, [Weston *et al.* 2010] ont proposé une méthode de tagging des images basée sur des caractéristiques visuelles appelée Visterms. Parmi les travaux qui se basent sur les techniques de recherche d'information, nous citons l'application STAR (Social Tag Recommender) [Musto *et al.* 2010] qui est un système de recommandation de tags personnalisés. Le but principal de STAR est d'améliorer les modèles implémentés dans AutoTags [Mishne 2006] et TagAssist [Sood *et al.* 2007]. Selon [Musto *et al.* 2010], l'approche de Mishne présente deux inconvénients:

- la formule utilisée pour ordonnancer les tags consiste à additionner les occurrences de chaque tag dans toute la folksonomie, sans prendre en compte la similarité avec la ressource à annoter.
- le modèle présenté ne prend pas en considération les anciennes activités de tagging de l'utilisateur. En effet, si deux utilisateurs annotent la même ressource, ils vont avoir la même recommandation.

Pour faire face à ces limites, [Musto *et al.* 2010] proposent une approche basée sur l'analyse des ressources similaires capable également de prendre en compte les tags déjà sélectionnés par l'utilisateur pendant son activité précédente de tagging, en les mettant en tête de classement de tags. STAR est fondé principalement sur l'exploitation du modèle BM25 [2] [Baeza-Yates *et al.* 1999] et se repose sur deux hypothèses:

- Si deux ou plusieurs ressources partagent certaines caractéristiques en commun (par exemple la même description textuelle), nous pouvons exploiter cette information en

2. BM25 est une mesure de calcul de pertinence utilisée par les moteurs de recherche pour classer des documents en fonction de leur pertinence par rapport à une requête donnée.

supposant qu'elles pourraient être annotées avec des tags similaires.

– Étant donné que chaque utilisateur a une manière typique pour annoter ses ressources, un système de recommandation doit exploiter ces informations pour pondérer l'importance des tags par rapport à l'utilisateur ayant déjà utilisé ces tags pour annoter des ressources similaires.

[Graham et Caverlee 2008] utilisent la combinaison de moteur de recherche de texte avec le modèle de rétroaction (ou feedback). Les tags provenant des ressources liées sont combinés via le modèle pondéré de plus proche voisin. Le processus répétitif de rétroaction permet à l'utilisateur d'améliorer itérativement la qualité des tags récupérés. En outre, ces méthodes souffrent du même problème que la sous catégorie précédente des systèmes de recommandation basés sur le contenu: le vocabulaire limité des tags. Pour être recommandé, le tag doit apparaître dans un grand nombre de documents pertinents. Donc, seulement les tags fréquents sont susceptibles d'être recommandés.

[Chirita *et al.* 2007] proposent un système de recommandation de tags permettant d'extraire les tags à partir du contenu des pages web. À part les scores de base utilisés pour estimer l'utilité des mots d'un site en tant que tags (en utilisant par exemple la fréquence des termes présentée dans 2.7), le système exploite aussi le contenu du site et les mots-clés présents dans le répertoire personnel des documents de l'utilisateur. Les tags rajoutés sont extraits des documents personnels connexes. De cette façon, le système est capable d'avoir accès à des tags supplémentaires qu'il ne peut pas trouver dans le contenu de site. Ces tags représentent des perspectives personnelles d'un utilisateur. [Medelyan *et al.* 2009] présentent un système de tagging *Maui*, purement basé sur des mots clés que l'on retrouve dans le contenu des ressources. Maui est une extension du système Kea [Frank *et al.* 1999]. Le système utilise un algorithme de classification binaire pour chaque mot ou expression du contenu de la ressource. L'ensemble des caractéristiques inclut la fréquence des termes, la distance du début du document, la longueur des mots en fonction de la mesure d'occurrence de mots dans le corpus de Wikipédia. En général, les méthodes d'extraction de mots clés ont un accès direct au contenu de la ressource. Par conséquent, elles sont plus adaptées à extraire des tags avec une grande précision. En outre, ils ne reposent pas sur des tags qui ont été fréquemment utilisés dans le système de tagging. D'autre part, ils sont limités par

le vocabulaire des ressources, susceptible d'être biaisé par l'auteur de la ressource.

2.2.3 Les approches topologiques

Dans la littérature des systèmes de recommandation, une folksonomie peut être vue comme un hypergraphe. En effet, les hypergraphes ne lient pas seulement un ou deux sommets, mais un nombre quelconque de sommets. Dans notre contexte, l'hypergraphe de notre folksonomie est représenté par un graphe tripartite composé par des nœuds qui correspondent aux éléments de la folksonomie: les utilisateurs, les ressources et les tags. Les arêtes entre ces trois types de nœuds reflètent le comportement de tagging des différents utilisateurs de la folksonomie.

Définition 4 *Représentation d'une folksonomie en graphe tripartite: une folksonomie* $F = (U, T, R, Y)$ *est un graphe tripartite (non orienté) noté* $G := (V, E)$ *avec:*

1. $V := U \cup T \cup R$ *est l'ensemble des nœuds de tags, des utilisateurs et des ressources.*
2. *Tous les liens entre les tags, les utilisateurs et les ressources deviennent des arêtes non orientées et non pondérées entre les nœuds respectivement:* $E = \{\{u,t\},\{t,r\},\{u,r\} \mid (u,t,r) \in Y\}$ *avec chaque arête* $\{u,t\}$ *pondérée par* $|\{r \in R : (u,t,r) \in Y\}|$, *chaque arête* $\{t,r\}$ *par* $|\{u \in U : (u,t,r) \in Y\}|$ *et chaque arête* $\{u,r\}$ *par* $|\{t \in T : (u,t,r) \in Y\}|$.

Beaucoup de systèmes de recommandation de tags adoptent cette représentation. C'est un choix justifié puisque ces approches ne dépendent ni de contexte ni de sémantique, mais elles s'intéressent plutôt aux interactions entre les nœuds du graphe. Par exemple, le système proposé dans [Rae *et al.* 2010] utilise les co-occurrences des tags pour suggérer aux utilisateurs des tags afin de compléter le tagging des photos dans Flickr. [Jäschke *et al.* 2008] proposent deux approches de recommandation de tags pour le bookmarking social appelées *l'adaptation de Pagerank* et *le Folkrank*. Ces algorithmes sont inspirés de *Pagerank* [Brin et Page 1998]. Dans cette section, nous proposons une classification des différentes approches topologiques en trois catégories. La première catégorie regroupe les approches basées sur le tri des nœuds tels que l'adaptation de Pagerank dans les folksonomies, l'algorithme de Folkrank [Hotho *et al.* 2006] et Folkdiffusion [Zhang *et al.* 2011] qui

représente une extension de Folkrank. La deuxième catégorie englobe les approches dynamiques, comme le système Liptar [Pujari et Kanawati 2012] qui emploie des techniques de prévision des liens dans les folksonomies. À la fin de cette section, nous introduisons la troisième catégorie des approches topologiques qui sont basées sur le clustering.

Adaptation de Pagerank. Pagerank est un algorithme de classement des pages web. L'idée de base est qu'une page web est importante s'il y a beaucoup de pages importantes qui pointent vers elle. [Hotho *et al.* 2006] ont utilisé le même principe pour le classement de nœuds dans une folksonomie. Afin d'appliquer l'algorithme Pagerank, il faut convertir la folksonomie $F = (U, T, R, Y)$ en un graphe tripartite non orienté $G = (V, E)$ (présenté dans la définition 4). Comme dans Pagerank, Hotho utilise la marche aléatoire *(random walk)*. Il est basé sur l'idée qu'une marche aléatoire idéale suit normalement les liens (par exemple, à partir d'une page ressource à un tag ou à une page utilisateur), mais se déplace de temps en temps vers un nouveau nœud sans suivre le lien. Le classement des nœuds du graphe est calculé (comme dans le Pagerank) en calculant le poids de propagation suivant:

$$w_{t+1} \leftarrow dA^T w_t + (1-d)\vec{p} \tag{2.9}$$

où w_t est un vecteur de poids avec une seule entrée pour chaque nœud appartenant à V, A est la matrice d'adjacence du graphe G, $\vec{p}$ est un vecteur utilisé pour désigner la préférence utilisateur. $d \in [0, 1]$ est une constante qui détermine l'influence de vecteur préférence p. Les résultats de l'adaptation de Pagerank sur les données de Delicious montrent que le vecteur de préférence n'a pas un poids très important pour surmonter la structure globale du graphe. Afin d'intégrer les intérêts utilisateur définis dans le vecteur préférence, [Hotho *et al.* 2006] ont développé l'algorithme de Folkrank qui calcule le différentiel entre le résultat de classement avec et sans le vecteur de préférence.

Folkrank. L'idée clé de Folkrank consiste à dire qu'une ressource annotée par des tags importants et par des utilisateurs importants devient elle même importante. Idem pour les tags et les utilisateurs. En effet, chaque nœud du graphe propage son importance à ses voisins. Cette diffusion permet d'obtenir comme résultat un graphe composé par des sommets qui sont mutuellement renforcés. Cet algorithme calcule le classement à un centre

d'intérêt bien spécifique comme suit:

1. Un vecteur de préférence $\vec{p}$ est utilisé pour indiquer les intérêts de l'utilisateur. Généralement, une valeur de poids élevée est associée à une seule ou à un petit ensemble d'entré et le reste a le même poids vu que la structure de la folksonomie est symétrique. En plus, ils ont la possibilité de définir la préférence utilisateur en attribuant un poids plus élevé soit à un ou plusieurs tags et/ou un ou plusieurs utilisateurs et/ou une ou plusieurs ressources.
2. Soit $\vec{w_0}$ le résultat d'application de Pagerank avec $d = 1$ (sans la prise en compte du vecteur préférence de l'utilisateur).
3. Soit $\vec{w_1}$ le résultat d'application de Pagerank avec $d < 1$ (avec la prise en compte du vecteur préférence de l'utilisateur avec un poids $d \in [0, 1]$).
4. $\vec{w} = \vec{w_1} - \vec{w_0}$ est le vecteur de poids final.

Donc, ils calculent les gagnants et les perdants de renforcement mutuel des ressources en comparant les deux vecteurs dans le cas de présence et d'absence de préférence utilisateur. Le poids résultant de ce différentiel noté $w[x]$ d'un élément x de la folksonomie est appelé indice de Folkrank de x. L'adaptation de Pagerank dans les folksonomies fournit un seul classement global, indépendamment de toutes les préférences utilisateur. Tandis que Folkrank produit un classement spécifique sur des sujets bien précis pour chaque vecteur de préférence. Notons qu'un sujet peut être défini dans le vecteur de préférence non seulement en attribuant un poids plus élevé à des tags spécifiques, mais aussi à des ressources et des utilisateurs spécifiques. Ces trois dimensions peuvent même être combinées dans un vecteur mixte. De même, le classement n'est pas limité à des ressources, il peut aussi bien être appliqué aux tags et aux utilisateurs. Lors de l'utilisation du vecteur utilisateur afin d'exprimer ses préférences (par exemple, en donnant un poids plus élevé à un tag par rapport aux autres), les tags, les utilisateurs et les ressources qui sont liés à cette préférence sont les mieux classés dans le résultat. Les expérimentations de [Hotho *et al.* 2006] montrent que la recommandation basée sur Folkrank surpasse les approches basées sur l'adaptation de *filtrage collaboratif (utilisateur-tags et utilisateur-ressources)* et sur *les tags les plus populaires basés sur les ressources*. Les tests ont été effectués sur le noyau dense de Delicious, de sorte qu'il n'est potentiellement pas très représentatif. En outre, ils ne prennent pas

en compte la nature dynamique d'une folksonomie. Différents travaux tentaient d'étendre l'algorithme de Folkrank, par exemple l'approche de FolkDiffusion [Liu *et al.* 2010] a été proposée pour éviter la recommandation des tags non pertinents qui sont hors contexte avec la thématique de l'utilisateur. Cette approche s'inspire du phénomène physique de la diffusion du flux de chaleur de la haute à la basse température. Tout d'abord, [Liu *et al.* 2010] commencent par la génération du graphe tripartite composé par les utilisateurs, les ressources et les tags. En se basant sur l'idée de la diffusion, l'utilisateur u et la ressource r de la requête sont attribués à la température la plus élevée. Tandis que tous les autres utilisateurs, ressources et tags reçoivent une valeur de température nulle. Après, la chaleur commence à se diffuser à partir de u et r vers tout le graphe à travers les arêtes. L'arête qui lie deux nœuds peut être vue comme la conduite d'un climatiseur face à la chaleur et le poids des arêtes indique la vitesse de la diffusion de la chaleur. Après plusieurs itérations de diffusion, les valeurs de la chaleur des tags indiquent leurs liaisons avec l'utilisateur cible u et de la ressource cible r et sont donc sélectionnés pour la recommandation.

Liptar. Dans [Pujari et Kanawati 2012], les auteurs proposent une approche de prévision des liens dans les graphes bipartites en appliquant un algorithme d'apprentissage supervisé. Le but de ce système est de calculer une liste de tags la plus adaptée pour annoter une ressource cible r_t par un utilisateur cible u_t. Le cycle de fonctionnement de Liptar est structuré en trois étapes principales:

1. Premièrement, le système détermine l'ensemble de k utilisateurs les plus similaires à u_t. Plusieurs métriques peuvent être utilisées pour calculer cette similarité. Dans ce travail, les auteurs ont utilisé la méthode de k plus proches voisins avec une mesure de similarité de l'utilisateur u basée à la fois sur les ressources et les tags. Un autre aspect important dans ce système est qu'il prend en considération le temps d'activité de l'utilisateur. De ce fait, les k utilisateurs trouvés ont au moins un an d'activité en commun avec l'utilisateur cible u_t. Ici, les auteurs explorent l'idée que les utilisateurs actifs au cours de cette période de temps peuvent avoir des intérêts et des choix communs.

2. Chaque utilisateur $u \in U_s$ est associé à une séquence temporelle des graphes bipartites

reliant les ressources ajoutées par l'utilisateur u aux tags qui les a utilisés dans de différents intervalles de temps. Ces graphes sont combinés pour donner un seul graphe bipartite ressource-tag qui servira à l'apprentissage (G_{learn}). Au cours de ce processus, seulement les graphes dont la date est incluse dans la période d'apprentissage seront utilisés. La génération de ce graphe est traduite par la formule suivante:

$$G_{learn} = \bigcup_{u \in U_s} \bigcup_{i=t_0}^{t_{learn}} G_i \tag{2.10}$$

3. Une ou plusieurs listes de classement de tags sont obtenues pour la ressource r_t et/ou pour les ressources similaires en utilisant des données relatives à déterminer les utilisateurs similaires. Ces listes incluent à la fois les tags déjà utilisés ainsi que les tags à recommander. Pour les fusionner, [Pujari et Kanawati 2012] appliquent une approche appropriée pour l'agrégation des listes de classement [Dwork *et al.* 2001].

Approches basées sur le Clustering. Le clustering des données est une technique d'analyse des données. Le clustering fournit un partitionnement d'un ensemble de données en sous-ensembles d'objets similaires ou des groupes (clusters) de données et qui sont dissimilaires aux autres en fonction d'un critère bien déterminé [Begelman *et al.* 2006]. Le principe de clustering consiste à regrouper deux ou plusieurs objets dans le même groupe s'ils sont *proches* selon une distance donnée. Cette distance peut être calculée par exemple en utilisant la fonction de plus court chemin. Dans le contexte de clustering de tags, une distance entre les tags issue d'un système social de tagging est définie. Ce clustering permet de regrouper les tags de manière à ce que les tags d'une même classe ont été toujours utilisés ensemble pour annoter des ressources. En effet, le clustering permet d'avoir une organisation des rubriques dans un système de tagging afin d'améliorer la phase de recherche des données aux utilisateurs [Brooks et Montanez 2006]. Pareillement, [Begelman *et al.* 2006] s'aperçoivent que l'utilisation des informations brutes des tags freine la découverte et l'exploration de contenu. D'où la nécessité d'un niveau supplémentaire d'organisation à travers le clustering de tags. Dans [Gemmell *et al.* 2008], les clusters de tags sont vus comme des liens entre les utilisateurs et leurs intérêts. L'utilisation de ces clusters au lieu de tags classiques sert à construire le profil utilisateur afin d'améliorer la

recommandation et de réduire la complexité du graphe de la folksonomie. Ils se sont avérés bénéfiques pour le classement de contenu personnalisé. Une autre approche de clustering de tags a été présentée dans [Au Yeung *et al.* 2009]. Dans ce travail, les clusters de tags ont été utilisés comme un moyen pour identifier les différents contextes d'utilisation d'un tag donné, c'est-à-dire pour résoudre le problème d'ambiguïté. Ils ont montré que l'intégration des clusters de tags a amélioré les résultats par rapport à l'utilisation de ressources externes telle que WordNet. Indépendamment de la méthode utilisée pour mesurer la similarité entre les tags, les clusters de tags doivent correspondre à des thématiques ayant une cohérence sémantique, ce qui peut être utile dans plusieurs applications, telle que l'exploration des informations et de la navigation ([Begelman *et al.* 2006], [Gruber 2007]), l'annotation automatique de contenu [Brooks et Montanez 2006], la construction de profil utilisateur [Gemmell *et al.* 2008], le clustering de contenu ([Giannakidou *et al.* 2008], [Java *et al.* 2008]) et la recommandation de tags ([Sigurbjörnsson et Van Zwol 2008], [Li *et al.* 2009]). Les méthodes de clustering de tags relèvent en grande partie à l'une des deux catégories suivantes:

- Soit à travers les techniques de clustering classiques tels que l'algorithme de k-means [Giannakidou *et al.* 2008] et le clustering ascendant hiérarchique (CAH) ([Brooks et Montanez 2006], [Shepitsen *et al.* 2008])
- Soit par l'application de clustering sur les graphes connu sous le nom *des méthodes de détection des communautés* que nous allons expliquer en détail dans le chapitre suivant 2.1 ([Begelman *et al.* 2006], [Brooks et Montanez 2006], [Papadopoulos *et al.* 2009]).

Dans la première catégorie des techniques de clustering, nous présentons le système de [Shepitsen *et al.* 2008]. Les auteurs proposent un algorithme de personnalisation pour la recommandation des ressources dans les folksonomies en s'appuyant sur le clustering hiérarchique des tags. Le processus de recommandation se déroule en deux étapes. Premièrement, étant donné un clic d'utilisateur sur un tag, l'algorithme de recommandation standard non personnalisé est appliqué afin de retourner comme recommandation un ensemble de ressources. Cet ensemble est ensuite personnalisé grâce à la prise en compte du profil utilisateur et du clustering de tags. Par conséquent, un nouveau classement des

résultats est retourné à l'utilisateur. En plus, ils supposent l'existence d'un ensemble de clusters de tags obtenu à la phase de clustering en mode déconnecté.

Par ailleurs, le processus de recommandation personnalisé passe par les trois étapes suivantes:

- **Étape 1:** calculer la similarité en utilisant la mesure de *cosinus*. Une recherche de base est effectuée sur la base de la requête q, en utilisant la métrique de similarité dans l'équation 2.11. Une similarité $S(q,r)$, est calculée pour chaque ressource $r \in R$.

$$cos(q,r) = \frac{tf(q,r)}{\sqrt{\sum_{t \in T} tf(t,r)^2}} \tag{2.11}$$

  En sortie, cette étape de l'algorithme produira un sous-ensemble des ressources $R^{'}$, ayant une certaine similitude avec la requête de tag.
- **Étape 2:** calculer la pertinence de tout $r \in R^{'}$ à u. Dans cette étape, les clusters présentent des liens entre les utilisateurs et les ressources permettant d'identifier les ressources qui reflètent les intérêts utilisateur. L'étape 2 se fait en trois sous étapes:
  - calculer l'intérêt de l'utilisateur dans chaque cluster. Pour chaque cluster c, l'intérêt de l'utilisateur est calculé comme le ratio de nombre de fois que l'utilisateur u a annoté une ressource par un tag du cluster c sur le nombre total des annotations de u.
  - calculer les plus proches clusters de chaque ressource. La relation d'une ressource r à un cluster c est calculée par le ratio du nombre des ressources qui ont été annotées par un tag de cluster c sur le nombre total des annotations de la ressource r.
  - Déterminer les intérêts utilisateurs dans chaque ressource. La pertinence d'une ressource r à un utilisateur $P(u,r)$ est la somme des produits de leurs poids dans l'ensemble de tous les clusters.
- **Étape 3:** calculer le classement final des scores personnalisés. À cette étape, les auteurs combinent la mesure de similarité *cosinus* trouvée à l'étape 1 avec la pertinence mesurée dans l'étape 2. La similitude personnalisée est calculée pour chaque ressource en multipliant la similarité *cosinus* par la pertinence d'une ressource à l'utilisateur. Cette similitude notée $S^{'}(u,q,r)$ est définie par:

$$S^{'}(u,q,r) = S(q,r) * P(u,r) \tag{2.12}$$

Une fois que $S'(u,q,r)$ est calculé pour chaque ressource et après avoir classé les ressources, les n meilleures ressources sont retournées à l'utilisateur en tant que résultat de recommandation.

Concernant le module clustering, les auteurs utilisent l'algorithme CAH. Cet algorithme prend comme entrée un ensemble de tags T et le coefficient de division. Les tags sont représentés par un vecteur de poids de l'ensemble des ressources. CAH souffre d'une complexité très élevée (second degré pour le nombre de tags à regrouper) et la nécessité de définir les paramètres adhoc (par exemple, trois paramètres doivent être définis dans le schéma de clustering utilisé dans [Gemmell *et al.* 2008]). Les approches de détection des communautés tentent de combler les lacunes du CAH à travers les implémentations efficaces avec une complexité de $O(Nlog(N))$ pour trouver le clustering optimal des N tags en des communautés. En outre, les méthodes de détection des communautés reposent sur la mesure de modularité [Newman et Girvan 2004] en tant que moyen pour évaluer la qualité de clustering. Ainsi, *les méthodes de maximisation de la modularité* ne nécessitent pas l'intervention de l'utilisateur pour définir les paramètres. Cependant, le principal problème rencontré par ces méthodes est leur tendance à produire des clusters avec une distribution de taille très inégale. Ce qui les rend inadaptés pour gérer la problématique de clustering de tags.

2.2.4 Les approches hybrides

Les systèmes de recommandation de tags hybrides tentent souvent de combiner les avantages de contenu des ressources et les graphes de la folksonomie. D'habitude, ils commencent par le traitement de contenu des ressources, c'est pour cela qu'ils sont souvent classés comme des approches basées sur le contenu. Les approches topologiques et celles basées sur le contenu adaptent généralement des techniques d'apprentissage automatique ou de recherche d'information pour faire face au problème de recommandation de tags. En effet, les systèmes hybrides tentent de combiner plusieurs sources d'informations dans les folksonomies. Ce type d'approche permet d'être plus efficace et de traiter une plus grande variété de publications, ce qui la rend plus pratique. Parmi les systèmes hybrides il y a le travail de [Tatu *et al.* 2008] qui ont proposé un système basé sur l'extraction de tags à partir des ressources et du profil utilisateur. L'ensemble des tags est étendu en utilisant

des techniques de NLP [3]. Plus tard, ils sont fusionnés avec le contenu des tags. Un autre système introduit par [Ju et Hwang 2009] consiste à analyser le contenu des documents déjà annotés, afin d'évaluer la vraisemblance d'utilisation d'un mot de contenu en tant que tag. Par la suite, cette vraisemblance est utilisée comme un score afin de pondérer les mots qui se produisent dans le contenu de la ressource publiée. Le contenu basé sur les tags est linéairement combiné avec des tags issus des deux profils utilisateurs et ressources. Le système de [Musto *et al.* 2009] a été fondé sur un moteur de recherche. En effet, le système récupère les ressources dont le contenu textuel est lié au titre de la ressource publiée, et construit une recommandation basée sur les tags pertinents à partir de leurs profils. Une importance particulière est donnée aux ressources publiées précédemment par l'auteur de la publication actuelle. De ce fait, leurs tags obtiennent un poids plus important lorsque les tags issus de toutes les ressources pertinentes sont combinés. Une perspective intéressante d'hybridation a été introduite dans l'approche de [Gemmell *et al.* 2010]. Contrairement aux autres approches, leur système est basé uniquement sur les informations extraites du graphe de la folksonomie et n'utilise pas le contenu des ressources. Le système est fondé sur six simples modèles de recommandation, qui comprennent les tags les plus fréquents à partir des ressources et du profil utilisateur. En plus, ils utilisent quatre méthodes de filtrage collaboratif avec différentes méthodes de calcul de similarité entre les utilisateurs et les ressources. Les auteurs ont évalué la performance des approches hybrides, ainsi que leurs composants. Les résultats montrent une grande différence au niveau du comportement de tagging dans diverses collections de données. Il est important de mentionner que les auteurs extraient les p-noyaux de chaque ensemble de données afin de se concentrer sur le noyau dense du graphe de la folksonomie. Les p-noyaux peuvent être extraits seulement pour *les folksonomies larges (broad)* même s'ils ne contiennent qu'une petite fraction de toutes les publications en entrée dans le système.

2.3 Analyse critique

Dans cette section, nous discutons les principaux avantages et inconvénients des algorithmes présentés dans la section 2.2. Les algorithmes de FC ont plusieurs avantages

3. Natural Language Processing

comme le fait de pouvoir prendre en compte les avis positifs ou négatifs des utilisateurs sur un item au moment de la recommandation, en particulier dans le cas des notes explicites des utilisateurs. Un deuxième avantage est que les algorithmes de FC sont particulièrement utiles dans les domaines où l'analyse de contenu est difficile ou coûteuse telle que la recommandation de musiques et de films. Nous rappelons que dans la section 2.2.1, nous avons vu que pour pouvoir appliquer les algorithmes de FC standards sur les folksonomies, une transformation des données doit être effectuée. De telles transformations conduisent à la perte d'informations, ce qui peut diminuer la qualité de recommandation. Un autre problème bien connu rencontré avec les méthodes de FC est que les grandes matrices de projection doivent être gardées en mémoire, qui peut être une consommation de temps/espace ainsi que de compromettre la capacité d'effectuer des recommandations en temps réel. Un autre problème est identifié dans la phase de démarrage du système de recommandation. Ce cas se produit quand il existe déjà de nombreux items dans le système, mais peu d'utilisateurs et peu d'évaluations. Ceci est connu sous le nom *du problème de démarrage à froid*. Par conséquent, le système de recommandation ne peut pas générer des recommandations [Schein *et al.* 2002]. L'avantage des approches basées sur le contenu (voir section 2.2.2) est d'être capable de recommander des tags même pour les objets qui n'ont pas été préalablement annotés par les utilisateurs. Ceci est clairement bénéfique pour des collections de documents qui sont actuellement peu annotées, tel est le cas dans les entreprises. L'avantage de recommandation de tags basée sur le contenu est qu'elle n'exige pas l'intervention humaine tout au long du processus de tagging.

L'apparition des nouvelles approches topologiques telle que Folkrank a amélioré considérablement la recommandation de tags. Ce type d'approche n'exige pas de savoir la spécificité des ressources et permet aussi le changement de mode (recommandation de ressources/tags) sans modification de l'algorithme. En plus, tout comme les algorithmes basés sur le FC, Folkrank est robuste contre les mises à jour en ligne, car il n'a pas besoin d'être informé chaque fois qu'un nouvel utilisateur, ressource ou un tag entre dans le système. Pour ces raisons, nous adoptons une approche topologique car elle peut s'appliquer sur plusieurs types de folksonomies. En plus, notre approche permet de faire face à la complexité des folksonomies, qui sont en train d'exploser en taille au cours du temps,

tout en proposant une approche de réduction qui utilise un algorithme de détection des communautés. Notre approche est basée sur deux étapes fondamentales:

- Un système de clustering permettant de réduire la taille et la complexité de la folksonomie.
- Une approche de recommandation de tags par niveau afin d'aider l'utilisateur à indexer ses ressources.

2.4 Conclusion

Dans ce chapitre, nous avons présenté les différents types de folksonomies et les problèmes qu'elles rencontrent, en particulier, le problème d'ambiguïté de tags. De ce fait, ce chapitre introduit un état de l'art sur les différents travaux de recommandation de tags qui tentent de faire face à cette problématique. En effet, les approches des systèmes de recommandation de tags sont variées, et peuvent être classées de différentes manières. Dans la section 2.2, nous avons proposé une classification en quatre catégories d'approches: le filtrage collaboratif, la recommandation basée sur le contenu, les approches topologiques et les approches hybrides. À la fin, nous avons fait une étude critique des différents travaux et nous avons exposé quelques exemples des systèmes de recommandation les plus connus comme les applications de recommandation utilisées dans les systèmes de bookmarking social: *Bibsonomy* et *Delicious*. Dans le cadre de cette thèse, nous proposons une approche topologique de recommandation de tags *TLTR* présentée dans le chapitre 5. Notre approche est basée sur le clustering. Elle fait appel *aux algorithmes de détection des communautés* afin d'améliorer la recommandation. Ce qui fait l'objet du chapitre suivant.

Chapitre 3

Algorithmes de détection de communautés

3.1 Introduction

Nous avons présenté dans le chapitre précédent, les différentes approches de recommandation dans le contexte des folksonomies. Les folksonomies peuvent être représentées par des réseaux hétérogènes composés de plusieurs types de nœuds et par des hyperliens. L'approche de recommandation proposée que nous détaillons dans le chapitre 5 s'inscrit dans le cadre des approches topologiques où les folksonomies sont représentées par un graphe tripartite composé par les trois composantes: utilisateurs, tags et ressources. Nous travaillons sur des approches de recommandation à double niveau. Le premier niveau consiste à réduire le graphe de la folksonomie en générant des regroupements des clusters utilisateurs, des tags et des ressources. La génération de ces clusters nous conduit vers le problème de détection des communautés. En effet, le contexte tripartite fournit pour chaque entité deux types d'informations en fonction des deux autres entités. Par exemple, les relations entre les tags peuvent être établies soit en fonction des utilisateurs où les tags utilisés par le même utilisateur sont liés ensemble, soit en fonction des ressources où les tags utilisés pour annoter la même ressource sont liés ensemble. De la même façon, nous déterminons deux points de vue de relations pour l'entité utilisateur et pour l'entité ressource. Afin de profiter de cette richesse d'information et d'exploiter simultanément les relations issues des différents points de vue de modélisation, nous nous tournons vers les réseaux multiplexes. En effet, la notion du réseau multiplexe a été récemment introduite afin de faciliter la modélisation

des réseaux multirelationnels, des réseaux dynamiques ou même des réseaux attribués. Un réseau multiplexe est un graphe multicouche. Concrètement, chaque entité est modélisée par un réseau multiplexe. Chaque couche du réseau représente un type de relation. Par exemple, le réseau multiplexe relatif à l'entité tags est composé de deux couches: la première désigne les relations entre les tags issues d'un point de vue utilisateur, et la deuxième indique les relations entre les tags issues d'un point de vue ressource. Dans ce chapitre, nous présentons une étude approfondie des techniques de détection de communautés dans les graphes multiplexes. Pour cela nous commençons par expliquer dans la suite comment se passe la détection de communautés dans les graphes simples puis dans les graphes multiplexes. Dans le même cadre de recommandation, d'autres travaux ont utilisé les techniques de détection de communautés pour faire face au problème de recommandation de tags [Papadopoulos *et al.* 2011], [Shepitsen *et al.* 2008]. En effet, [Papadopoulos *et al.* 2011] ont appliqué le clustering uniquement sur la composante tags. Du coup, ils n'ont exploité qu'un seul type de relation: les tags issus de point de vue ressource.

Dans la section 3.2, nous passons en revue les principales approches de détection de communautés dans les graphes simples. Par la suite, la section 3.3 présente les principales approches d'extension des approches étudiées dans la section 3.2 dans le cas des graphes multiplexes. Nous mettons l'accent sur les différentes techniques d'évaluation existantes. Dans la section 3.4, nous terminons par la conclusion.

3.2 Détection de communautés dans les graphes simples

Le problème de détection de communautés est très similaire aux problèmes traités dans d'autres domaines, tels que le *clustering* de données, le problème de calcul de cut dans des graphes ou encore les problèmes d'optimisation. Ce qui amène à avoir une grande variété d'approches pour l'identification des communautés. Une communauté est un ensemble de nœuds dont la densité des liens internes est plus forte que la densité des liens externes. Trois études de synthèse intéressantes, mais non exhaustives, sont présentées dans [Tang et Liu 2010], [Fortunato 2010] et [Papadopoulos *et al.* 2012]. Ici, nous proposons de classifier les approches existantes dans quatre classes non exclusives entre elles:

- *Approches centrées groupe* où des nœuds sont regroupés en communautés en fonction des propriétés topologiques partagées.
- *Approches centrées réseau* où la structure globale du réseau est examinée pour la décomposition du graphe en communautés.
- *Approches centrées propagation* qui appliquent souvent une procédure d'émergence de la structure communautaire par échange de messages entre nœuds voisins.
- *Approches centrées graine* où la structure communautaire est construite autour d'un ensemble de nœuds choisis d'une manière informée.

Le tableau 3.1 donne les principales notations utilisées.

TABLE 3.1 – Notations utilisées

Notation	Description
$G =< V, E >$	G: Graphe non orienté, V: Ensemble de nœuds, E: Ensemble de liens
$n =\| V \|$	Nombre de nœuds
$m =\| E \|$	Nombre de liens
A_G	La matrice d'adjacence du graphe G
$\Gamma(x)$	Ensemble de voisins directs d'un nœud
$d_x =\| \Gamma(x) \|$	Degré de x
$dist(x, y)$	Distance géodésique entre les nœuds x et y

3.2.1 *Approches centrées groupe*

Les nœuds sont regroupés dans une même communauté en fonction de leurs propriétés topologiques partagées. L'exemple le plus trivial est d'assimiler une communauté à une clique maximale dans le graphe ou à une γ-dense quasi clique. Une clique est un sous-graphe complet. Une clique est maximale si on ne peut l'étendre qu'en ajoutant de nouveaux nœuds. Une γ-dense quasi clique est un sous-graphe dont la densité est supérieure à un certain seuil $\gamma \in [0, 1]$. Or, le problème de calcul des cliques maximales est un problème NP-difficile, ce qui rend difficile d'envisager son utilisation dans le contexte de très grands graphes. Un autre concept utile, souvent employé dans le domaine de l'analyse des réseaux sociaux, est le concept de *K-core*. Un K-core est un sous-graphe connexe maximal dans lequel le degré de chaque nœud est supérieur ou égal à k. Les graphes de terrain sont principalement des graphes très parcimonieux, de telles structures sont souvent minoritaires

dans les graphes. Par contre, des groupements denses des nœuds peuvent servir comme des graines pour la détection des communautés (voir section 3.2.4).

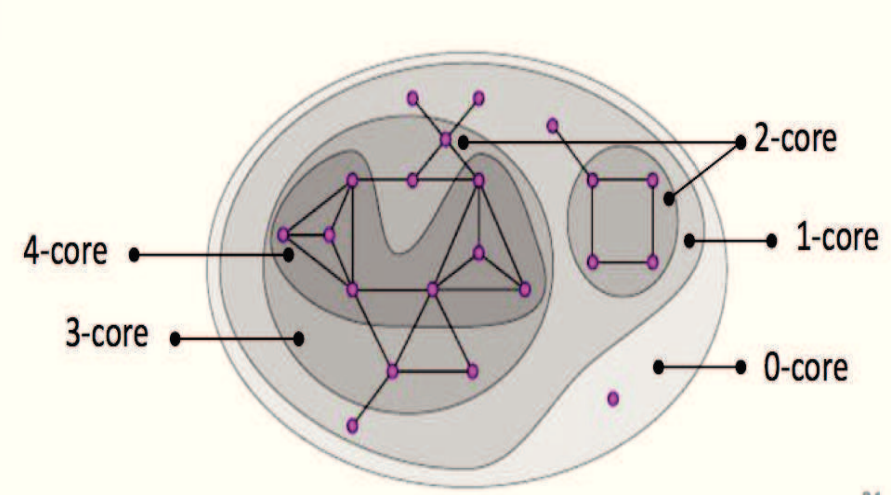

FIGURE 3.1 – Exemple de K-core dans un graphe - Exemple tiré de [Papadopoulos *et al.* 2012]

3.2.2 *Approches centrées réseau*

Un examen de la structure globale du réseau est effectué afin d'identifier les communautés à partir du graphe. Dans la littérature, la plupart des approches proposées s'appuient sur un schéma de calcul prenant en considération la connexion globale du graphe cible. Une classification des approches centrées réseau a été proposée par [Tang et Liu 2010] où nous en distinguons trois familles:

1. **Les approches de clustering:** une approche simple a été proposée par [Aggarwal et Reddy 2013] pour la détection de communautés consistant à transformer ce problème en problème classique de clustering de données.

   Étant donné n individus à regrouper en clusters, plusieurs algorithmes commencent à calculer une matrice de similarité $\mathcal{S}$ de dimension $n \times n$ où un élément S_{ij} est la similarité entre deux individus i et j calculée par une mesure de similarité donnée. Dans le cas d'un graphe G de n nœuds, il est aussi possible de construire une matrice de similarité entre les nœuds du graphe en utilisant une mesure de similarité topologique. Différentes mesures de similarité topologiques peuvent être définies. Nous les

classifions en trois catégories:

Les mesures basées sur le voisinage des nœuds: dites aussi *les mesures locales*. Le tableau 3.2 résume les principales mesures locales les plus utilisées.

Mesure	*Formule*	*Référence*
Voisins communs (VC)	$sim^{VC}(x,y) = \parallel \Gamma(x) \cap \Gamma(y) \parallel$	[Lü et Zhou 2011]
Cosine (ou indice de Salton)	$sim^{cos}(x,y) = \frac{\|\Gamma(x)\cap\Gamma(y)\|}{\sqrt{\|\Gamma(x)\|\times\|\Gamma(y)\|}}$	[Salton et McGill 1983]
Jaccard	$sim^{Jaccard}(x,y) = \frac{\|\Gamma(x)\cap\Gamma(y)\|}{\|\Gamma(x)\cup\Gamma(y)\|}$	[Jaccard 1901]
Adamic-Adar (AA)	$sim^{AA}(x,y) = \sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{log(\|\Gamma(z)\|)}$	[Adamic et Adar 2003]
Allocation de ressource (RA)	$sim^{RA}(x,y) = \sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\|\Gamma(z)\|}$	[Zhou *et al.* 2009]
Attachement préférentiel (AP)	$sim^{AP}(x,y) = d_x \times d_y$	[Barabási et Albert 1999]
Sorensen Index	$sim^{Sorensen}(x,y) = \frac{2\times\|\Gamma(x)\cap\Gamma(y)\|}{\|\Gamma(x)\|+\|\Gamma(y)\|}$	[Sorensen 1948]
HPI [1]	$sim^{HPI}(x,y) = \frac{\|\Gamma(x)\cap\Gamma(y)\|}{min(\|\Gamma(x)\|,\|\Gamma(y)\|)}$	[Ravasz *et al.* 2002]
HDI [2]	$sim^{HPI}(x,y) = \frac{\|\Gamma(x)\cap\Gamma(y)\|}{max(\|\Gamma(x)\|,\|\Gamma(y)\|)}$	[Ravasz *et al.* 2002]

TABLE 3.2 – Mesures de similarité dyadiques centrées voisinage

Les mesures basées sur les chemins entre les nœuds: dites aussi *les mesures globales*. Parmi les principales mesures basées sur les chemins, nous prenons:

La proximité: $sim^{proxi}(x,y) = \frac{1}{dist(x,y)}$: Plus la distance géodésique entre deux nœuds est petite plus la proximité des deux nœuds est grande. Or, rappelons qu'une caractéristique phare des graphes de terrain est le faible degré de séparation. Autrement dit, la distance moyenne entre chaque couple des nœuds est faible. Ce qui rend une telle mesure peu discriminante dans beaucoup de situations.

La mesure de Katz: soit $\sigma^l(x,y)$ l'ensemble des chemins de longueur l reliant deux nœuds x et y. La mesure de Katz proposée initialement dans Katz [1953] est définie par:

$$sim^{katz}(x,y) = \sum_{l=1}^{\infty} \beta^l \times \parallel \sigma^l(x,y) \parallel \tag{3.1}$$

où $\beta << 1$ est un facteur favorisant la prise en compte des chemins courts. Fouss *et al.* [2007] montrent que si β est inférieur à la plus grande valeur propre de A_G alors le calcul de cette mesure pour chaque couple de nœuds converge pour les valeurs calculées par la formule matricielle suivante:

$$sim^{Katz} = (I - \beta \times A_G)^{-1} - I \tag{3.2}$$

où I est la matrice identité. Le calcul de cette mesure est très coûteux pour les grands graphes.

Les mesures semi-locales: une mesure semi-locale vise à réaliser un compromis entre l'exploration de la structure du graphe qui dépasse le simple voisinage d'une part, et l'efficacité computationnelle d'autre part. Souvent, une mesure semi-locale est dérivée d'une mesure centrée chemin comme c'est le cas par exemple de la mesure tronquée de Katz définie par:

$$sim^{t-katz} = \sum_{l=1}^{l_{max}} \beta^l A^l \tag{3.3}$$

Une autre mesure similaire est l'indice de chemin local proposé dans [Zhou *et al.* 2009] et définie par:

$$sim^{LPI} = A^2 + \epsilon A^3 \tag{3.4}$$

où ϵ est un paramètre à fixer pour pondérer l'apport du nombre de chemins de longueur 3 à la valeur de cette mesure. Si $\epsilon = 0$, alors cette mesure revient à calculer le nombre des voisins communs.

2. **Les approches fondées sur les modèles des blocs:** le principe de ce type d'approches a été introduit dans [Tang et Liu 2010]. Le but est d'estimer la structure du graphe représentée par la matrice d'adjacence A par une structure de blocs. La

matrice d'adjacence A peut être estimée par le produit suivant:

$$A \approx S\Sigma S^T \tag{3.5}$$

où $S \in \{0,1\}^{n\times k}$ est la matrice d'appartenance des nœuds aux blocs, k est le nombre de blocs, et Σ est la matrice de densité d'interactions dans les blocs. Une fonction objective naturelle à minimiser est la suivante:

$$min \parallel A - S\Sigma S^T \parallel_F^2 \tag{3.6}$$

où $\parallel . \parallel_F^2$ est la norme de Frobenius: $\parallel A \parallel_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} |A_{ij}|^2$. Ce problème de minimisation est connu pour être NP-difficile quand S est à valeurs discrètes. Une approximation consiste à assouplir la contrainte et considérer la matrice S avec des valeurs continues, mais en imposant l'orthogonalité des vecteurs de S. Autrement dit, on impose que $SS^T = I_k$. Dans ce cas la valeur optimale de S sera les k premiers vecteurs propres de la matrice A associés aux k plus grandes valeurs propres de cette matrice.

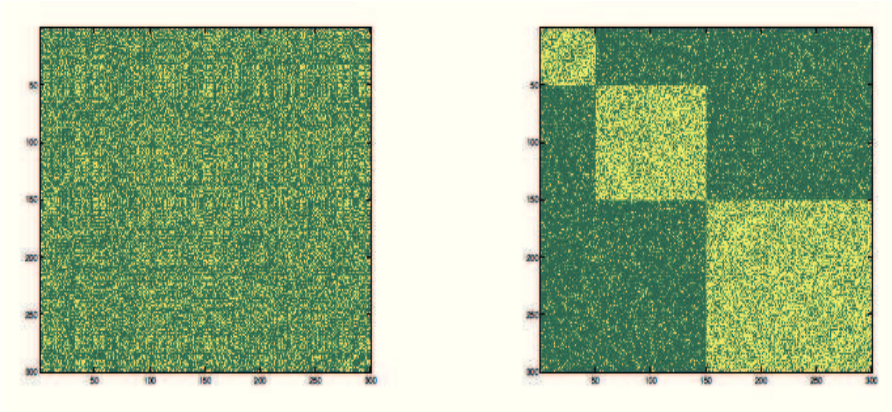

FIGURE 3.2 – Illustration de l'approche de modèle de blocs: exemple tiré de [Tang et Liu 2010]

3. **Approches d'optimisation:** Soit $\Pi = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{2^n}\}$ l'ensemble des partitions possibles d'un graphe G où n est le nombre des nœuds du graphe. Une autre façon pour traiter le problème de détection de communautés est de le ramener à un problème d'optimisation en se basant sur une fonction objective de qualité d'une partition. La

fonction objective la plus utilisée est le critère de la *modularité* [Newman 2004]. D'une manière informelle, la modularité d'une partition mesure la différence entre la proportion des liens intercommunautaires et la même quantité dans un modèle aléatoire où aucune structure communautaire n'est attendue. Ce modèle aléatoire est un graphe ayant les mêmes caractéristiques que le graphe initial c'est-à-dire même nombre de nœuds, même nombre de liens et la même distribution de degrés. La définition formelle de la modularité est comme suit:

Étant donné une partition $\mathcal{P} = \{c_1, \ldots, c_k\}$ composée de k communautés. Pour une communauté c_i, la qualité est donnée par $\sum_{i,j \in c_i} (A_{ij} - \frac{d_i d_j}{2m})$. Pour une partition, la qualité est égale à la somme des qualités de chacune de ses composantes: $\sum_{c_i \in \pi} \sum_{i,j \in C_i} (A_{ij} - \frac{d_i d_j}{2m})$.

La modularité d'une partition $\mathcal{P}$ est alors donnée par la formule suivante:

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{c \in \mathcal{P}} \sum_{i,j \in c} (A_{ij} - \frac{d_i d_j}{2m}) \tag{3.7}$$

Le terme $\frac{1}{2m}$ est ajouté pour normaliser les valeurs possibles de Q dans l'intervalle $[-1, 1]$. La maximisation de la modularité est un problème NP-difficile [Brandes *et al.*

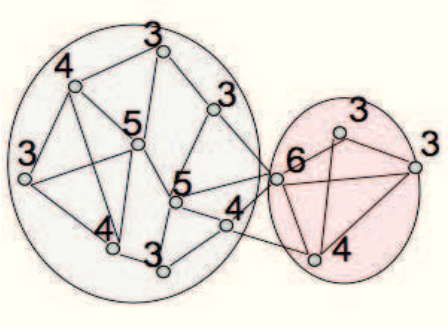

FIGURE 3.3 – Exemple de calcul de la modularité: $Q = \frac{(15+6)-(11.25+2.56)}{25} = 0.275$

2008]. Des méthodes d'optimisation sont proposées pour calculer, en temps et en espace polynomiaux, des partitions que l'on espère proches de l'optimum. Des méthodes d'optimisation directe utilisant les techniques d'algorithmique génétique ([Li et Song 2013], [Pizzuti 2012] et [Cai *et al.* 2011]), de recuit simulé ([Reichardt et Bornholdt 2006] et [Guimera *et al.* 2004]) ou de l'optimisation extrémale [Duch et Arenas 2005]

ont été proposés. Cependant, les heuristiques les plus appliquées sont fondées sur le principe de la classification hiérarchique. Deux approches totalement différentes sont largement expérimentées:

- Les approches agglomératives (ou ascendantes) selon lesquelles on part de la partition atomique (ensemble des singletons), et on fusionne deux communautés à chaque itération. Les communautés à fusionner sont celles qui promettent une modularité maximale. Des exemples de ces approches sont donnés dans [Blondel *et al.* 2008], [Newman 2004], [Donetti et Munoz 2004], [Pons et Latapy 2005].
- Les approches divisives (ou descendantes) dans lesquelles on part du graphe entier. À chaque itération, une communauté cherche à se scinder en deux de sorte à maximiser la modularité. Des exemples de cette approche sont donnés dans [Newman 2004], [Lancichinetti *et al.* 2008].

Les deux types d'approches produisent des hiérarchies de communautés. Dans [Tang et Liu 2010], une formulation matricielle du problème d'optimisation de la modularité est proposée. On définit la matrice

$$B = A - \frac{dd^T}{2m} \tag{3.8}$$

L'expression de la modularité d'une partition (voir 3.7) peut alors être formulée comme suit:

$$Q = \frac{1}{2m}\sum_{C} S_C^T B S_C = \frac{1}{2m} Tr(S^T B S) = Tr(S^t \tilde{B} S) \tag{3.9}$$

où $S_C \in \{0,1\}^n$ est le vecteur d'appartenance communautaire des nœuds dans C, S est la matrice d'indication d'appartenance d'un couple de nœuds à une même communauté, et

$$\tilde{B} = \frac{1}{2m} B = \frac{A}{2m} - \frac{dd^T}{(2m)^2} \tag{3.10}$$

La maximisation de Q peut se ramener alors au calcul des k premiers vecteurs propres associés aux k valeurs propres les plus grandes de la matrice $\tilde{B}$ sous condition de relaxation spectrale de S (i.e. $SS^T = I$) [Newman 2006]. Les approches fondées sur l'optimisation de la modularité se basent sur les hypothèses suivantes:

(a) Pour un réseau à structure communautaire, les partitions qui possèdent des valeurs de modularité très grandes sont structurellement identiques.

(b) On trouve un bon partitionnement si la valeur de modularité est maximale.

(c) C'est possible de déterminer une partition pour laquelle la modularité est maximale dans le cas où le réseau possède une structure communautaire.

Or, de récentes études ont montré que les trois hypothèses énumérées ci-dessus sont toutes fausses. Dans [Fortunato et Barthelemy 2007] les auteurs montrent que les algorithmes fondés sur l'optimisation de la modularité souffrent d'un problème de limite de résolution dans le sens qu'ils ne peuvent pas distinguer des communautés plus petites d'une certaine taille limite. Pour des graphes non pondérés la maximisation de la modularité ne permet pas de distinguer des communautés ayant un nombre de liens inférieur à $\sqrt{\frac{m}{2}}$. Dans une tentative de traitement de ce problème de limite de la résolution, une correction de la fonction de la modularité est proposée dans [Reichardt et Bornholdt 2006] en ajoutant un paramètre de résolution λ comme suit:

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{c \in \mathcal{P}} \sum_{i,j \in c} (A_{ij} - \boldsymbol{\lambda} \frac{d_i d_j}{2m}) \tag{3.11}$$

Plus la valeur de λ est grande plus les communautés de petite taille seront favorisées par Q puisque la maximisation de Q nécessite la minimisation du terme $\boldsymbol{\lambda} \frac{d_i d_j}{2m}$. Inversement, les communautés de grandes tailles seront favorisées en diminuant λ. Pour $\lambda = 1$, nous obtenons la même fonction de modularité initiale. Si cette nouvelle fonction de modularité, appelée *modularité mutli-résolution*, peut être réglée pour explorer des communautés à différentes échelles, elle apporte néanmoins une réponse partielle au problème de la limite de résolution puisque les tailles de communautés dans les réseaux réels sont très hétérogènes et suivent aussi une distribution selon une loi de puissance. D'autre part, [Lancichinetti et Fortunato 2011] montrent que la maximisation de la modularité n'a pas seulement tendance à fusionner les petits groupes, mais aussi à éclater des grandes communautés, et il semble impossible d'éviter simultanément les deux problèmes.

L'étude reportée dans [Good *et al.* 2010] montre l'existence d'un grand nombre de partitions très différentes entre elles, mais qui ont une valeur de modularité optimale.

Ce plateau étendu de partitions différentes entre elles, mais qui ont des modularités maximales explique les différences dans les résultats des différentes approches d'optimisation de la modularité. [Aynaud et Guillaume 2010] montrent que les algorithmes de maximisation de la modularité sont très sensibles à des perturbations minimales appliquées au graphe étudié.

3.2.3 *Approches centrées propagation*

Comme son nom l'indique, ce type d'approche applique souvent un processus d'émergence de la structure communautaire par un échange de messages entre les nœuds voisins. Principalement, les approches centrées propagation se basent sur la propriété de la densité des liens intra communautés. Dans le cas où les liens inter communautés sont faibles et à cause de la densité relative des communautés, nous constatons qu'il est plus probable qu'un *signal* émis par un nœud et rediffusé par ses voisins de rester dans la communauté du nœud source que de se propager vers les autres communautés. Cette propriété a été exploitée autrement dans différents algorithmes. Par exemple, les algorithmes de propagation de label (LPA) présentés dans l'algorithme 1 [Raghavan *et al.* 2007] sont basés sur l'idée de propagation des labels. Ce sont des algorithmes itératifs dont à chaque étape un nœud choisit le label le plus fréquenté par ses voisins directs jusqu'à la convergence vers une structure communautaire. La propagation de labels peut se faire en mode synchrone, asynchrone ou semi-synchrone. L'avantage de ce type d'approche est qu'elle est rapide, mais souffre de problème de convergence (problème d'oscillation) et de robustesse (où les différentes exécutions donnent de différentes solutions pour un même réseau). Il y a eu des solutions pour pallier la convergence en utilisant les approches semi-synchrone [Cordasco et Gargano 2012] avec coloriage de graphe, mais cela ne résout pas le problème d'instabilité. Pour le rendre stable, il y a d'autres approches proposées soit par la limitation des labels [Leung *et al.* 2009], soit par la propagation équilibrée [Šubelj et Bajec 2011], soit en utilisant les approches basées sur l'ensemble clustering ([Seifi 2012], [Lancichinetti et Fortunato 2012]).

$\Gamma^l(v)$: représente l'ensemble des voisins ayant le label l. Il existe d'autres algorithmes centrés propagation basés sur les techniques de propagation de labels [Corlette et Shipman III

Algorithm 1 Algorithme de Propagation de Label

Require: $G =< V, E >$ un graphe connecté,
1: Initialisation de chaque nœud par un label unique l_v
2: **while** labels sont instables **do**
3: **for** $v \in V$ **do**
4: $l_v = \text{argmax}_l \, \Gamma^l(v) \, ||$
5: **end for**
6: **end while**
7: **return** communautés à partir des labels

2010], [Raghavan *et al.* 2007], [Xie et Szymanski 2011], [Šubelj et Bajec 2011], [Gregory 2010].

3.2.4 *Approches centrées graine*

Ces approches sont fondées sur deux étapes fondamentales:

- La détermination des *graines* qui correspondent à un ensemble de nœuds ou à des groupes de nœuds dans le graphe.
- L'application d'un processus de calcul des communautés locales qui jouent un rôle particulier (cœurs de communautés) autour des graines afin de détecter les communautés dans le réseau.

L'algorithme 2 présente les grandes lignes d'un algorithme typique de détection de communautés centré graine. Nous distinguons les trois principales étapes suivantes:

1. Calcul des graines.
2. Calcul des communautés locales des graines.
3. Calcul des communautés à partir de l'ensemble des communautés locales calculées à l'étape 2.

Différentes mesures de choix de graine ont été proposées. Soit en utilisant les mesures classiques de centralité [Khorasgani *et al.* 2010], [Shah et Zaman 2010] dans le cas où la graine est composée d'un seul nœud. Soit en se basant sur la connectivité dans le cas où la graine est composée d'un ensemble de nœuds [Papadopoulos *et al.* 2011]. Plusieurs techniques d'expansion des graines sont proposées. La plupart des algorithmes utilisent les heuristiques développées pour l'identification de communautés locales. Par contre, ces approches ne peuvent pas assurer dans la structure communautaire calculée de couvrir

Algorithm 2 Algorithme général de détection de communautés centré graine

Require: $G =< V, E >$ un graphe connecté,
1: $\mathcal{C} \leftarrow \emptyset$
2: $S \leftarrow$ **calcul_graine(G)**
3: **for** $s \in S$ **do**
4: $C_s \leftarrow$ **calcul_local_com(s,G)**
5: $\mathcal{C} \leftarrow \mathcal{C} + C_s$
6: **end for**
7: **return calcul_communauté**($\mathcal{C}$)

l'ensemble des nœuds d'un graphe. [Yakoubi et Kanawati 2014] ont proposé une approche originale où après avoir identifié les graines, chaque nœud dans le graphe (graine ou pas) calcule un vecteur de préférence d'appartenance aux communautés de chaque graine. En effet, cette appartenance communautaire des nœuds est déterminée à travers une procédure de vote entre chaque nœud et ses voisins directs. Dans [Kanawati 2014], une étude comparative plus approfondie des approches centrées graine est présentée. Dans cette section, nous expliquons un exemple de ces algorithmes centré graines appelé Licod[3] où les communautés se forment autour des nœuds *leaders*.

En effet, notre approche présentée dans le chapitre 4 se base sur l'adaptation de cet algorithme afin de pouvoir l'exploiter dans le cadre des graphes multiplexes. Licod est composé de trois étapes:

1. La première étape est l'identification de l'ensemble des Leaders $\mathcal{L}$ en utilisant la mesure de centralité. Un nœud est désigné comme leader si sa valeur de centralité dépasse les centralités de ses voisins. Il y a plusieurs mesures de centralités qui peuvent être employées telles que: la centralité d'intermédiarité (betweenness), la centralité de degré (degree) et la centralité de proximité (closeness). Par la suite, $\mathcal{L}$ est réduit en un ensemble $\mathcal{C}$ de communautés de leaders. Si deux leaders ont un nombre de voisins communs très élevé, alors ces deux leaders sont regroupés dans la même communauté.
2. Chaque $x \in V$ définit un vecteur de préférence P_x^0 où les communautés identifiées dans $\mathcal{C}$ sont triées par ordre décroissant. La version actuelle détermine le degré d'appartenance d'un nœud x à une communauté $c \in \mathcal{C}$ par $min_{c_i} dist(x, c_i) c_i \in c$.
3. Une phase d'intégration commence dès que chaque nœud a son vecteur de préférence.

3. Leaders Identification for Community Detection in Complex Networks

En effet, le vecteur de préférence d'un nœud est fusionné avec ceux de ses voisins directs. Ce qui permet de favoriser la plus dominante dans l'ensemble des nœuds voisins. La tâche de vote est assurée par les algorithmes issus de la théorie de choix social [Chevaleyre *et al.* 2007]. Ce processus est répété jusqu'à stabilisation du vecteur d'appartenance de chaque nœud.

4. À la stabilisation, chaque nœud x est attribué aux communautés placées en tête de vecteur de préférence.

Algorithm 3 L'algorithme Licod

1: Déterminer l'ensemble des Leaders L
2: Répéter jusqu'à stabilisation ou max fois:
Chaque $x \in V$ trie les communautés $c \in C$ dans un ordre décroissant selon le degré d'appartenance P_x^0.
Pour chaque $x \in V$, calculer
$P_x^t = FusionVotes(P_y^{t-1} y \in X \cup \Gamma(x))$
Recalculer L
3: Retourner pour chaque nœud la communauté placée en tête de vecteur de préférence.

3.2.5 Évaluation des communautés

Dans la littérature, les algorithmes de détection de communautés peuvent être comparés en fonction de leurs complexités de calcul, d'espace mémoire et aussi en mesurant la qualité des communautés. Or, la tâche d'évaluation de la qualité des communautés souffre encore des problèmes malgré l'apparition de nombreux travaux dans ce domaine. Les mesures d'évaluation sont classées en trois grandes familles d'approches:

- Les indices d'évaluation par rapport à une partition de référence.
- Les indices d'évaluation des qualités topologiques des communautés.
- Évaluation guidée par une tâche.

3.2.5.1 Indices d'évaluation de communautés par rapport à une partition de référence

Une partition de référence d'un graphe G peut être issue à partir de l'un des trois processus suivants:

L'annotation par un expert: les graphes pour lesquels des experts ont défini des partitions de références sont souvent des graphes de très petite taille. Le tableau 3.3 décrit les caractéristiques des principaux réseaux réels annotés par des experts et qui sont souvent utilisés comme un benchmark pour les algorithmes de détection de communautés [4].

TABLE 3.3 – Quelques réseaux réels souvent utilisés comme un Benchmark pour les algorithmes de détection de communautés

Réseau	n	m	# **communautés**
Club de Karaté de Zachary	34	78	2
Football	115	616	11
Strike	24	38	3
Livres politiques	100	441	3
Dauphins	62	159	2

Les approches basées sur l'inférence de communautés: cette approche prend en compte dans le calcul, les aspects sémantiques notamment les nœuds et/ou les liens d'un réseau. Elle est utilisée récemment dans le travail de [Yang et Leskovec 2012]. Or, certaines règles appliquées pour l'inférence de communautés sont plus que discutables: par exemple, dans le cas du réseau des publications reportées dans la fameuse base Dblp, les auteurs proposent que si deux auteurs publient dans une même conférence alors ils appartiennent à une même communauté. Dans le cas du réseau social *Live Journal*, ils proposent d'assimiler les groupes de fans d'artistes à des communautés. Il est bien sûr difficile de définir des règles plus précises sans une analyse approfondie, mais les communautés ainsi définies sont souvent très nombreuses, de petite taille par rapport à la taille du réseau et sont mono thématiques.

Génération par un modèle artificielle: le but est de produire des graphes artificiels dont les structures communautaires sont paramétrables. Le modèle *LFR* introduit dans [Lancichinetti *et al.* 2011] est l'un des modèles les plus récents. En effet, les générateurs sont fondés sur l'identification des communautés denses. Par la suite, on les relie entre elles

4. La plupart de ces réseaux de Benchmark sont disponibles sur la page de jeux de donnée de Pakek: http://vlado.fmf.uni-lj.si/pub/networks/data/esna/

avec une densité paramétrable pour pouvoir contrôler la complexité de reconnaissance de la structure communautaire. L'avantage d'avoir une partition de référence est de pouvoir utiliser les différentes mesures de *distance* entre les clusters. Ces mesures ont été développées pour évaluer les approches de classification non supervisée appelées aussi *le clustering* [Aggarwal et Reddy 2013]. Soit un graphe G, V est l'ensemble de nœuds. Notons une partition de référence de V par $R = \{r_1, \ldots, r_n\}$. Soit $U = \{u_1, \ldots, u_m\}$ une partition calculée à travers un algorithme de détection de communautés. Afin de mesurer la similarité des deux partitions R et U, nous pouvons faire appel à l'une des mesures suivantes:

La pureté: la pureté d'une communauté $u_i \in U$ par rapport une partition R est définie par:

$$purity(u_i, R) = \max_{j=1 \to n} \frac{\parallel u_i \cap r_j \parallel}{\parallel u_i \parallel} \tag{3.12}$$

Cette fonction permet de déterminer le taux de recouvrement maximal entre la communauté u_i et les communautés définies dans R. Par conséquent, la pureté de la partition U par rapport à R est la somme pondérée de la pureté de chaque communauté de U par rapport à R:

$$purity(U, R) = \sum_{i=1}^{m} w_{u_i} \times purity(u_i, R) \tag{3.13}$$

où $w_{u_i} = \frac{\|u_i\|}{\sum_{l=1}^{m} \|u_l\|}$ est la prévalence de la communauté u_i dans U.

L'indice de rand: cette mesure, initialement proposée dans [Campello 2007], est basée sur le comptage de nombre d'accords entre deux partitions sur l'appartenance communautaire de chaque paire des nœuds. Soient:

- a le nombre des paires placées dans une même communauté selon U et R
- b le nombre des paires placées dans une même communauté selon U et en différente communauté selon R.
- c le nombre des paires placées dans une même communauté selon R et en différente communauté selon U.
- d le nombre des paires placées en différentes communautés selon U et selon R.

La somme $a+d$ donne le nombre d'accords entre les deux partitions, tandis que $b+c$ donne le nombre de désaccords. L'indice de rand est simplement défini par [5]:

$$Rand(U,R) = \frac{a+d}{\binom{n}{2}} \tag{3.14}$$

Une version ajustée de cet indice, appelée ARI [6] est proposée dans [Hubert et Arabie 1985] afin d'avoir une mesure dont l'espérance est nulle pour des partitions aléatoires. L'indice ARI est donné par:

$$ARI(U,R) = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c)+(c+d)(b+d)]}{(\binom{n}{2})^2 - [(a+b)(a+c)+(c+d)(b+d)]} \tag{3.15}$$

Mesures basées sur l'information mutuelle: l'information mutuelle mesure le degré de dépendance entre deux variables aléatoires X et Y. Elle est donnée par la formule générale:

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \tag{3.16}$$

où $H(X)$ est l'entropie de Shanon de la variable X et $H(X,Y)$ est l'entropie conjointe des deux variables X et Y. Rappelons que l'entropie d'une variable X est mesurée par:

$$H(X) = -\sum_{i=1}^{n_x} p(x_i) log(p(x_i))$$

et l'entropie conjointe de deux variables X, et Y est donnée par:

$$H(X,Y) = -\sum_{i=1}^{n_x}\sum_{j=1}^{n_y} p(x_i,y_j) log(p(x_i,y_j))$$

où $\{x_i\}$ (reps. $\{y_j\}$) est l'ensemble de n_x (reps. n_y) valeurs possibles de X (reps. Y). $p(x_i)$ est la probabilité pour X d'avoir la valeur x_i. et $p(x_i,y_j)$ est la probabilité pour que conjointement X ait la valeur x_i et Y ait la valeur y_j. En assimilant une partition U à une variable aléatoire, on peut écrire son entropie sous la forme:

$$H(U) = -\sum_{i=1}^{|U|} p(u_i) log(p(u_i))$$

5. rappel: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

6. Adjusted Rand Index

La probabilité d'appartenance $p(u_i)$ d'un nœud de V à la communauté u_i est égale à: $\frac{\|u_i\|}{n}$. En substituant dans l'expression 3.16, on obtient l'expression de l'information mutuelle entre deux partitions:

$$IM(U,R) = \sum_{u \in U} \sum_{r \in R} p(u,r) log(\frac{p(u,r)}{p(u)p(r)}) \tag{3.17}$$

Une version normalisée de l'information mutuelle est introduite dans [Strehl et Ghosh 2003] afin d'obtenir une mesure entre 0 et 1. L'information mutuelle normalisée (NMI) est donnée par:

$$NMI(U,V) = \frac{I(U,V)}{\sqrt{H(U)H(V)}} \tag{3.18}$$

Une version ajustée de la mesure NMI est récemment introduite dans [Vinh *et al.* 2009]. Dans [Meilă 2003], une mesure similaire est introduite, appelée variation de l'information (IV). Elle est donnée par:

$$VI(U,V) = H(U) + H(V) - 2 \times IM(U,V)$$

Les mesures de comparaison des partitions basées sur la théorie de l'information sont assez corrélées entre elles. En pratique, la mesure NMI reste la plus utilisée dans la littérature scientifique.

3.2.5.2 Mesures topologiques pour l'évaluation de communautés

Ici, nous distinguons deux types de mesures topologiques:

- Les mesures basées sur la qualité des communautés isolées formant une partition: beaucoup de mesures de qualité d'une communauté individuelle ont été introduites pour faire face au problème d'identification de la communauté d'un nœud connu aussi par la détection de communauté locale. Dans ce cadre, la qualité d'une partition est la moyenne des qualités de toutes ses communautés.

$$Q(\mathcal{C}) = \frac{\sum_i f(S_i)}{|\mathcal{C}|} \tag{3.19}$$

où $f()$ est une fonction de qualité d'une communauté. Le tableau 3.4 résume l'essentiel de ces fonctions de qualité. Nous utilisons la notation suivante:

- n_c: le nombre des nœuds dans la communauté c.
- m_c: le nombre des liens dans la communauté c.
- b_c: le nombre des liens sortants de la communauté c.
- d^m: la médiane des degrés des nœuds dans V

Mesure	**Type**	**Formule**	**Description**
Densité interne	int.	$\frac{2\times m_c}{n_c\times(n_c-1)}$	Densité du sous-graphe induit par la communauté
Degrés moyens	int.	$\frac{2\times m_c}{n_c}$	La moyenne des degrés internes
FOMD	int.	$\frac{\lvert\{u:u\in c,\lvert(u,v),v\in c\rvert>d^m\}\rvert}{n_c}$	Le pourcentage des nœuds internes ayant un degré > la médiane des degrés
TPR	interne	$\frac{\lvert\{u\in c\}:\exists v,w\in c:(u,v),(w,v),(u,w)\in E\rvert}{n_c}$	Taux d'implication dans des triangles
Expansion	Ext.	$\frac{b_c}{n_c}$	Nombre des liens sortants par nœud
Taux de coupe	Ext.	$\frac{b_c}{n_c\times(N-n_c)}$	Taux des liens sortants sur les liens sortants possibles
Conductance	Hybride	$\frac{b_c}{2m_c+b_c}$	La fraction des liens sortants
MAX-ODF	Hybride	$\max_{u\in c}\frac{\lvert\{(u,v)\in E,v\notin c\}\rvert}{d_u}$	Le max des liens sortants par nœud
AVG-ODF	Hybride	$\frac{1}{n_c}\times\sum_{u\in c}\frac{\lvert\{(u,v)\in E,v\notin c\}\rvert}{d_u}$	

TABLE 3.4 – Mesures topologiques d'évaluation d'une communauté c

- Les mesures globales permettant d'évaluer la qualité d'une partition: le critère le plus utilisé pour calculer la qualité intrinsèque d'une partition est la modularité, présentée dans la section 3.

3.2.5.3 Évaluation guidée par une tâche

L'évaluation guidée par une tâche semble être une alternative intéressante. Le principe est simple: soit T une tâche où la détection de communautés peut être appliquée. Soit $per(T, Algo^x_{com})$ un indicateur de performance de l'exécution de la tâche T en utilisant l'algorithme de détection de communautés $Algo^x_{com}$. Nous pouvons comparer les performances des deux algorithmes différents en fonction des indicateurs $per(T, Algo^x_{com})$ et $per(T, Algo^y_{com})$. Dans [Papadopoulos *et al.* 2012], les auteurs proposent d'utiliser la tâche de recommandation de tags dans les folksonomies. Dans [Yakoubi et Kanawati 2012], [Yakoubi et Kanawati 2013] la tâche de classification non supervisée de données *non relationnelles* est employée. Afin de classifier les données, l'approche commence par structurer les données sous forme de graphe de voisinage défini à l'aide d'une fonction de distance appropriée [Toussaint et Bhattacharya 1981]. Les algorithmes de détection de communautés peuvent alors être appliqués sur ce graphe pour identifier les clusters. Dans ce travail, nous utilisons l'évaluation guidée par une tâche afin d'évaluer notre approche de détection de communautés dans la tâche de recommandation de tags (voir chapitre 5).

3.3 Détection de communautés dans les graphes multiplexes

3.3.1 Définition

Un graphe multiplexe dit aussi multicouche est un graphe constitué de différentes couches dont chacune est composée par les mêmes nœuds notés par V, mais chaque couche de type différent correspond à une catégorie de relations. À titre d'exemple, nous prenons le réseau bibliographique de Dblp (voir figure 3.4) où [Davis *et al.* 2011] et [Pujari et Kanawati 2014] le définissent par un graphe multicouche. Chacune de ses couches correspond aux relations de co-citation, co-participation à une conférence et de co-publication et les nœuds sont les auteurs (ici nous avons trois types de relations donc trois couches).

La représentation formelle d'un réseau multiplexe composé par α couches est comme suit: $G =< V, E_1, \ldots, E_\alpha >$, où chaque couche correspond à une matrice d'adjacence $A^{[\alpha]}_G$.

Le tableau 3.5 résume les notations que nous utilisons dans la suite de cette section.

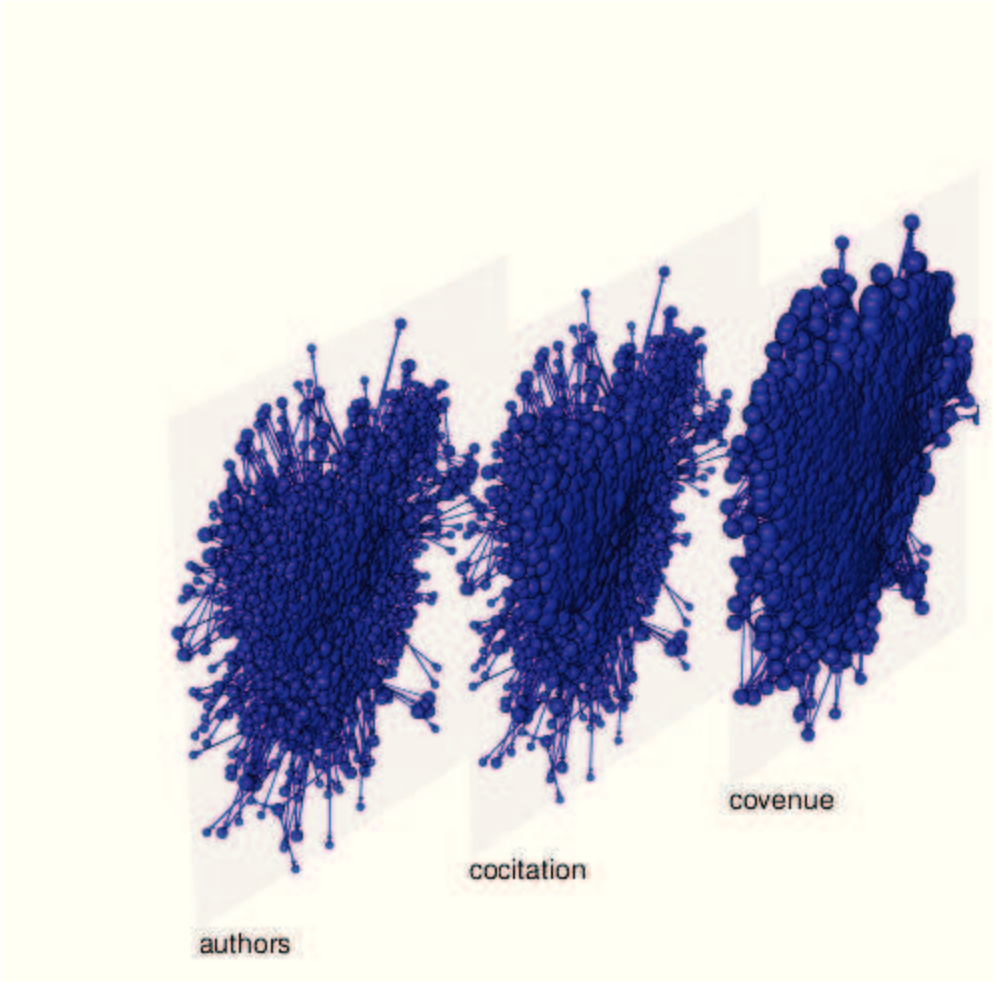

FIGURE 3.4 – Réseau multiplexe de Dblp: Les couches représentent les différents types de relations

TABLE 3.5 – Notations utilisées pour les réseaux multiplexes

Notation	Description
$A^{[k]}$	la matrice d'adjacence de la couche k
$d_i^{[k]}$	le degré de nœud i dans la couche k
$m^{[k]}$	le nombre des liens dans la couche k
C_{ij}^{kl}	le poids de lien inter couches k et l entre le nœud i et le nœud j

En se référant à la définition d'une communauté dans un graphe simple, une communauté multiplexe peut être vue comme un sous-graphe dense fortement lié dans le réseau multiplexe et qui est faiblement connecté aux autres communautés dans le graphe. Dans

un graphe multiplexe, le concept de densité reste vague: par exemple dans la figure 3.5, on ne sait pas quel est le graphe le plus dense dans le réseau multiplexe.

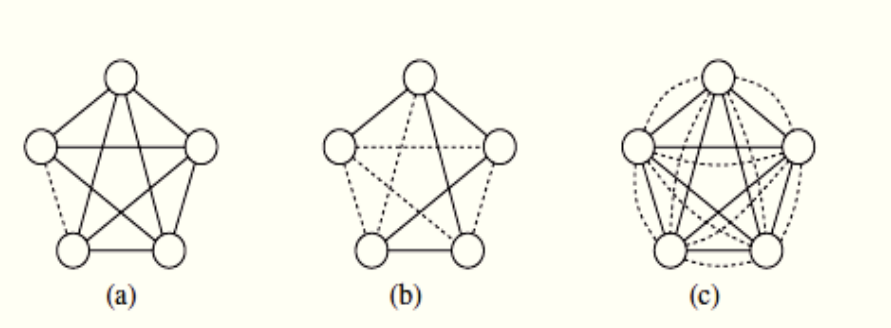

FIGURE 3.5 – Sous graphes denses dans un réseau multiplexe: source [Berlingerio *et al.* 2011]

Dans les travaux existants traitants les problèmes de détection de communautés dans les réseaux multiplexes, trois classes d'approches sont identifiées:

Agrégation des couches: ce type d'approche consiste à fusionner les couches du réseau multiplexe afin d'obtenir un réseau simple en utilisant une technique d'intégration des différentes couches. De ce fait, nous pouvons appliquer les algorithmes de détection des communautés classiques sur le nouveau réseau.

Agrégation des partitions: l'idée principale est d'identifier les structures communautaires séparément dans chacune des couches du réseau multiplexe, en appliquant un algorithme de détection de communautés. Puis, une agrégation des structures communautaires résultantes de chaque couche se fait en faisant appel à une stratégie de fusion des partitions.

Exploration simultanée des couches: le but est d'explorer les couches de réseau multiplexe dans le même processus de la détection des communautés.

3.3.2 Agrégation des couches (AC)

Soit $G_M =< V, E_1, \ldots, E_\alpha >$ un graphe multiplexe composé de α couches. $A^{[k]}$ présente la matrice d'adjacence. Le but est de transformer le graphe G_M en graphe simple pondéré $G =< V, E, W >$ avec W la matrice de poids des liens $e \in E$. En effet, ce type d'approche

tente de maintenir autant que possible les informations présentées dans le graphe multiplexe initial. Plusieurs métriques de calcul des poids sont proposées. Nous présentons les plus utilisées:

La pondération binaire: l'idée principale est de relier deux nœuds i et j dans le graphe simple s'il y a un lien entre ces deux nœuds dans au moins l'une des α couches [Berlingerio *et al.* 2011],[Suthers *et al.* 2013]. Plus formellement, nous avons:

$$w_{ij} = \begin{cases} 1 & \text{si } \exists 1 \leq i \leq \alpha : (i,j) \in E_i \\ 0 & \text{sinon} \end{cases} \tag{3.20}$$

Pondération en fonction de la fréquence: une autre méthode a été proposée par [Tang et Liu 2010] qui consiste à pondérer les liens, tout en calculant la moyenne de leurs poids dans toutes les couches. Formellement:

$$w_{ij} = \frac{1}{\alpha} \sum_{k=1}^{\alpha} A_{ij}^{[k]} \tag{3.21}$$

En plus, une deuxième manière de calcul de pondération de poids est introduite dans [Berlingerio *et al.* 2011] où le poids d'un lien cette fois-ci correspond à sa redondance dans les différentes couches:

$$w_{ij} = \parallel \{d : A_{ij}^{[d]} \neq 0\} \parallel \tag{3.22}$$

Pondération par une mesure de similarité: une manière plus générale est de pondérer un lien (i,j) dans un graphe simple représentant un graphe multiplexe, par une similarité multiplexe. On peut pratiquement utiliser les mêmes techniques employées pour le calcul des versions temporelles des mesures de similarité dyadiques dans un graphe dynamique [Potgieter *et al.* 2009]. En effet, l'historique de l'évolution d'un graphe sur β pas de temps peut être assimilé à un graphe multiplexe de β couches. La différence est que le temps induit un ordre sur les couches contrairement à un réseau multiplexe où aucun ordre ne peut être défini sur les couches. Dans [Berlingerio *et al.* 2011] les auteurs proposent l'utilisation du cœfficient de clustering d'un lien potentiel (i,j) comme une mesure de similarité pour la pondération du graphe G.

Combinaison linéaire: [Cai *et al.* 2005] mettent l'accent sur l'idée que les couches d'un multiplexe possèdent différents apports variables à la génération des communautés. Ainsi, l'intégration des couches peut se faire grâce à une combinaison linéaire des matrices d'adjacence des couches comme suit:

$$A = \sum_{k=1}^{\alpha} w_k A^{[k]} \tag{3.23}$$

où A est la matrice d'adjacence du graphe simple résultat. Le poids w_k est déterminé à partir des exigences de l'utilisateur sur l'appartenance ou non d'un ensemble de nœuds à des communautés bien précises. L'inconvénient majeur de ce type d'approche est la perte d'information sur la multiplicité des liens. En plus, la transformation du graphe multiplexe en graphe simple ne se fait que dans un seul sens et il n'est plus possible de retrouver le graphe initial. Cependant, il ne faut pas nier que ces approches ont une simplicité dans leur mise en place et la possibilité de choisir parmi les algorithmes de détection de communautés utilisés dans les réseaux simples.

3.3.3 Ensemble clustering (EC)

Ce type de méthode consiste à l'agrégation des partitions. Le principe consiste à choisir un algorithme de détection de communautés afin de l'appliquer à chacune des α couches de réseau multiplexe qui sont composées par l'ensemble des nœuds V. À l'issue de cette étape, nous obtenons α partitions. En agrégeant ces partitions résultantes grâce à des méthodes d'ensemble clustering [Topchy *et al.* 2005; Sammut et Webb 2010; Goder et Filkov 2008; Strehl et Ghosh 2003], nous obtenons une seule partition. [Seifi 2012] propose une approche de calcul des cœurs de communautés. Tout d'abord, cette approche construit une matrice $\mathcal{F}$ de dimension $n \times n$ où chaque élément F_{ij} de la matrice reflète la fréquence d'appartenance des nœuds i et j à une même communauté dans l'ensemble des partitions $P^{[k]}$. À partir de la matrice $\mathcal{F}$, un nouveau graphe G^{β} est créé dont les nœuds sont l'ensemble V et un lien est rajouté entre deux nœuds si $F_{ij} \geq \beta$. [Seifi 2012] désigne les composantes connexes du graphe G^{β} comme cœurs de communautés du graphe initial G. Le but principal de cette approche est de calculer les cœurs des communautés en fixant deux paramètres: un algorithme de détection de communautés instable et une variable N qui désigne le nombre

d'exécutions de l'algorithme. De plus, cet algorithme peut être utilisé dans le contexte d'agrégation des partitions issues des différentes couches d'un réseau multiplexe.

3.3.4 Exploration simultanée des couches (ESC)

Peu de travaux ont traité le problème d'exploration simultanée de toutes les couches d'un réseau multiplexe pour la détection des communautés. [Tang et Liu 2010] fait partie des premiers travaux qui ont essayé de transformer le problème de détection des communautés dans les graphes simples vers les graphes multiplexe. En effet, [Tang et Liu 2010] proposent un modèle unifié (voir figure 3.6) où l'agrégation peut se faire à deux endroits; soit au niveau des matrices d'utilité, soit au niveau des matrices d'indication d'appartenance communautaire. Cette approche n'est pas adaptée aux graphes de taille importante, car l'utilisation de l'algorithme K-means exige de connaître le nombre des communautés à trouver.

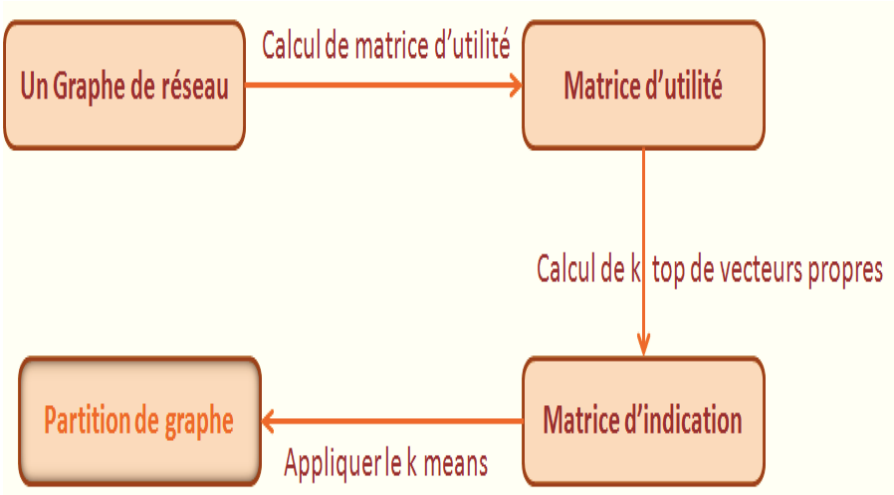

FIGURE 3.6 – Modèle unifié pour la détection des communautés [Tang et Liu 2010]

L'efficacité de la modularité et de son optimisation dans les approches de détection de communautés des graphes simples a motivé son extension dans le cadre des graphes multiplexes. [Mucha *et al.* 2010] ont proposé une généralisation de la modularité. La formule est la suivante:

$$Q_{multiplexe}(P) = \frac{1}{2\mu} \sum_{c \in P} \sum_{\substack{i,j \in c \\ k,l:1 \to \alpha}} \left(\left(A_{ij}^{[s]} - \lambda_k \frac{d_i^{[k]} d_j^{[k]}}{2m^{[k]}} \right) \delta_{kl} + \delta_{ij} C_{ij}^{kl} \right) \tag{3.24}$$

où $\mu = \sum_{\substack{j \in V \\ k,l:1 \to \alpha}} m^{[k]} + C_{jkl}$ est le facteur de normalisation, et λ_k est le facteur de résolution. Dans le contexte de réseau multiplexe, seulement les liens internes d'une couche sont des liens implicites reliant un nœud i à lui même dans les autres couches. Donc nous avons $C_{ij}^{kl} = 0 \; \forall i \neq j$. Parmi les travaux qui ont utilisé la modularité multiplexe nous trouvons [Costa *et al.* 2011]. Ils ont proposé une version inspirée de l'algorithme de Louvain appelée *GenLouvain*.

Principe de GenLouvain: Il est basé sur l'algorithme de Louvain [Blondel *et al.* 2008] et il implémente une méthode d'optimisation gloutonne locale de la modularité. À l'état initial, chaque nœud est affecté à une communauté différente des autres. L'algorithme applique ensuite une itération de succession de deux phases :

Phase d'affectation des nœuds: pour chaque nœud x on évalue le gain de la modularité multiplexe si on le déplace dans la communauté de ses voisins directs. On déplace x dans la communauté du voisin qui maximise le gain de la modularité. Si aucun gain n'est trouvé, le nœud reste dans sa communauté.

Phase de compression: on compresse le graphe obtenu en remplaçant chaque communauté par un seul nœud. Deux nœuds c_x, c_y dans le nouveau graphe sont liés par un lien s'il existe un lien entre un nœud de la communauté représentée par c_x et un nœud de la communauté représentée par c_y. Le poids de lien entre deux communautés est égal à la somme des poids des liens reliant des nœuds de deux communautés.

L'algorithme s'arrête s'il n'y avait plus la possibilité de réaffectation des nœuds ou si un maximum de modularité est atteint. La complexité théorique de l'algorithme n'est pas étudiée, mais d'une manière expérimentale, cette complexité est évaluée à $\mathcal{O}(nlogn)$ ce qui fait de Louvain la méthode la plus rapide pour l'identification de communautés.

Récemment, [Battiston *et al.* 2013] ont proposé d'étendre les mesures dans les graphes simples afin de pouvoir les utiliser dans le contexte des graphes multiplexes. Cette approche

repose sur l'idée qu'un nœud (ou lien) doit être impliqué dans plus qu'une couche afin de pouvoir être pris en compte, sinon sa valeur est nulle. Par exemple, la définition de degré d'un nœud dans un réseau multiplexe se fait comme suit: Soit $d_i^{[tot]} = \sum_{i=2}^{\alpha} d_i^{[i]}$ le degré total d'un nœud i dans le réseau multiplexe. Le degré multiplexe du nœud i est défini par :

$$d_i^{multiplexe} = -\sum_{k=1}^{\alpha} \frac{d_i^{[k]}}{d_i^{[tot]}} log\left(\frac{d_i^{[k]}}{d_i^{[tot]}}\right) \tag{3.25}$$

La fonction d'entropie proposée par [Battiston *et al.* 2013] exige l'implication d'un nœud dans plus d'une couche du multiplexe. Le degré multiplexe d'un nœud x est nulle si tous ses voisins sont concentrés dans une seule couche. Cependant, elle atteint sa valeur maximale si le nombre de ses voisins est le même dans toutes les couches.

3.3.5 Critères d'évaluation

Dans la section 3.2.5, nous avons fait un aperçu des différents critères d'évaluation des communautés dans le contexte des graphes simples. Par contre, peu de travaux ont traité cette problématique dans le réseau multiplexe. Actuellement, il n'existe pas des graphes réels de référence ou même des graphes générés d'une manière artificielle permettant de produire des réseaux multiplexes de benchmark. Par conséquent, la mesure de la modularité multiplexe est considérée comme mesure générale au niveau des indicateurs topologiques sans oublier ses limites au niveau de son optimisation. Un deuxième indicateur topologique appelé *mesure de redondance* a été proposé par [Berlingerio *et al.* 2011]. En effet, ils utilisent la mesure de redondance pour évaluer si la structure communautaire est partagée ou pas par toutes les couches du multiplexe. La performance de cette technique dépendra du niveau d'importance des différentes couches pour la tâche d'identification des communautés. Le calcul de cet indicateur se fait comme suit. Soient:

- P l'ensemble des couples (u, v) qui sont directement connectés dans une couche au moins.
- $\bar{\bar{P}}$ l'ensemble des couples (u, v) qui sont directement connectés dans deux couches au moins.
- $P_c \subset P$ l'ensemble des liens dans la communauté c.
- $\bar{\bar{P}}_c \subset \bar{\bar{P}}$ le sous-ensemble de $\bar{\bar{P}}$ et qui sont aussi dans c.

La redondance d'une communauté c est alors donnée par:

$$\rho(c) = \sum_{(u,v)\in\bar{\bar{P}}_c} \frac{\parallel \{k : \exists A_{uv}^{[k]} \neq 0\} \parallel}{\alpha \times \parallel P_c \parallel} \tag{3.26}$$

La qualité d'une partition multiplexe peut être donnée par:

$$\rho(\mathcal{P}) = \frac{1}{\parallel \mathcal{P} \parallel} \sum_{c\in\mathcal{P}} \rho(c) \tag{3.27}$$

3.4 Conclusion

Dans ce chapitre, nous avons étudié les différentes approches de détection de communautés dans les réseaux multiplexes. En effet, il existe plusieurs classes d'approches. Certains travaux classiques font recours à l'agrégation des couches en amont du clustering ou à l'agrégation des partitions obtenues à l'issue du clustering. D'autres approches traitent le réseau multiplexe en tant qu'entité entière et essaient d'explorer toutes les couches simultanément. Par ailleurs, les approches basées sur l'optimisation de la modularité souffrent de plusieurs problèmes (présentés dans la section 3) notamment la limite de résolution et de la maximisation de la modularité dans le cadre des graphes non pondérés. Dans la même logique, les approches centrées propagation, notamment LPA, souffrent des problèmes d'instabilité et de robustesse (voir section 3.2.3). Nous écartons ces deux types d'approches et nous nous intéressons à l'exploration d'un algorithme centré graine dans un cadre de réseaux multiplexes. Dans le chapitre suivant, nous introduisons notre approche de détection de communautés Mux-Licod en explorant simultanément les couches de réseau multiplexe. Ce type d'approche va nous permettre d'avoir en premier temps des communautés de meilleures qualités (chapitre 4) et en deuxième temps d'améliorer la recommandation de tags (chapitre 5). Le majeur inconvénient que nous rencontrons est le manque d'outils d'évaluation des communautés multiplexes.

Troisième partie

Contribution

Chapitre 4

Mux-Licod

4.1 Introduction

Comme nous l'avons vu dans le chapitre 3, la plupart des travaux dans le domaine d'analyse des réseaux complexes s'intéressent à des réseaux simples et statiques. Or, dans de nombreux contextes, les réseaux d'interactions peuvent être:

- *Hétérogènes*: ils contiennent différents types de nœuds et de différents types de liens.
- *Dynamiques*: où les nœuds et les liens peuvent varier avec le temps.
- *Attribués*: où les nœuds du réseau sont décrits par un ensemble d'attributs.

Récemment [Berlingerio *et al.* 2013] ont proposé un modèle permettant de prendre en compte les principales caractéristiques de ces réseaux. Nous rappelons qu'un réseau multiplexe est formé de plusieurs couches interconnectées. Chaque couche contient exactement le même ensemble de nœuds, mais le type de liens varie d'une couche à une autre. Dans ce type de réseaux, nous identifions deux familles d'approches pour aborder la problématique de détection de communautés dans une structure multiplexe. La première consiste à ramener le problème en un problème de détection de communautés dans un réseau monoplexe comme l'agrégation des couches (AC) [Berlingerio *et al.* 2011] et l'ensemble clustering (EC) [Brown 2010]. La deuxième famille consiste à faire évoluer un algorithme existant pour s'adapter au cas du réseau multiplexe tel est le cas des approches basées sur l'optimisation de la modularité [Mucha *et al.* 2010] dont nous connaissons les limites (voir section 3 du chapitre 3). C'est pour ces raisons que nous nous sommes orientés vers les approches locales centrées graines. Dans ce chapitre, nous proposons dans la section 4.2 notre algo-

rithme de détection des communautés centré graine *Mux-Licod* qui est une extension de Licod [Yakoubi et Kanawati 2014]. La généralisation de l'approche requiert la redéfinition des métriques de base comme: le degré, le voisinage, la centralité. Dans la section 4.3, nous menons nos expérimentations sur différents types de réseaux où l'évaluation se fait d'une manière non supervisée. En effet, comme nous l'avons évoqué dans le chapitre 3, le majeur problème des réseaux multiplexes est l'absence d'une partition de référence. Nous comparons dans la section 4.3.3, l'algorithme Mux-Licod avec les autres approches de détection de communautés présentées dans le chapitre 3 état de l'art en fonction des mesures de modularité multiplexe et de redondance expliquées dans la section 4.3.3.2. À la fin de ce chapitre, nous exposons les différents résultats et nous terminons par une conclusion.

4.2 L'algorithme Mux-Licod

4.2.1 Description informelle

Dans ce travail, nous proposons d'étendre une approche de détection de communautés centrée graine pour les réseaux multiplexes. Nous rappelons qu'une graine est un nœud ou un ensemble des nœuds sélectionnés d'une manière informée autour de laquelle les communautés locales peuvent être calculées. À partir de ces communautés locales, nous calculons une structure communautaire globale. Cela nous mène à décomposer le problème en trois sous problèmes: (1) Comment identifier les graines? (2) Comment calculer leurs communautés locales? Et (3) comment passer d'une communauté locale à une communauté globale? La présentation d'algorithme 2 typique centré graine a été introduite dans la section 3.2.4 du chapitre 3. Le choix d'une approche centrée graine est justifié par le fait qu'elle est basée sur des calculs locaux ce qui permet d'envisager son application sur de grands graphes [Kanawati 2014] contrairement aux approches basées sur la modularité et sur l'identification des cliques qui ne traitent pas les graphes de grandes tailles. Les grandes lignes de l'algorithme Mux-Licod se présentent comme suit:

1. **Calcul des graines:** le but de cette étape est d'identifier les nœuds qui semblent être des leaders. Plusieurs heuristiques peuvent être utilisées pour estimer le rôle d'un nœud. La détermination des graines se divise à son tour en deux étapes:
   – Premièrement, tous les nœuds ayant une centralité plus élevée que la centralité de

leurs voisins directs sont choisis comme des candidats pour être des graines. Dans l'algorithme 4 cette étape est réalisée par la fonction *isLeader()* (ligne 3).

- Deuxièmement, Mux-Licod intègre une phase de regroupement des leaders (ligne 7 dans l'algorithme 4) estimés être dans la même communauté. Cette étape permet de faire face à un éventuel grand nombre de leaders ce qui va affecter directement le nombre de communautés final.

Par conséquent, la question qui se pose est comment déterminer le degré d'un nœud ainsi que les degrés de ses voisins dans plusieurs couches d'un réseau multiplexe.

2. **Calcul des communautés locales des graines:** cette étape se fait également en deux temps:

   - Tout d'abord, chaque nœud calcule son degré d'appartenance à chaque communauté dans $\mathcal{C}$. Une liste ordonnée des communautés peut alors être obtenue pour chaque nœud, et dans laquelle les communautés ayant le plus grand *degré d'appartenance* sont classées en premier (lignes 9-13 dans 4).
   - Ensuite, chaque nœud ajuste sa liste de préférence d'appartenance aux communautés par la fusion de celle-ci avec les listes de préférences de ses voisins directs dans le réseau. Une fonction proposée dans [Yakoubi et Kanawati 2014] est basée sur la distance.

   À cette étape, nous avons besoin de déterminer la distance entre deux nœuds dans un réseau multiplexe.

3. **Calcul des communautés:** enfin, chaque nœud est affecté à la communauté placée en tête de la liste de préférence.

Nous avons remarqué précédemment que l'analyse des réseaux multiplexes nécessite la redéfinition des métriques de base habituellement appliquées aux réseaux simples ([Battiston *et al.* 2013], [Brodka et Kazienko 2014]). Dans la suite, nous proposons d'étudier trois principales métriques sur lesquelles Mux-Licod se base permettant de calculer respectivement *le voisinage* , *le degré d'un nœud*, et *la distance* dans le cadre multiplexe.

Algorithm 4 L'algorithme Mux-Licod

Require: $G =< V, E >$ un graphe multiplexe
1: $\mathcal{L} \leftarrow \emptyset$ {Ensemble des leaders}
2: **for** $v \in V$ **do**
3: **if** $isLeader(v)$ **then**
4: $\mathcal{L} \leftarrow \mathcal{L} \cup \{v\}$ /* *Calcul des candidats leaders* */
5: **end if**
6: **end for**
7: $\mathcal{C} \leftarrow computeCommunitiesLeader(\mathcal{L})$ /* *Calcul des leaders* */
8: **for** $v \in V$ **do**
9: **for** $c \in \mathcal{C}$ **do**
10: $M[v, c] \leftarrow membership(v, c)$ /* *Calculer le degré d'appartenance* */
11: **end for**
12: $P[v] =$ **sortAndRank(**$M[v]$**)**
13: **end for**
14: **repeat**
15: **for** $v \in V$ **do**
16: $P^*[v] \leftarrow \mathbf{rankAggregate}_{\mathbf{x} \in \{\mathbf{v}\} \cap \mathbf{\Gamma_G(v)}} \mathbf{P[x]}$
17: $P[v] \leftarrow P^*[v]$
18: **end for**
19: **until** Stabilisation de $P^*[v] \forall v$
20: **for** $v \in V$ **do**
21: /* *attribution* v *aux communautés* */
22: **for** $c \in P[v]$ **do**
23: **if** $|M[v, c] - M[v, P[0]]| \leq \epsilon$ **then**
24: $COM(c) \leftarrow COM(c) \cup \{v\}$
25: **end if**
26: **end for**
27: **end for**
28: **return** $\mathcal{C}$

4.2.2 Mise en œuvre

4.2.2.1 Le voisinage

Il y a plusieurs manières pour définir les voisins d'un nœud. Parmi les méthodes les plus simples, nous trouvons la mesure d'union/intersection de tous les nœuds voisins dans toutes les couches du réseau multiplexe. Une première proposition a été faite par [Brodka et Kazienko 2014], [Kazienko *et al.* 2010] pour définir le concept de voisinage d'un nœud. Dans [Brodka et Kazienko 2014], les auteurs définissent le voisinage multiplexe d'un nœud par l'introduction d'un seuil sur le nombre de couches dans lesquelles deux nœuds sont liés. Formellement $\Gamma_m(v) = \{u \in V \text{ tel que } count(k) \geq m : A_{uv}^{[k]} > 0\}$. Nous proposons ici une nouvelle mesure pour calculer le voisinage d'un nœud basée sur la similarité. L'idée est fondée sur le fait que si deux nœuds partagent plusieurs voisins dans toutes les couches combinées, cela signifie qu'ils sont étroitement liés et doivent être pris en compte dans la structure de voisinage multiplexe. Les étapes détaillées de calcul de mesure de voisinage multiplexe $\Gamma^{mux}(v)$ d'un nœud v sont:

1. Nous déterminons un ensemble de candidats $\Gamma_{tot}^{[k]}(v)$ tel que $\Gamma_{tot}^{[k]}(v) = \bigcup \Gamma(v)^{[k]} \forall k \in \{1, \ldots, \alpha\}$, α étant le nombre de couches.
2. Pour chacun des candidats $c \in \Gamma_{tot}^{[k]}(v)$, nous déterminons ses voisins $\Gamma_{tot}^{[k]}(c)$ de la même façon que dans la première étape: $\Gamma_{tot}^{[k]}(c) = \bigcup \Gamma(c)^{[k]} \forall k \in \{1, \ldots, \alpha\}$.
3. À ce niveau, nous avons besoin d'une métrique permettant le calcul de la similarité entre le nœud v et chacun des nœuds appartenant à l'ensemble des candidats $\Gamma_{tot}^{[k]}(v)$. Dans la littérature, il existe plusieurs mesures de similarité telle est le cas de la mesure du cosinus ou Jaccard. En utilisant *la mesure de similarité de Jaccard* la formule est la suivante:
$$Jaccard(v, c) = \frac{\|\Gamma_{tot}^{[k]}(v) \cap \Gamma_{tot}^{[k]}(c)\|}{\|\Gamma_{tot}^{[k]}(v) \cup \Gamma_{tot}^{[k]}(c)\|} \tag{4.1}$$
4. Un seuil $\delta \in [0, 1]$ est fixé pour pouvoir garder seulement les candidats dont les valeurs de similarité avec le nœud v sont supérieures ou égales à ce seuil.

Exemple: Nous prenons un exemple de réseau multiplexe composé par 3 couches présenté dans la figure 4.1.

Le but est de calculer les voisins du nœud numéro 7 avec les méthodes suivantes:

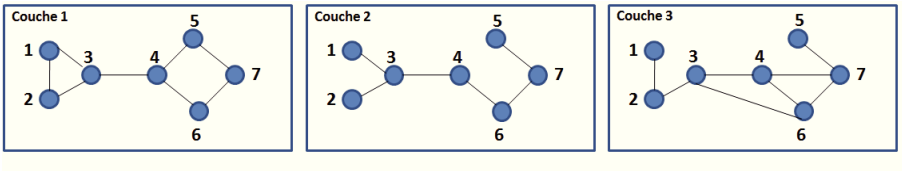

FIGURE 4.1 – Exemple de calcul de voisinage multiplexe du nœud 7

- $Union : \Gamma^{\bigcup}(7) = \{4, 5, 6\}$
- $Intersection : \Gamma^{\bigcap}(7) = \{5, 6\}$
- $Similarite_{mux}$: $\Gamma^{mux}(7) =?$

  $\Gamma^{mux}_{tot}(7) = \{4, 5, 6\}$

  $Jaccard(7, 4) = \frac{\|\Gamma_{tot}(7) \cap \Gamma_{tot}(4)\|}{\|\Gamma_{tot}(7) \cup \Gamma_{tot}(4)\|} = \frac{\|\{(5,6) \cap (3,5,6)\}\|}{\|\{(5,6) \cup (3,5,6)\}\|} = \frac{2}{3}$

  $Jaccard(7, 5) = \frac{\|\Gamma_{tot}(7) \cap \Gamma_{tot}(5)\|}{\|\Gamma_{tot}(7) \cup \Gamma_{tot}(5)\|} = \frac{\|\{(6,4) \cap (4)\}\|}{\|\{(6,4) \cup (4)\}\|} = \frac{1}{2}$

  $Jaccard(7, 6) = \frac{\|\Gamma_{tot}(7) \cap \Gamma_{tot}(6)\|}{\|\Gamma_{tot}(7) \cup \Gamma_{tot}(6)\|} = \frac{\|\{(5,4) \cap (4,3)\}}{\|\{(5,4) \cup (4,3)\}\|} = \frac{1}{3}$

  Nous gardons comme voisin du nœud 7 noté $\Gamma^{mux}(7)$ uniquement les nœuds ayant une similarité $Jaccard(7, ?) \geq \delta$, avec $\delta \in [0, 1]$.

  Si $\delta = 0.6$ alors $\Gamma^{mux}(7) = \{4\}$.

  Si $\delta = 0 \rightarrow \Gamma^{\bigcup}(7)$.

  Si $\delta = 1 \rightarrow \Gamma^{\bigcap}(7)$.

4.2.2.2 Le degré d'un nœud

Le degré d'un nœud est défini comme étant la cardinalité de l'ensemble des voisins directs. Il peut être défini comme la cardinalité de $|\Gamma(v)|$ de différentes méthodes utilisées pour calculer le voisinage. Dans [Brodka et Kazienko 2014], le degré multiplexe d'un nœud est simplement défini comme une agrégation des degrés du nœud dans chaque couche. Par exemple, cela peut être simplement défini comme la valeur moyenne des degrés des nœuds dans les différentes couches:

$$d_i^{multiplex} = \frac{\sum_{k=1}^{\alpha} d_i^{[k]}}{\alpha} \tag{4.2}$$

Toutefois, l'utilisation des fonctions d'agrégation (ex. min, max, somme) ne permet pas de prendre en compte les différentes répartitions du degré du même nœud dans les différentes couches. Une façon permettant de prendre en compte la distribution des degrés d'un nœud v dans les différentes couches, consiste à appliquer une fonction d'entropie comme proposée dans [Battiston *et al.* 2013]. Cette entropie est basée sur l'idée qu'un nœud (ou lien) doit être impliqué dans plus d'une couche afin d'être pris en compte. Si ce n'est pas le cas, sa valeur est nulle. Formellement, la mesure d'entropie est donnée comme suit:
Soit $d_i^{[tot]} = \sum_{i=1}^{\alpha} d_i^{[i]}$ le degré total d'un nœud i dans le réseau multiplexe. Le degré multiplexe d'un nœud i est défini par:

$$d_i^{multiplex} = -\sum_{k=1}^{\alpha} \frac{d_i^{[k]}}{d_i^{[tot]}} log\left(\frac{d_i^{[k]}}{d_i^{[tot]}}\right) \tag{4.3}$$

Le degré d'un nœud i est nul si tous ses voisins sont concentrés dans une seule couche. Cependant, il atteint sa valeur maximale, lorsque le nombre de ses voisins est le même dans toutes les couches. En reprenant l'exemple dans la figure 4.1, le degré du nœud 7 peut être calculé de différentes manières:

1. En utilisant les mesures de voisinage:
   - $d_7^{multiplex} = \left|\Gamma^{\cup}(7)\right| = 3$ en utilisant la fonction d'union.
   - $d_7^{multiplex} = \left|\Gamma^{\cap}(7)\right| = 2$ en utilisant la fonction d'intersection.
   - $d_7^{multiplex} = |\Gamma^{mux}(7)| = 1$ si $\delta = 0.6$ en utilisant la fonction basée sur la similarité.
2. En utilisant la mesure de Battiston:
   - $d_7^{multiplex} = 0.234$

4.2.2.3 La distance

D'une manière semblable à la définition du degré multiplexe, deux approches peuvent être appliquées pour définir les mesures dyadiques multiplexes (y compris le plus court chemin). Soit $X^{[k]}(u, v)$ une mesure dyadique simple impliquant les nœuds u et v dans la couche k. Deux versions différentes de la métrique dyadique multiplexe X peuvent alors être définies:

1.

$$X^{multiplex}(u, v) = \mathcal{F}(X^{[1]}(u, v), \ldots, X^{[\alpha]}(u, v)) \tag{4.4}$$

où $\mathcal{F}$ est une fonction d'agrégation (moyenne, min, max, ...).

2. Une autre définition est basée sur l'entropie pour les deux nœuds impliqués dans les différentes couches [Pujari et Kanawati 2014]:

$$X^{multiplex}(u,v) = -\sum_{k=1}^{\alpha} \frac{X(u,v)^{[k]}}{X^{[tot]}} \log(\frac{X(u,v)^{[k]}}{X^{[tot]}}) \tag{4.5}$$

où $X^{[tot]}(u,v) = \sum_{k=1}^{\alpha} X(u,v)^{[\alpha]}$.

Le plus court chemin peut être défini comme la moyenne des longueurs de plus court chemin dans chaque couche.

$$SPath(v,x) = \sum_{k=1}^{\alpha} \frac{path_k(x,v)}{\alpha} \tag{4.6}$$

avec $path_k(x,v)$ le nombre des liens entre les nœuds x et v dans α couches.

4.3 Expérimentations et résultats

Cette section décrit le protocole expérimental utilisé et décrit les résultats des diverses expérimentations menées afin de mettre en exergue notre contribution de détection de communautés dans les réseaux multiplexes ayant pour but l'amélioration de la tâche de recommandation de tags. Cette section présente enfin une étude comparative de Mux-Licod avec les approches existantes.

4.3.1 Les jeux de données

Dans un premier temps, nous allons travailler sur trois réseaux de petites tailles utilisés dans la littérature des réseaux multiplexes: le réseau d'innovations des médecins, le réseau des cabinets d'avocats et le réseau de Vickers. En deuxième temps, nous menons nos expérimentations sur des données de plus grande taille: le réseau bibliographique Dblp.

– **Réseau d'innovations des médecins (CKM)** [Coleman *et al.* 1957]: Coleman, Katz et Menzel ont recueilli des données sur l'innovation médicale. Les nœuds représentent des médecins. Trois relations entre les médecins sont modélisées: relations d'amitié, de demande de conseils et une troisième relation qui donne pour chaque médecin les confrères avec qui il préfère discuter.

- **Réseau de cabinets d'avocats (Lazega)** [Lazega 2001]: ce réseau modélise les relations entre les avocats associés. Trois relations sont mises en exergue: collaboration, conseil et amitié.
- **Réseau de Vickers**: les données ont été collectées auprès de 29 élèves de la septième année primaire dans une école de Victoria en Australie. Les élèves étaient amenés à désigner leurs camarades de classe sur un certain nombre de relations, y compris:

  1. Avec qui vous entendez-vous ?
  2. Qui sont vos meilleurs amis ?
  3. Avec qui préférez-vous travailler ?

  Chaque question représente une couche du multiplexe.

Graphes	# Nœuds	# Arêtes	Densité
Conseil	246	480	0.01592832
Discussion	246	565	0.01874896
Amitié	246	507	0.01682429

TABLE 4.1 – Réseau d'innovations des médecins: CKM

Graphes	# Nœuds	# Arêtes	Densité
Conseil	71	612	0.24627767
Collaborer	71	757	0.30462777
Amitié	71	855	0.34406439

TABLE 4.2 – Réseau des cabinets d'avocat: Lazega

Graphes	# Nœuds	# Arêtes	Densité
Meilleur-ami	29	181	0.44581281
Sortir-avec	29	361	0.88916256
Travailler-avec	29	199	0.4835966

TABLE 4.3 – Réseau de Vickers

- **Les réseaux de Dblp:** Dblp est une base bibliographique scientifique contenant principalement des articles liés à l'informatique et à partir desquels nous avons créé trois jeux de données. Les nœuds de tous les graphes correspondent aux auteurs et les liens dans chaque graphe désignent un type de relation différente. L'ensemble de données couvre une période de cinq ans (de 1980 jusqu'à 1985).

En effet, les liens entre les auteurs dans le graphe projeté de co-auteurs représentent les collaborations pour des publications scientifiques; deux auteurs sont reliés par un lien s'ils ont co-publié au moins un article. De même, deux auteurs sont liés dans le graphe co-venues s'ils ont participé à la même conférence. Tandis que dans le graphe co-citations, un lien est généré entre deux auteurs si ces deux derniers ont été cités dans le même papier.

Après avoir effectué des prétraitements, nous obtenons un réseau bibliographique multiplexe composé de trois couches.

Néanmoins, un problème reste à résoudre concernant les couches du réseau multiplexe qui doivent avoir le même ensemble de nœuds dans chacune des couches. Pour résoudre ce problème, nous extrayons la plus grande composante connectée à partir du graphe ayant la plus grande taille. Ce graphe correspond au graphe de co-auteurs. Ensuite, nous procédons de la même façon pour les graphes de co-citations et co-venues avec le même ensemble de nœuds de la composante extraite du premier graphe de la couche de co-auteurs. À ce niveau, nous obtenons trois graphes composés du même ensemble de nœuds mais avec des relations différentes entre les auteurs.

Le tableau ci-dessous présente les données sur les graphiques de DBLP.

Graphes	**# Nœuds**	**# Arêtes**	**Densité**
co-auteurs	2809	5109	0.001295439
co-venues	2809	251819	0.000000161
co-citations	2809	36187	0.000000780

TABLE 4.4 – Réseau Dblp

4.3.2 Étude des effets des paramètres de Mux-Licod

Mux-Licod possède beaucoup de paramètres. Pour cela, nous avons fait une étude sur les méthodes de fusion de votes (*Borda [Sculley 2007], Kemeny [Dwork* et al. *2001], Majorité*) en analysant les trois principaux paramètres de Mux-Licod: le voisinage, le degré et la distance. Les méthodes de fusion sont détaillées dans l'annexe 6. Tout d'abord, nous fixons une configuration de base ($voisinage = \Gamma_{mux}(v)$ avec $\delta \in [0,1]$, $degre = d_i^{multiplex}$ (équation 4.2) et distance=$\sum_{n=1}^{\alpha} \frac{path_n(x,v)}{n}$). Par la suite, nous étudions la variation de seuil δ de 0 à 1 par un pas de 0.1. Ce qui permet aussi d'étudier les modes Union (avec $\delta = 0$)

et Intersection (avec $\delta = 1$). En variant à chaque fois les paramètres, nous déterminons quatre modes:

- **Mode 1**= ($voisinage = \Gamma_{mux}(v)$ avec $\delta \in [0,1]$, $degre = |\Gamma_{mux}(v)|$, distance= $\sum_{n=1}^{\alpha} \frac{path_n(x,v)}{n}$).
- **Mode 2**= ($voisinage = \Gamma_{mux}(v)$ avec $\delta \in [0,1]$, $degre = d_i^{multiplex}$, distance= $\sum_{n=1}^{\alpha} \frac{path_n(x,v)}{n}$).
- **Mode 3**= ($voisinage = \Gamma_{mux}(v)$ avec $\delta \in [0,1]$, $degre = |\Gamma_{mux}(v)|$, distance= $X^{multiplex}(u,v)$.
- **Mode 4**= ($voisinage = \Gamma_{mux}(v)$ avec $\delta \in [0,1]$, $degre = d_i^{multiplex}$, distance= $X^{multiplex}(u,v)$ (équation 4.4).

Nous représentons ici les résultats trouvés avec la méthode *Borda* en termes de redondance et de modularité. Les autres méthodes de fusion des listes *Majorité* et *Kemeny* sont représentées dans l'annexe 6.

4.3.2.1 Synthèse

Nous remarquons que les résultats sont très semblables en utilisant les différentes méthodes de fusion des listes. Nous utilisons la méthode de fusion *Borda* pour avoir le classement final des quatre modes en fonction de redondance et de modularité.

Classement des modes en terme de redondance: en comparant les quatre modes, nous remarquons que les deux meilleurs sont: mode 1 et mode 2 en terme de maximum. Les résultats des deux modes 3 et 4 en utilisant la distance $X^{multiplex}(u,v)$ (équation 4.4) ne donnent pas de bons résultats.

- L1=[mode2, mode1, mode3, mode4]
- L2=[mode1, mode2, mode4, mode3]
- L3=[mode2, mode4, mode3, mode1]

- Résultat avec borda: mode1=7, mode2=4, mode3=10, mode4=9
- Résultat final=[**mode2, mode1**, mode3, mode4]

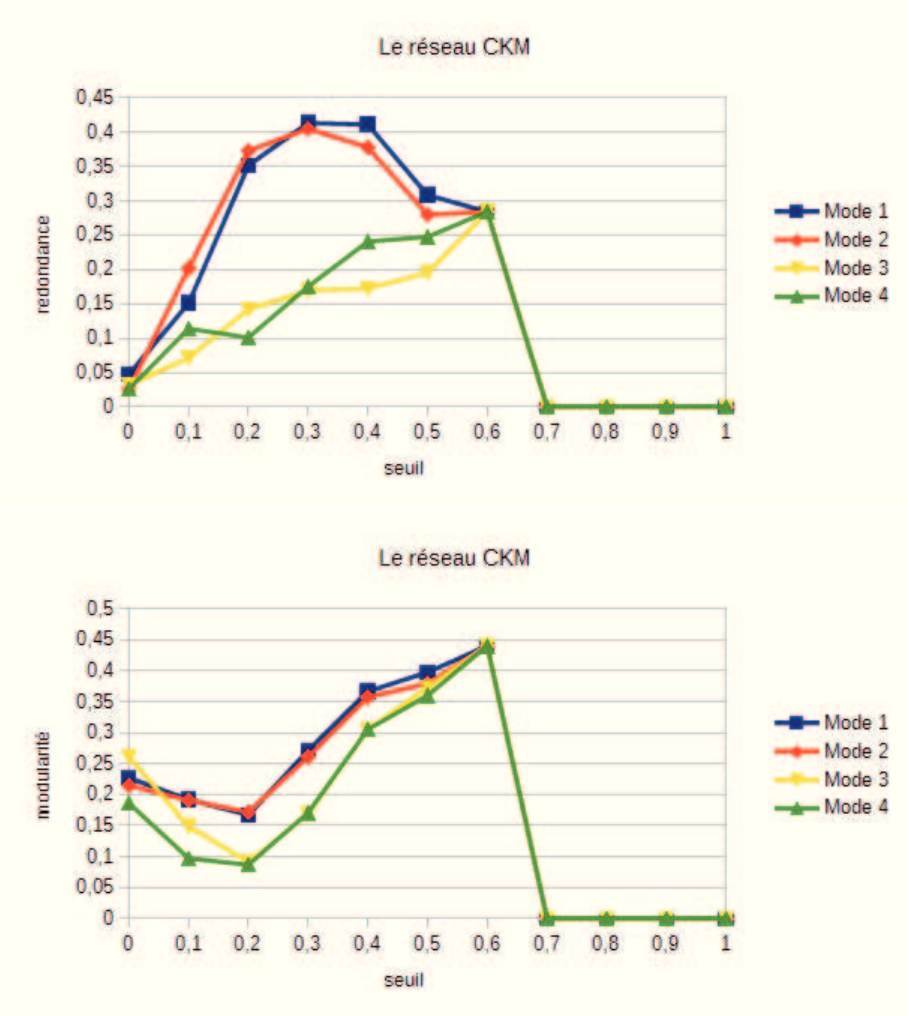

FIGURE 4.2 – Étude des paramètres de Mux-Licod sur le réseau CKM

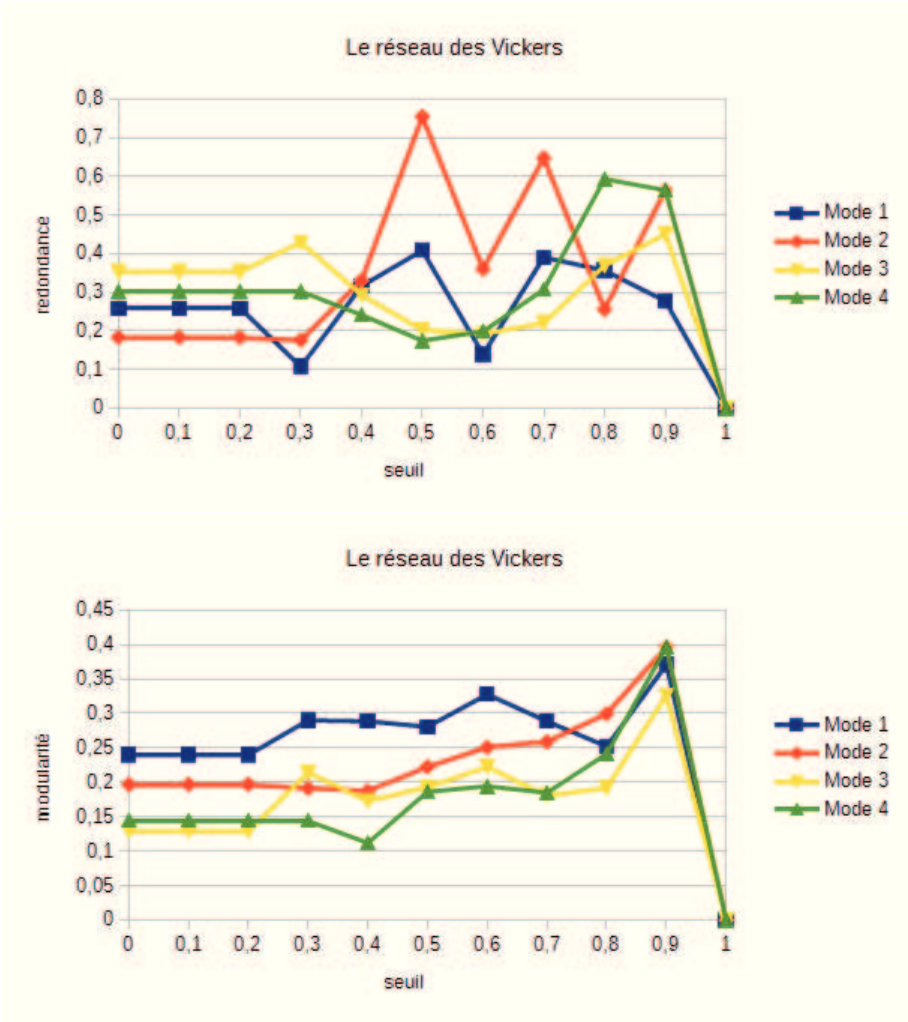

FIGURE 4.3 – Étude des paramètres de Mux-Licod sur le réseau Vickers

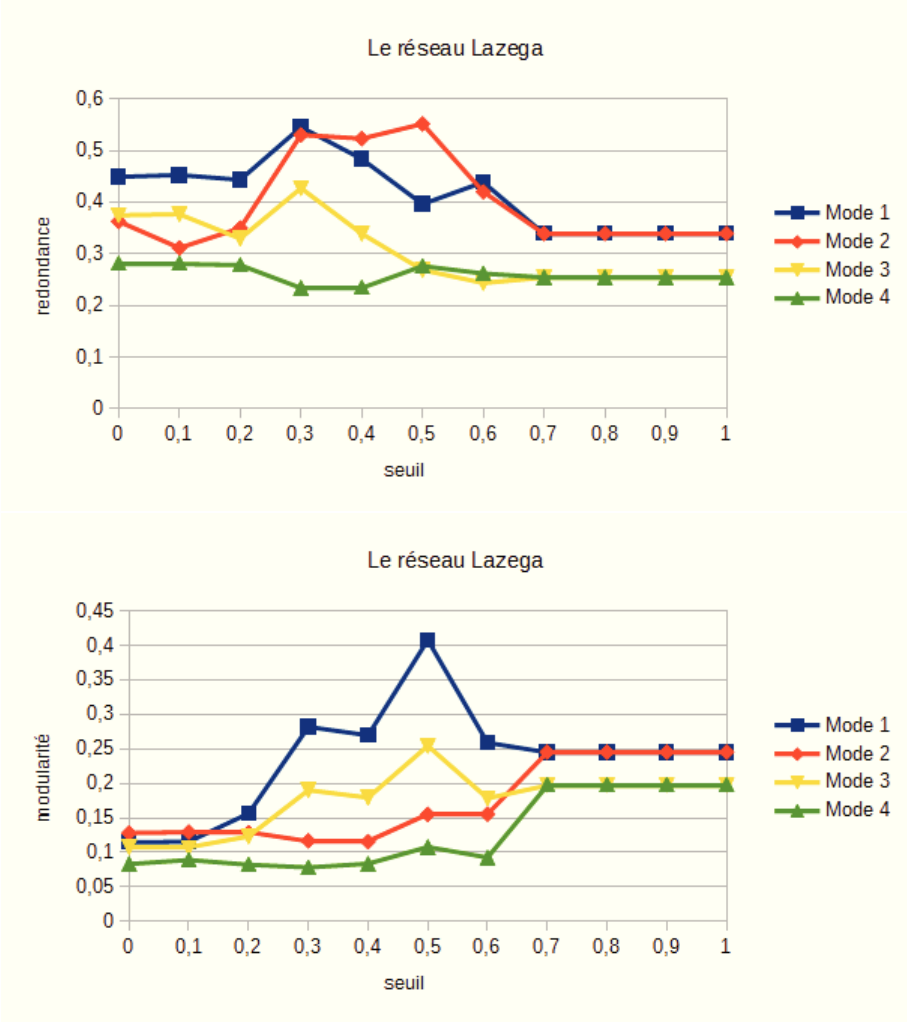

FIGURE 4.4 – Étude des paramètres de Mux-Licod sur le réseau Lazega

Classement des modes en terme de modularité: les mêmes résultats se reproduisent en terme de modularité, nous remarquons que les mode1 et mode2 sont meilleurs avec un *seuil* = 0.5. Dans la section 4.3.3, nous nous limitons à ces deux modes pour évaluer et comparer Mux-Licod par rapport aux autres approches existantes.

- L1=[mode1, mode2, mode3, mode4]
- L2=[mode1, mode2, mode4, mode3]
- L3=[mode2, mode4, mode3, mode1]

- Résultat avec borda: mode1=6, mode2=5, mode3=10, mode4=9
- Résultat final=[**mode2, mode1**, mode4, mode3]

4.3.3 Étude comparative

Afin d'analyser les performances de notre approche, nous la comparons aux méthodes existantes de l'agrégation de couches, d'ensemble clustering et d'exploration simultanée des couches appelé *GenLouvain* [Mucha *et al.* 2010].

4.3.3.1 Description des différents algorithmes: paramétrage

Pour chacun des algorithmes que nous avons utilisés pour l'étude comparative des différentes méthodes, nous décrivons les paramètres utilisés pour les exécuter. Le préfixe `AC` (reps. `EC`, `ESC`) désigne l'approche d'agrégation des couches (reps. ensemble clustering, Exploration simultanée des couches).

- *Mux-Licod.* L'étude que nous avons effectuée sur le paramétrage de Mux-Licod dans la section 4.3.2 montre que ce dernier est plus performant dans le *mode*1 et le *mode*2. Pour cette raison, nous nous limitons dans cette comparaison à ces deux modes. Ici nous avons utilisé comme méthode de fusion des listes de vote Borda introduite dans [Sculley 2007]. Pour le calcul des voisins d'un nœud nous avons fixé le seuil de Jaccard: $\alpha_{jaccard} = 0.5$.
- *EC.* Nous avons utilisé l'algorithme de fusion proposé dans [Seifi 2012] pour fusionner les partitions issues des différents algorithmes que nous avons testés (Louvain [Blondel *et al.* 2008], Licod classique [Yakoubi et Kanawati 2014], Walktrap [Pons et Latapy

2005], Infomap [Rosvall *et al.* 2009], EdgeBetweenness [Girvan et Newman 2002]) avec une valeur de seuil permettant de connecter deux sommets: $\alpha = 0,8$.

– *AC.* Nous utilisons la méthode d'union pour agréger les couches afin de pouvoir appliquer par la suite un algorithme de détection des communautés (Louvain, Licod classique, Walktrap, Infomap, EdgeBetweenness).

4.3.3.2 Critères d'évaluation

Comme nous l'avons vu dans le chapitre 3, très peu de travaux ont abordé cette épineuse question dans le cas des graphes multiplexes. A notre connaissance, nous n'avons pas de graphes réels ni de générateurs de graphes artificiels qui peuvent nous donner des réseaux multiplexes de Benchmark. Au niveau des indicateurs topologiques, nous utilisons deux mesures: *la modularité multiplexe* permettant de mesurer la qualité de nos partitions, et *la mesure de redondance* qui est un indicateur de qualité topologique d'une communauté dans le réseau multiplexe proposé par [Berlingerio *et al.* 2011].

1. ***La modularité multiplexe:*** comme nous l'avons vu dans la section 3.3.4 de chapitre 3, cette métrique est utilisée pour mesurer la proportion des liens internes aux communautés et la même quantité dans un modèle nul où aucune structure communautaire n'est attendue.
2. ***La mesure de redondance:*** la redondance calcule la moyenne de la redondance de chaque lien intra communauté dans toutes les couches de multiplexes. L'intuition est que les liens intra communautés doivent être des liens récurrents dans les différentes couches (voir section 3.3.4 de chapitre 3).

4.3.3.3 Résultats

Résultats sur les petits réseaux: Sur les trois premiers réseaux, nous avons comparé Mux-Licod avec les deux types d'approches de base d'agrégation de couches et d'agrégation des partitions en utilisant différents algorithmes de base reconnus dans l'état de l'art à savoir: Licod classique [Yakoubi et Kanawati 2014], Edge Betweenness [Girvan et Newman 2002], Walktrap [Pons et Latapy 2005], Louvain [Blondel *et al.* 2008] et Infomap

[Rosvall *et al.* 2009]. En plus, nous avons comparé Mux-Licod par rapport à une approche appartenant à la même classe d'exploration simultanée des couches: GenLouvain.

Les figures 4.5, 4.6 et 4.7 montrent les performances des différentes approches sur les trois petits réseaux, en matière de redondance.

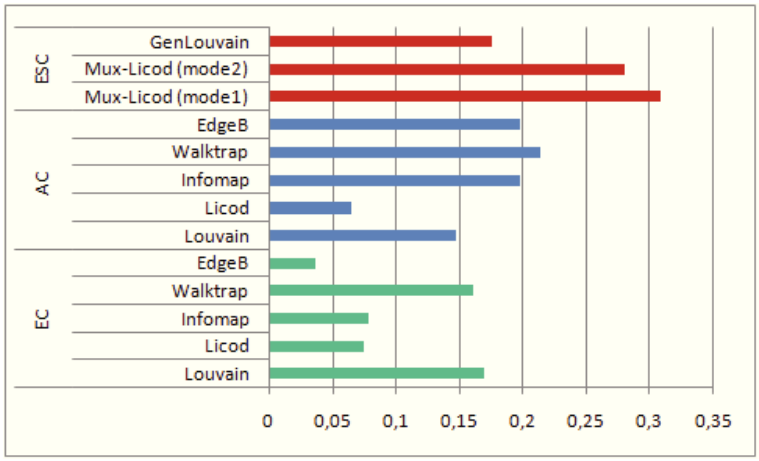

FIGURE 4.5 – Mesure de redondance sur le réseau de CKM

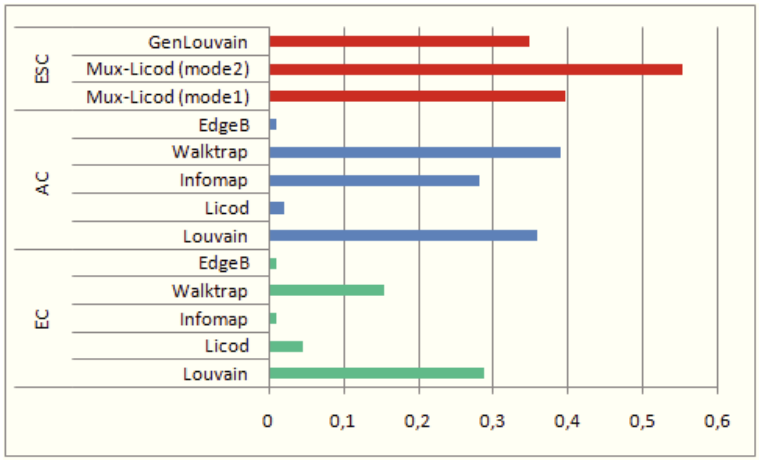

FIGURE 4.6 – Mesure de redondance sur le réseau Lazega

Les figures 4.8, 4.9 et 4.10 montrent les performances des différentes approches sur les trois petits réseaux, en terme de modularité multiplexe.

En terme de redondance, les approches d'exploration simultanée des couches (ESC ou d'extension) comme Mux-Licod et GenLouvain ont de meilleures performances que les autres. En comparant les deux approches d'extension entre elles, nous remarquons que

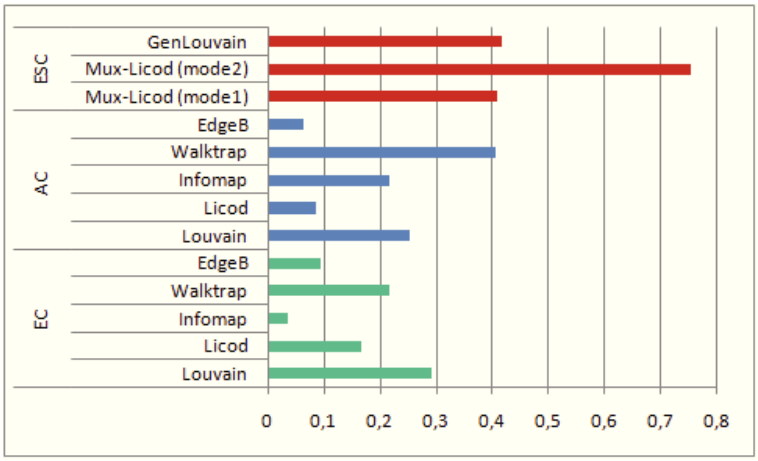

FIGURE 4.7 – Mesure de redondance sur le réseau des Vickers

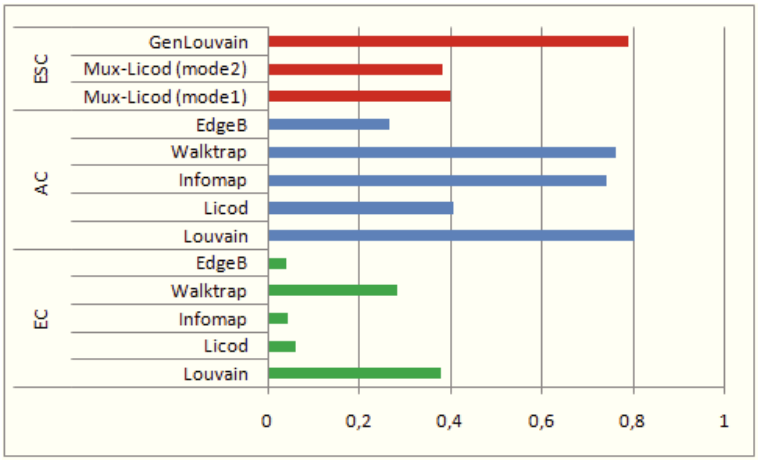

FIGURE 4.8 – Mesure de modularité sur le réseau CKM

Mux-Licod surpasse GenLouvain, quelque soit le mode utilisé (1 ou 2).

Dans les petits réseaux, Mux-Licod a surpassé les autres types d'approches en terme de qualité de partitionnement. En effet, les liens intra communautés sont plus redondants dans les différentes couches du multiplexe avec notre approche que ceux retrouvés avec les deux approches de base.

En terme de modularité, notre approche est un peu moins compétitive que l'algorithme de Louvain [Blondel *et al.* 2008], basé sur l'optimisation de la modularité. En prenant en compte les deux critères d'évaluation, notre approche se situe clairement dans le front Pareto.

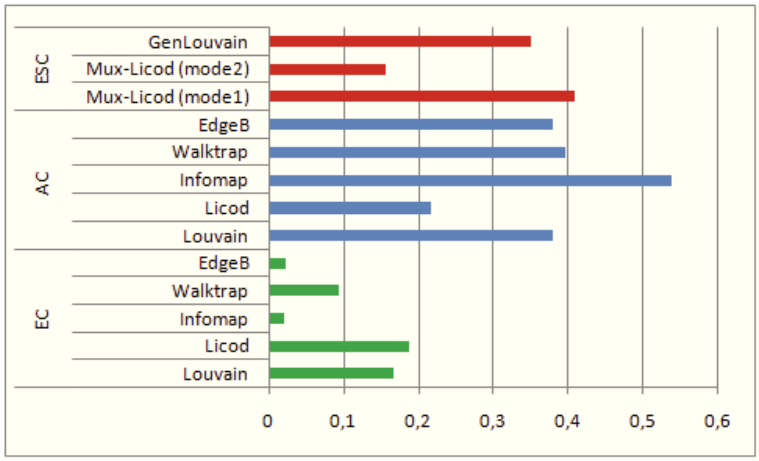

FIGURE 4.9 – Mesure de modularité sur le réseau Lazega

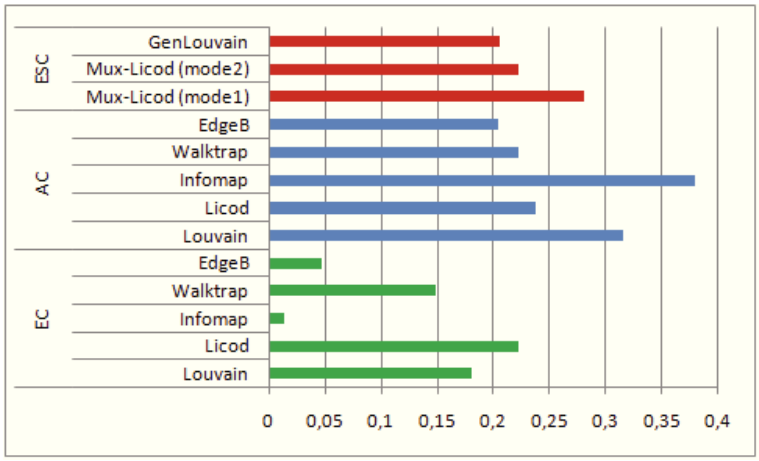

FIGURE 4.10 – Mesure de modularité sur le réseau Vickers

Résultats sur les grands réseaux Dblp: Après avoir comparé Mux-Licod par rapport aux différentes approches de clustering sur les petits réseaux, nous l'évaluons sur des données de grande taille: Dblp en fonction des mesures de modularité et de redondance.

Nous remarquons que les résultats se confirment sur les grands réseaux. Dans la classe des approches d'extension, on voit qu'il y a un effondrement des résultats de GenLouvain. Dans les grands réseaux, nous constatons la présence d'une relation inverse entre la redondance et la modularité. Cela nécessite une confirmation sur d'autres bases.

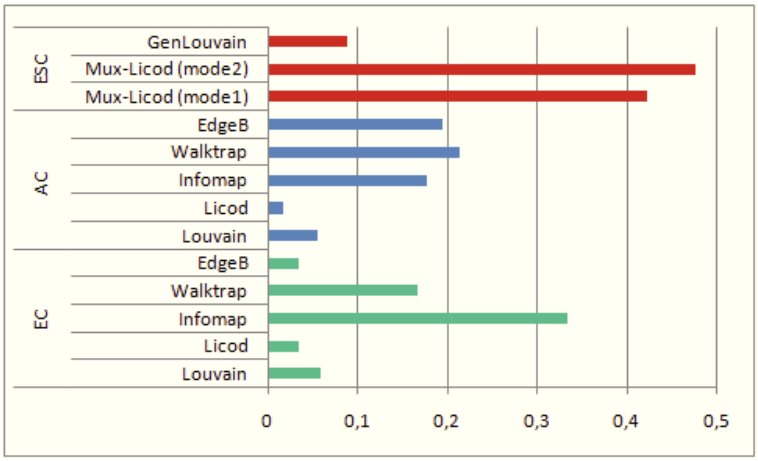

FIGURE 4.11 – Dblp: mesure de redondance

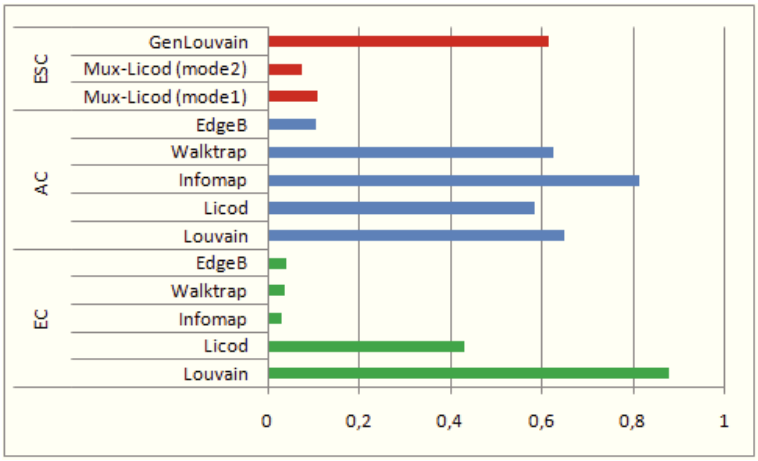

FIGURE 4.12 – Dblp: mesure de modularité multiplexe

4.4 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche de détection de communautés centrée graine prenant en compte les différents types de relations entre les noeuds de différentes couches dans le réseau mutiplexe. Les expérimentations montrent que l'algorithme Mux-Licod permet d'avoir un bon clustering par rapport aux autres approches en fonction des mesures topologiques. Dans le chapitre suivant, nous testons les différentes approches de clustering sur la tâche de recommandation.

Chapitre 5

Notre approche de recommandation de tags

5.1 Introduction

Différentes approches topologiques pour la recommandation de tags ([Rae *et al.* 2010], [Jäschke *et al.* 2008]) ont été abordées dans le chapitre 2, celles-ci souffrent d'un problème de complexité de calcul. Cette complexité provient du fait que le calcul de la recommandation des tags pour la plupart des approches existantes repose sur le graphe entier représentant la folksonomie tout en s'affranchissant de l'application des prétraitements par exemple la compression de taille ou l'élimination des données inutiles pour la recommandation des tags. Pour pallier ces problèmes, nous proposons dans ce chapitre une nouvelle approche de recommandation de tags personnalisée basée sur la réduction des données. En effet, cette approche permet de réduire la complexité de calcul en réduisant la taille de toutes les composantes de graphe initial qui la caractérisent (ressources, tags et utilisateurs). L'approche est fondée principalement sur deux étapes (voir figure 5.1):

– *Étape1: Réduction du graphe:* cette étape vise à réduire la taille du graphe initial par la génération des clusters en utilisant la technique de détection de communautés. De plus, nous proposons d'améliorer la qualité de nos partitions en ayant recours à la transformation de l'hypergraphe représentant la folksonomie en un réseau multiplexe. Le réseau multiplexe permet de garder plus d'informations sur les nœuds et sur leurs différents types de liens sous forme de couches. Cette nouvelle représentation requiert

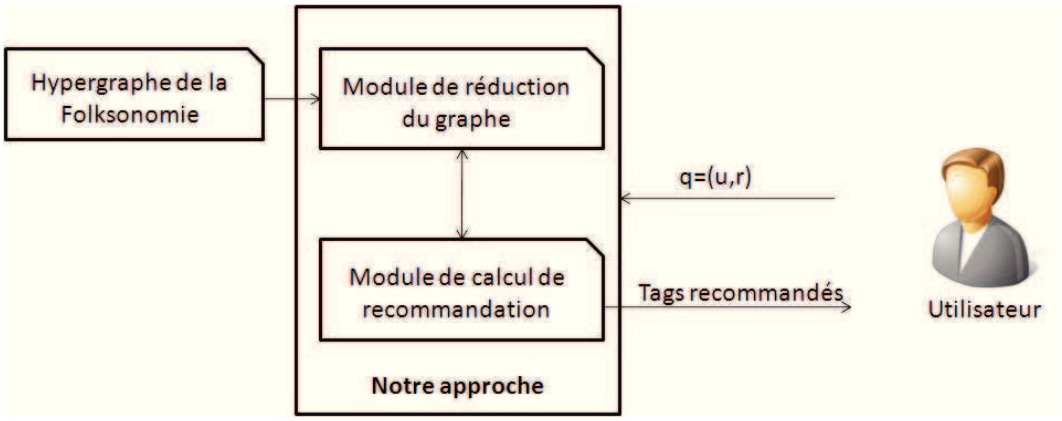

FIGURE 5.1 – Principales composantes de notre approche

la redéfinition de la plupart des algorithmes existants dédiés à l'analyse de réseaux.

– *Étape2: Calcul de recommandation:* dans le chapitre 2.1, nous avons vu les algorithmes topologiques performants comme *Folkrank*. Ici, nous proposons une approche de recommandation topologique dite *par niveaux* basée sur Folkrank en intégrant une phase de réduction de graphe permettant de réduire la complexité. Tout d'abord, nous réduisons le graphe initial sur lequel nous appliquons Folkrank afin de sélectionner les clusters de tags les plus appropriés. Ces clusters sont ensuite utilisés pour construire un autre graphe contextuel de taille beaucoup plus réduite extrait du graphe original représentant la folksonomie. La méthode Folkrank est à nouveau appliquée afin de calculer la liste de tags à recommander. La phase de réduction de graphe assure le filtrage des tags par l'élimination dès le départ des tags non intéressants par rapport au contexte de l'utilisateur. Par conséquent, la quantité des données traitée est nettement moins importante que celle des données de départ.

Ce chapitre suit le plan suivant: notre approche est décrite dans la section 5.2 dont nous introduisons le modèle de l'approche ainsi que les différentes étapes de recommandation de tags permettant d'assister l'utilisateur à l'annotation des différentes ressources.

À la fin de ce chapitre, nous présentons le protocole expérimental pour la validation de l'approche ainsi que les différents résultats obtenus.

5.2 Approche de recommandation de tags par niveaux: TLTR

5.2.1 Description informelle

L'approche de recommandation se fait à deux *niveaux*. Les grandes lignes de notre approche sont:

1. *Réduction des graphes:* consiste à compresser le graphe de la folksonomie. Ce processus se fait en deux phases:
   - Phase de génération des clusters: l'identification des clusters de chaque type de nœuds est assurée par un algorithme de détection des communautés où les nœuds représentent les clusters (clusters des utilisateurs, clusters des tags et clusters des ressources). Afin de pouvoir assurer ce clustering, il fallait appliquer un ensemble des prétraitements de projection de graphe tripartite en graphe unipartite en passant par la projection bipartite.
   - Phase de génération de graphe réduit: chaque cluster obtenu dans la première étape correspond à un nœud dans le nouveau graphe réduit. Les liens entre les différents types de nœuds se construisent de la manière suivante: un lien est créé entre deux clusters s'il y a déjà au moins un lien entre deux éléments appartenant à ces deux clusters dans le graphe initial de la folksonomie.
2. *Traitement de la requête:* consiste à transformer la requête initiale de l'utilisateur pour qu'elle puisse interroger le nouveau graphe réduit.
3. *Sélection des clusters tags:* consiste à sélectionner les K_c clusters de tags les plus pertinents par rapport à la requête de l'utilisateur en utilisant l'algorithme de Folkrank [Jäschke *et al.* 2008].
4. *Extraction d'un graphe contextuel:* consiste à extraire un sous-graphe personnalisé (contextuel) pour chaque requête de l'utilisateur. Ce graphe est composé par les nœuds présents dans les clusters activés dans l'étape précédente. Ces nœuds sont étendus afin d'intégrer les utilisateurs similaires, les ressources similaires et des tags intéressants. Le graphe contextuel est alors un graphe réduit de petite taille.
5. *Recommandation de tags:* consiste à recommander les K_t les plus pertinents par rapport à la requête initiale de l'utilisateur à partir du sous-graphe contextuel. De la

même façon que l'étape 3, cette étape est assurée par l'algorithme Folkrank [Jäschke *et al.* 2008].

Le modèle TLTR présenté dans la figure 5.2 est composé de deux principaux modules: (1) Module de compression (algorithme 6) et (2) Module de recommandation (algorithme 5). Le module de compression a été détaillé dans le chapitre 4. Le module de recommandation est fondé sur cinq principales étapes. Nous détaillons ces étapes dans les sous-sections suivantes.

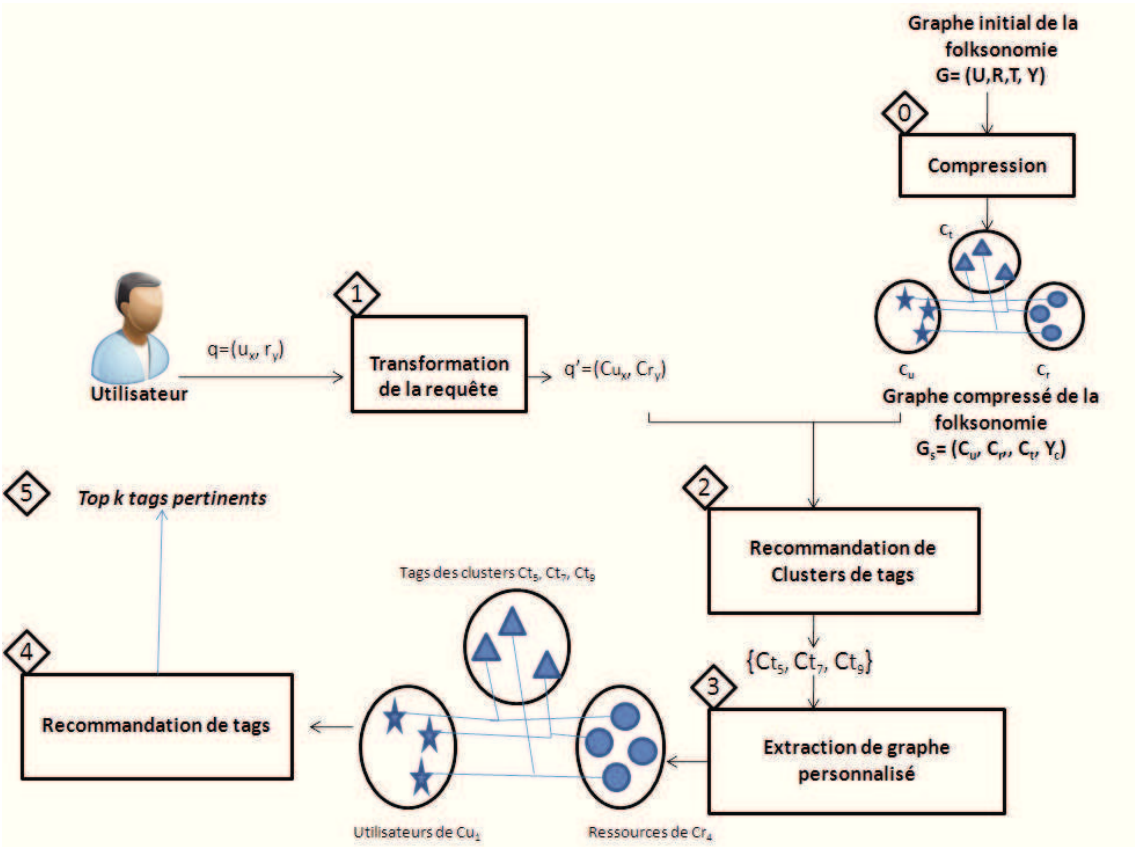

FIGURE 5.2 – Schéma décrivant le modèle TLTR

5.2.2 Réduction des graphes

Pour réduire toutes les composantes du graphe tripartite de la folksonomie, nous avons besoin d'une fonction de similarité topologique afin de regrouper les éléments appartenant au même contexte dans le même cluster. Nous obtenons deux niveaux pour chacune des

Algorithm 5 Algorithme TLTR

Require: $G =< (U, R, T), E >$ le graphe de la folksonomie,
$G_c =< (C_U, C_R, C_T), E_c >=$ Compression(G),
$q = (u, r)$: requête utilisateur
Ensure: K_c: les clusters de tags à sélectionner,
K_t: les tags à recommander
1: $q' \leftarrow Generalize(q)$ /* $q' = (C_u, C_r) : u \in C_u$ *et* $r \in C_r$ */
2: $K_c \leftarrow Folkrank(G_c, q')$ /* $G_c =< (C_U, C_R, C_T), E_c >= Compression(G)$ */
3: $G_s \leftarrow Graph - contextual(U_s, R_s, T_s)$ /* *avec* $U_s = \{u \in C_u\}$, $R_s = \{r \in C_r\}$ *et* $T_s = \{t \in K_c\}$ */
4: $K_t \leftarrow Folkrank(G_s, q)$
5: **return** K_t

Algorithm 6 Algorithme de compression

Require: $G =< (U, R, T), E >$ le graphe de la folksonomie.
$G_T = construct\_multiplex(G, T)$
$G_R = construct - multiplex(G, R)$
$G_U = construct - multiplex(G, U)$
1: $C_T \leftarrow community(G_T)$
2: $C_R \leftarrow community(G_R)$
3: $C_U \leftarrow community(G_U)$
4: $G_c \leftarrow compress(C_T, C_R, C_U)$
5: **return** G_c

composantes de la folksonomie: le niveau générique construit par des clusters et le niveau spécifique composé par les éléments de la folksonomie de départ. Pour cela, nous faisons appel à la technique de clustering. L'application de clustering ne se fait que sur les graphes unipartites. Or, le graphe initial de la folksonomie est tripartite, nous passons par un ensemble des prétraitements afin d'assurer la projection de graphe tripartite en graphe unipartite en passant par la projection bipartite. À la fin de ces prétraitements, nous obtenons six graphes unipartites où chacun des deux graphes sont composés par le même type et nombre de nœuds (Utilisateurs, Tags, Ressources) et dont les liens représentent deux types de relations différentes. Par exemple, la composante tag est définie par deux graphes (couches) où les liens de premier représentent les ressources (deux tags sont liés s'ils ont été utilisés pour annoter la même ressource), et les utilisateurs pour le deuxième (deux tags sont liés s'ils ont été utilisés par le même utilisateur). Ce clustering nous permettra de faciliter et de diminuer l'espace de recherche des tags à recommander.

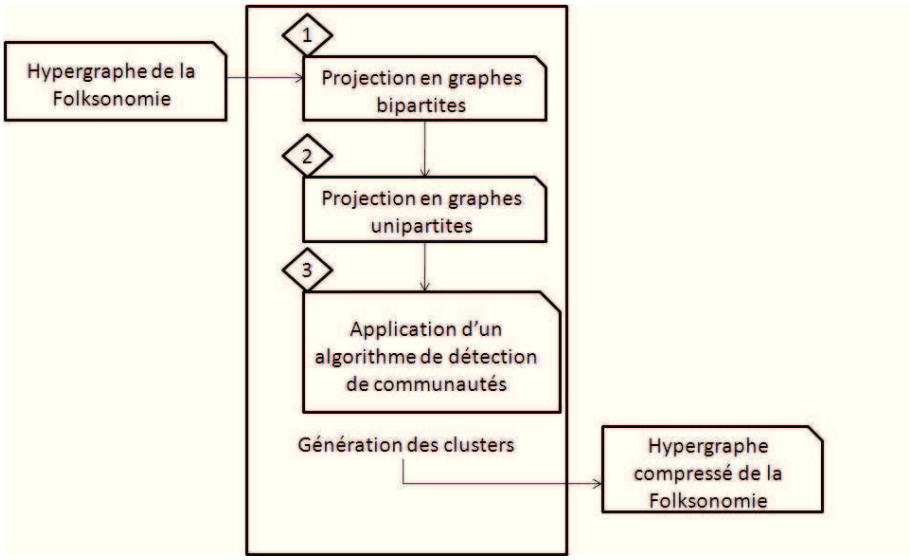

FIGURE 5.3 – Les différentes étapes de réduction de la folksonomie

Comme nous pouvons le constater dans la figure 5.3, le résultat de ce processus génère un graphe tripartite compressé de taille beaucoup plus réduite constitué par des nœuds où chaque nœud correspond à un cluster. Les clusters correspondent aux tags, ressources ou à des utilisateurs.

5.2.2.1 Les prétraitements

Projection en graphes bipartites: Afin de pouvoir manipuler et appliquer les algorithmes de détection de communautés sur le graphe tripartite de la folksonomie, il est indispensable de transformer ce dernier en un graphe unipartite. Pour y parvenir, nous passons par la transformation du graphe initial en graphe bipartite [Mika 2005]. Ces graphes bipartites connus aussi sous le nom *graphes d'association*, entre les utilisateurs et les ressources (UR), les utilisateurs et les tags (UT) et les ressources et les tags (RT) voir 5.4. Une représentation formelle des graphes bipartites est proposée comme suit:

- $G_{RT} = \{R \times T, E_{rt}\}, E_{rt} = \{(r,t) | \exists u \in U : (u,r,t) \in E\}$.
- $G_{UT} = \{U \times T, E_{ut}\}, E_{ut} = \{(u,t) | \exists r \in R : (u,r,t) \in E\}$.
- $G_{RU} = \{R \times U, E_{ru}\}, E_{ru} = \{(r,u) | \exists t \in T : (u,r,t) \in E\}$.

Par exemple, la génération des clusters de tags nécessite de produire deux graphes bipartites contenant les nœuds tags issus de deux différents points de vue: le point de vue ressource

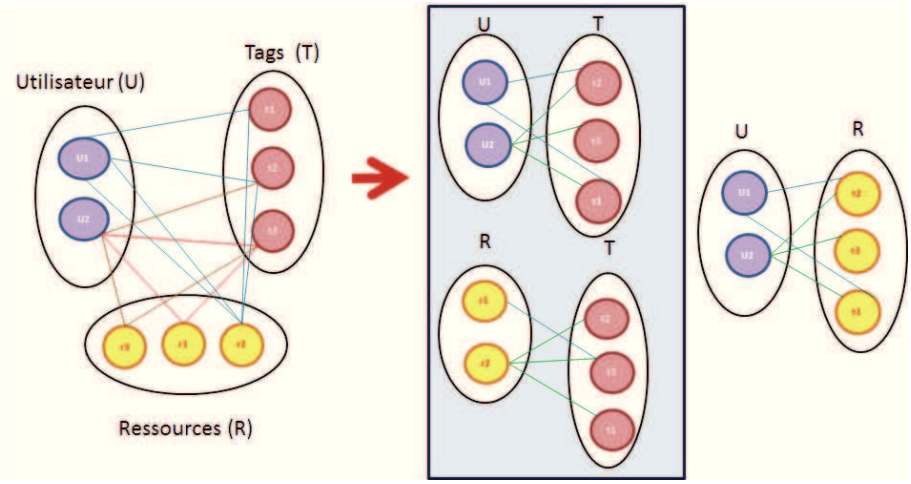

FIGURE 5.4 – Projection du graphe tripartite en trois graphes bipartites

et le point de vue tag (G_{RT} et G_{UT}), qui sont encadrés dans la figure 5.4. De même pour les autres clusters.

Projection en graphes unipartites: Nous rappelons que le but est de générer les clusters de tags, des ressources et des utilisateurs à partir du graphe initial. Pour la génération des clusters de tags par exemple, nous avons besoin principalement des deux graphes bipartites contenant l'ensemble des tags T. Pour chacun de ces deux graphes, nous appliquons la projection suivante:

- $G_T^R = \{T, E_{tt}\}, E_{tt} = \{(t_i, t_j) \in T \times T | \exists r \in R : (r, t_i), (r, t_j) \in E_{rt}\}.$
- $G_T^U = \{T, E_{tt}\}, E_{tt} = \{(t_i, t_j) \in T \times T | \exists u \in U : (u, t_i), (u, t_j) \in E_{ut}\}.$

Dans la figure 5.5, nous constatons que lorsque deux tags sont utilisés pour annoter la même ressource, un lien est créé entre eux dans le graphe G_T^R. Tandis que, dans le cas du graphe G_T^U, un lien est rajouté entre deux tags s'ils ont été utilisés par le même utilisateur pour annoter une ressource (deuxième graphe dans la figure 5.5). Nous répétons les mêmes étapes pour générer les graphes unipartites des utilisateurs et des ressources en remplaçant à chaque fois les tags, respectivement par les utilisateurs et par les ressources. À l'issue de cette étape, nous obtenons au total six graphes unipartites dont chaque paire représente un réseau multiplexe. Par exemple dans la figure 5.6, le réseau multiplexe de tags est composé de deux couches qui correspondent respectivement aux graphes G_T^R et G_T^U.

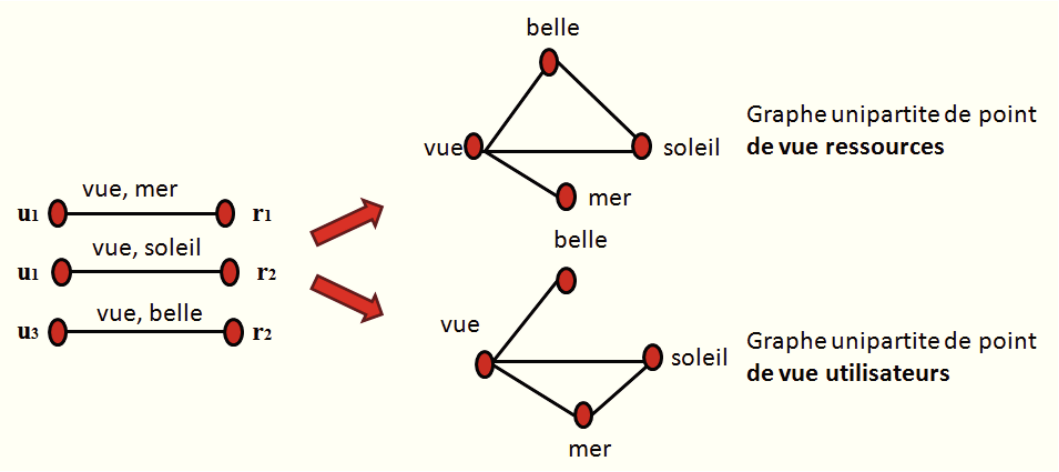

FIGURE 5.5 – Exemple de projection des graphes bipartites en graphes unipartites

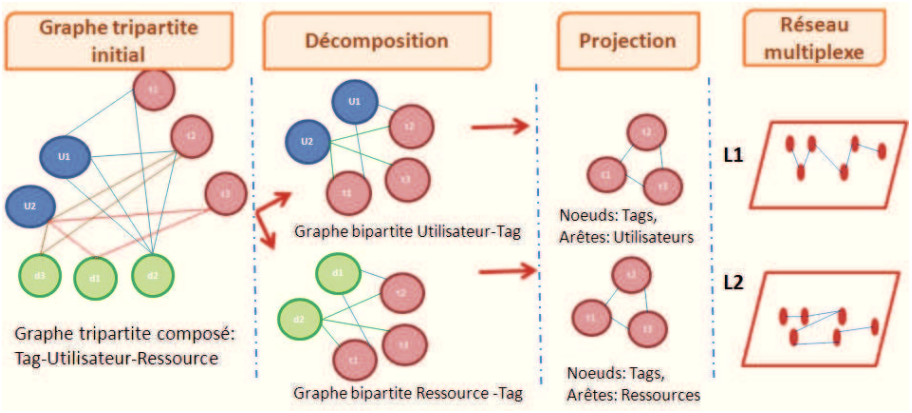

FIGURE 5.6 – Le réseau multiplexe des tags: les étapes de transformation

Une fois que nous avons fini la projection des graphes, nous obtenons trois réseaux multiplexes.

5.2.2.2 Compression de graphe

À ce niveau, le but est de réduire le graphe initial représentant la folksonomie. Pour cela, nous avons besoin principalement:

1. Un réseau multiplexe: un graphe dont la structure est multicouche que nous obtenons à l'issue de l'étape des prétraitements présentée dans la section 5.2.2.1.
2. Un algorithme de détection des communautés: cet algorithme permet de détecter les communautés dans les graphes multiplexes. Nous évaluons dans ce travail les

différents algorithmes présentés dans la section 3.3 du chapitre 3.

Après avoir appliqué un algorithme de détection de communautés sur le réseau multiplexe, chaque communauté (cluster) obtenue correspond à un nœud dans le nouveau graphe réduit. Les liens entre les différents types de nœuds se construisent de la manière suivante: un lien est créé entre deux clusters s'il y a déjà au moins un lien entre deux éléments appartenant à ces deux clusters dans le graphe initial de la folksonomie. Nous prenons un exemple pour illustrer la génération des liens non pondérés dans la figure 5.7. Après avoir appliqué un algorithme de détection des communautés, nous obtenons 1 cluster utilisateur, 2 clusters ressources et 1 cluster tag.

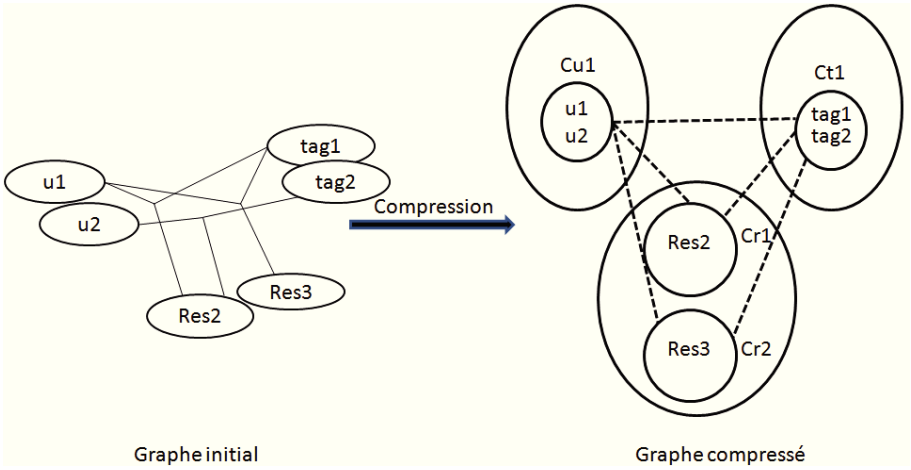

FIGURE 5.7 – Exemple de compression de graphe de la folksonomie

5.2.3 Traitement de la requête

Après la phase de clustering du graphe initial de la folksonomie, nous avons obtenu un graphe compressé composé par des clusters de tags, des clusters des utilisateurs et des clusters des ressources. De ce fait, la première étape consiste à transformer la requête de l'utilisateur $q = (u_x, r_y)$ par des clusters auxquels l'utilisateur et la ressource de la requête initiale appartiennent. C'est-à-dire à ce stade, nous allons nous intéresser uniquement au niveau cluster. Par exemple, si l'utilisateur $u_x \in C_{u_x}$ et la ressource $r_y \in C_{r_y}$ alors la nouvelle requête devient: $q' = (C_{u_x}, C_{r_y})$. Nous nous basons sur l'hypothèse que les requêtes utilisateur (utilisateurs et ressources) existent déjà dans le système. Par conséquent, notre

TLTR souffre de problème de démarrage à froid.

5.2.4 Sélection des clusters de tags

Le but de cette étape est de sélectionner les K_c les plus pertinents par rapport à la requête transformée q' dans la deuxième étape de l'approche. Le K_c est un paramètre clé de l'approche, car il a un impact direct sur la taille du graphe contextuel réduit dont nous faisons l'extraction. Nous sélectionnons les $top - K_c$ pour extraire un graphe, personnalisé à chaque requête, appelé graphe contextuel. Dans ce travail, nous faisons une étude de l'influence de K_c sur la recommandation. Le choix de ces clusters est assuré par l'algorithme topologique Folkrank [Jäschke *et al.* 2008].

5.2.5 Extraction d'un graphe contextuel

À cette étape, l'objectif est de revenir à la structure initiale du graphe où chaque nœud représente soit un utilisateur, soit une ressource soit un tag. L'idée est de proposer un graphe contextuel pour chaque requête dans lequel nous réduisons au maximum la quantité des éléments inutiles. En effet, si un utilisateur est intéressé par exemple par le thème informatique c'est inutile de parcourir les éléments du thème agriculture. Nous utilisons pour cela le graphe réduit obtenu lors de la section 5.2.3.

Donc, à partir du graphe initial $G := (U, T, R)$ avec: $U, TetR$ sont des ensembles finis respectivement des utilisateurs, des tags et des ressources, nous allons extraire un sous-graphe contextuel $G_s := (U_s, T_s, R_s)$ par rapport à chaque requête utilisateur. Cette extraction est basée sur l'utilisateur posant la requête ainsi que sur la ressource qu'il veut annoter. En effet, elle consiste à extraire les clusters C_{u_x} et C_{r_y} contenant l'utilisateur et la ressource de la requête q. Les tags T_s sont formés par l'ensemble des tags appartenant aux clusters $CTR(q')$.

5.2.6 Recommandation de tags

Cette étape est similaire à l'étape de recommandation des clusters sauf que cette fois-ci nous appliquons la recommandation sur le graphe contextuel par rapport à la requête q avec le même algorithme Folkrank utilisé dans la section 5.2.4. Le résultat est l'ensemble

des tags à recommander. Nous distinguons deux points d'originalité dans notre approche. Le premier point se manifeste dans le processus de la recommandation qui se fait sur deux niveaux: à un niveau générique (cluster) en premier temps puis un niveau spécifique (tag) en deuxième temps. Le deuxième point est que nous proposons un graphe personnalisé à chaque requête. Cela a pour objectif d'améliorer la précision dans la recommandation.

5.2.7 Étude de la complexité

Nous comparons la complexité de notre méthode TLTR avec la méthode de recommandation de tags Folkrank. L'approche TLTR est fondée principalement sur le clustering et sur la recommandation. Nous prenons comme exemple l'approche TLTR basée sur la méthode de clustering Mux-Licod.

Le calcul de la complexité de TLTR est fondé sur: la complexité de Mux-Licod, la complexité de recommandation de clusters et la complexité de recommandation de tags.

La complexité de l'algorithme Mux-Licod dépend de la complexité des trois étapes fondamentales de calcul des leaders, de calcul de degré d'appartenance communautaire et la méthode d'agrégation des préférences appliquées. L'utilisation de la centralité de degré pour l'identification des nœuds leaders requiert $\mathcal{O}(m)$ étapes où $m = \max_{i \in [1,\alpha]} |E_i|$ est le nombre maximum des arêtes dans les couches du multiplexe. L'étape la plus coûteuse est le calcul des préférences d'appartenance des nœuds aux communautés identifiées qui est basé sur l'algorithme de Dijkstra [Dijkstra 1959] pour le calcul du plus court chemin entre chaque nœud et les nœuds leaders identifiés. En appliquant l'algorithme plus court chemin, la complexité est $\mathcal{O}(m + |V|log(|V|)$. L'étape de fusion des préférences avec l'algorithme Borda a une complexité $\mathcal{O}(l \times log(l))$ où $l = |C|$ est le nombre des leaders identifiés. Actuellement, la procédure de fusion locale des rangs est simplement mise en œuvre en utilisant l'algorithme de tri rapide comme détaillé dans [Dwork *et al.* 2001]. Cependant, le nombre des leaders identifiés est généralement très faible par rapport au nombre total des nœuds. Nous signalons que le clustering est exécuté une fois hors ligne. En conséquence, la complexité de l'algorithme de clustering est ignorée et la complexité de TLTR se limite à la complexité de Folkrank exécuté sur le graphe réduit et sur le graphe contextuel. Pour résumer, la complexité de la tâche de recommandation est la somme

respectivement de la complexité de recommandation des clusters sur le graphe réduit $\mathcal{O}(iter \times (|A| + |U| + |R| + |T|) + |T| \times N)_{clusters}$ et la recommandation de tags effectuée sur le graphe contextuel $\mathcal{O}(iter \times (|A| + |U| + |R| + |T|) + |T| \times N)_{contextuel}$. Donc, la complexité de TLTR est:
$\mathcal{O}(iter \times (|A| + |U| + |R| + |T|) + |T| \times N)_{clusters} + \mathcal{O}(iter \times (|A| + |U| + |R| + |T|) + |T| \times N)_{contextuel}$.
[Jäschke *et al.* 2008] ont prouvé que la complexité d'exécution des top-N recommandations avec l'algorithme Folkrank est $\mathcal{O}(iter \times (|A| + |U| + |R| + |T|) + |T| \times N)$ où iter est le nombre d'itérations. $|A|$, $|U|$, $|R|$, $|T|$ sont respectivement le nombre des arêtes, des utilisateurs, des ressources et des tags.

Le tableau 5.1 présente une comparaison des différentes tailles de graphe initial de la folksonomie, de graphe réduit (par exemple avec Mux-Licod) ainsi que de graphe contextuel. Pour estimer la taille de graphe contextuel, nous avons calculé la moyenne des graphes contextuels provenant d'environ 509 requêtes. Le grand graphe de la folksonomie est utilisé en entier pour calculer la recommandation dans l'approche de Folkrank classique. Tandis qu'avec TLTR, nous calculons la recommandation une fois sur le graphe réduit et une autre fois sur le graphe contextuel de petite taille. Plus le nombre d'itérations d'exécution de Folkrank augmente, plus la complexité de TLTR est moins importante que celle de Folkrank classique.

Graphes	**#Nœuds**	**#Arêtes**	**#U**	**#T**	**#R**
G	889	24297	116	412	361
G_c(graphe réduit)	434	1677	97	154	183
Taux compression en %	51,18	93,1	16,37	62,62	49,30
$Moyenne(G_s)$(graphe contextuel)	36	89	9	17	10
Taux compression en %	95,95	99,63	92,24	95,87	97,22

TABLE 5.1 – Taux de compression des deux graphes (compressé et contextuel)

5.3 Expérimentations et résultats

Les expérimentations que nous menons présentent une étude comparative des différentes méthodes de recommandation de tags. Nous évaluons pour cela les performances de notre système par niveaux $TLTR$ en comparant les différents algorithmes de détection de communautés présentés dans le chapitre 3 sur les données de Bibsonomy en terme de mesure de *précision*.

5.3.1 Le jeu de données

Bibsonomy est un système de partage des références bibliographiques [Jäschke *et al.* 2008]. Le tableau 5.2 présente une description de ces données: Après avoir effectué les

# **Utilisateurs**	# **Tags**	# **Ressources**	# **Arêtes**
116	412	361	24297

TABLE 5.2 – Les données de Bibsonomy

prétraitements sur les données de Bibsonomy décrits dans la section 5.2.2.1, nous obtenons trois réseaux multiplexes. La table 5.3 décrit en détail les différentes données de chacun des réseaux multiplexes de Bibsonomy.

Réseau Multiplexe	**Couches**	**Nœuds**	**Arêtes**	**Densité**
Utilisateur	*Utilisateur basé Ressource*	116	901	0,135
	Utilisateur basé Tag	116	985	0,147
Tag	*Tag basé Ressource*	412	2496	0,0294
	Tag basé Utilisateur	412	1956	0,0231
Ressource	*Ressource basé Tag*	361	2814	0,0433
	Ressource basé Utilisateur	361	1685	0,0259

TABLE 5.3 – Les réseaux multiplexes de Bibsonomy

Nous testons ici notre approche de recommandation avec les différentes méthodes de clustering. Comme algorithme de recommandation, nous avons choisi Folkrank [Hotho *et al.* 2006]. Ce choix est justifié par la popularité et l'efficacité de cet algorithme. Nous comparons notre système de recommandation par niveau TLTR en utilisant les différentes approches de clustering présentées dans l'état de l'art des graphes multiplexes telles que: les approches

d'exploration simultanée des couches, l'agrégation des couches et l'ensemble clustering par rapport à l'algorithme Folkrank classique [Hotho *et al.* 2006].

5.3.2 Les paramètres de l'approche et méthodologie

Nous présentons dans cette section les trois principaux paramètres de l'approche:

- L'algorithme de détection de communauté: le choix de l'algorithme est assez important dans notre méthodologie. En effet, l'étape de clustering constitue l'un des piliers de notre approche (section 5.2.2). Pour cela, nous avons étudié les différentes approches de clustering dans les réseaux multiplexes: AC, EC et ESC (détaillées dans le chapitre 3).
- Le nombre des clusters à sélectionner K_c: ce paramètre détermine le nombre de clusters à sélectionner pour l'extraction du graphe contextuel. Nous ne fixons pas le paramètre K_c mais au contraire nous étudions l'intégralité des valeurs possibles.
- Le nombre de tags à recommander K_t: afin de déterminer le nombre de tags à recommander à l'utilisateur, nous avons fait une analyse des données de Bibsonomy pour limiter l'intervalle d'étude du nombre de tags à recommander. L'histogramme 5.8 montre que la plupart des ressources sont annotées par des tags dont le nombre est entre 1 et 4 tags. De ce fait, nous menons nos expérimentations de recommandation de tags (section 5.3.4) dans cet intervalle.

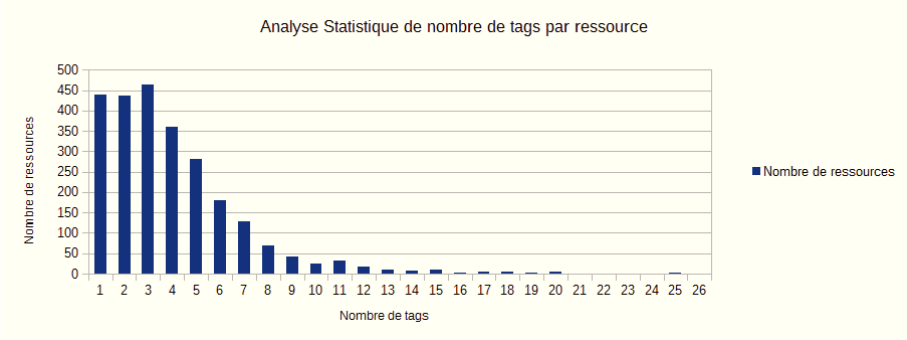

FIGURE 5.8 – Bibsonomy: étude statistique de nombre de tags par ressource

Notre objectif est d'évaluer l'impact de ces trois paramètres sur la performance de la recommandation. Par ailleurs, nous fixons les paramètres de l'algorithme Folkrank. Ce dernier utilise l'équation 2.9 qui calcule et propage les poids des nœuds. Nous utilisons la

meilleure configuration trouvée dans [Jäschke *et al.* 2008] où le paramètre $d = 0,7$ et le nombre d'itérations est égal à 10. Initialement, chaque élément du vecteur préférence de l'utilisateur $\vec{P}$ a un poids égal à 1. En revanche, l'utilisateur et la ressource de la requête sont favorisés en leur associant respectivement les poids $1 + |U|$ et $1 + |R|$.

5.3.3 Taux de compression de Bibsonomy

Comme nous l'avons vu dans la section 5.2.2, les données de Bibsonomy ont été réduites en utilisant les différents algorithmes de clustering. Dans cette section, nous présentons le taux de compression de graphe initial avec chacune de ces méthodes.

Graphes	**#Nœuds**	**#Arêtes**	**#U**	**#T**	**#R**
G	889	24297	116	412	361
G_c (Mux-Licod)	434	1677	97	154	183
Taux compression en %	51,18	93,1	16,37	62,62	49,30
G_c (GenLouvain)	16	79	4	6	6
Taux compression en %	98,2	99,67	96,55	98,54	98,33
G_c (AC (Licod))	91	46	13	40	38
Taux compression en %	89,76	99,81	88,79	90,29	89,47
G_c (AC (Louvain))	9	27	3	3	3
Taux compression en %	98,98	99,88	97,41	99,27	99,16
G_c (EC (Licod))	151	993	3	89	59
Taux compression en %	83,08	95,91	97,41	78,39	83,65
G_c (EC (Louvain))	25	187	8	11	6
Taux compression en %	97,18	78,96	93,10	97,33	98,33

TABLE 5.4 – Taux de compression du graphe initial

Nous constatons que dans le cas où nous obtenons un nombre réduit de clusters il y a un risque que la taille de ces clusters soit grande. Ceci impacte directement le graphe contextuel qui peut avoir une taille conséquente. Dans la même logique, un nombre important de clusters peut avoir un effet inverse et nous obtenons des clusters de petites tailles ce qui permet d'avoir un graphe contextuel de taille très réduite. Dans la suite, nous présentons les statistiques sur le nombre de tags/ressources/utilisateurs par cluster pour chaque méthode.

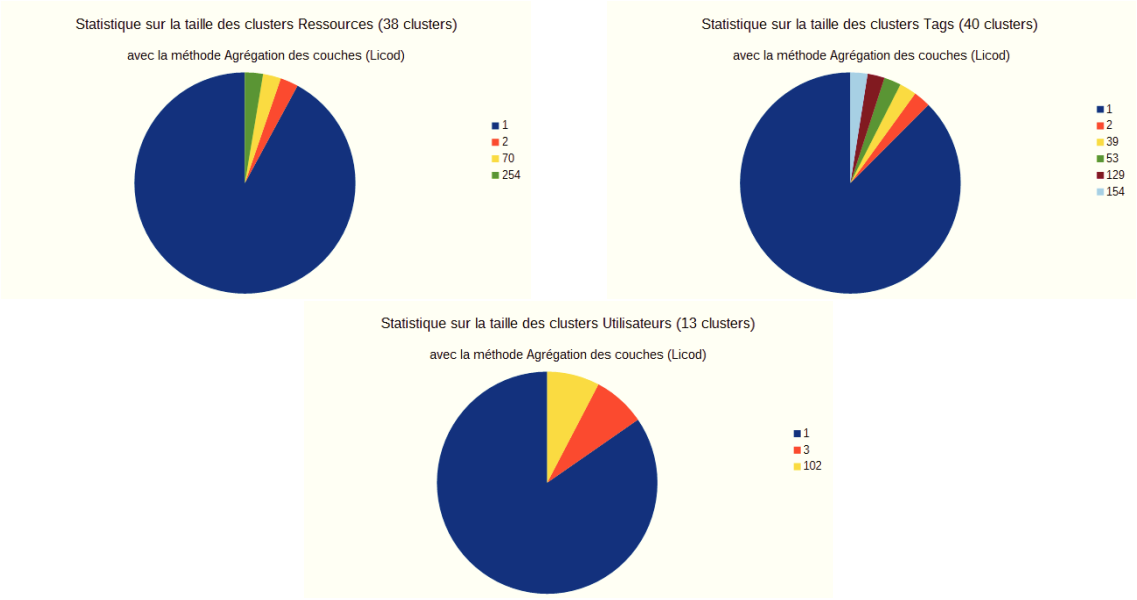

FIGURE 5.9 – Statistiques sur le nombre d'éléments par cluster avec Agrégation des couches (Licod)

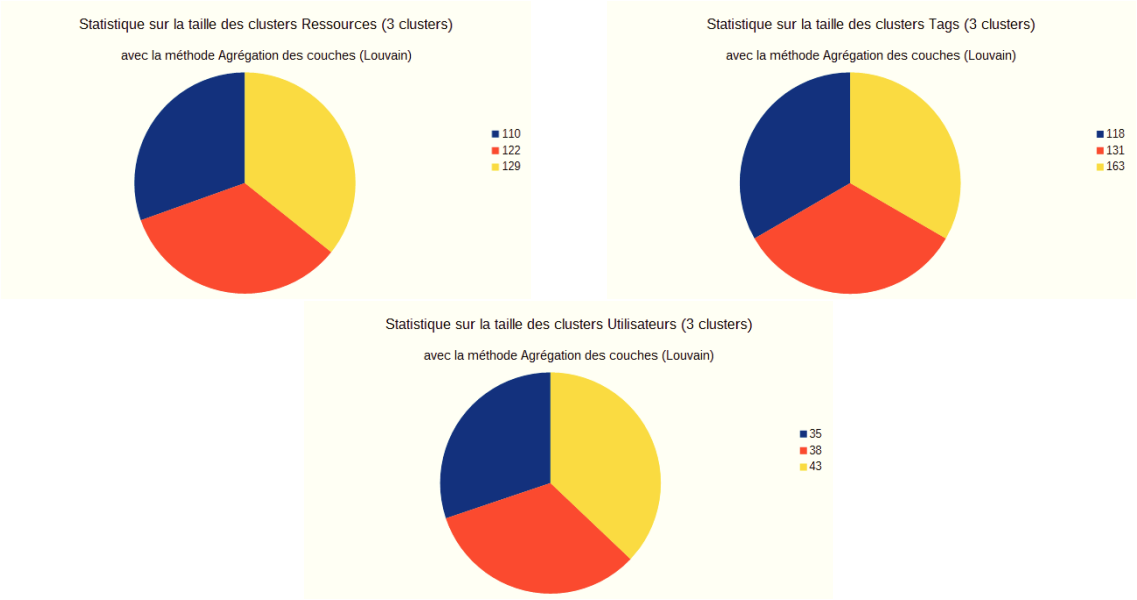

FIGURE 5.10 – Statistiques sur le nombre d'éléments par cluster avec Agrégation des couches (Louvain)

5.3.4 Résultats

Afin d'évaluer notre système de recommandation, nous choisissons aléatoirement la ressource r parmi les annotations précédemment effectuées par l'utilisateur u. Notons par

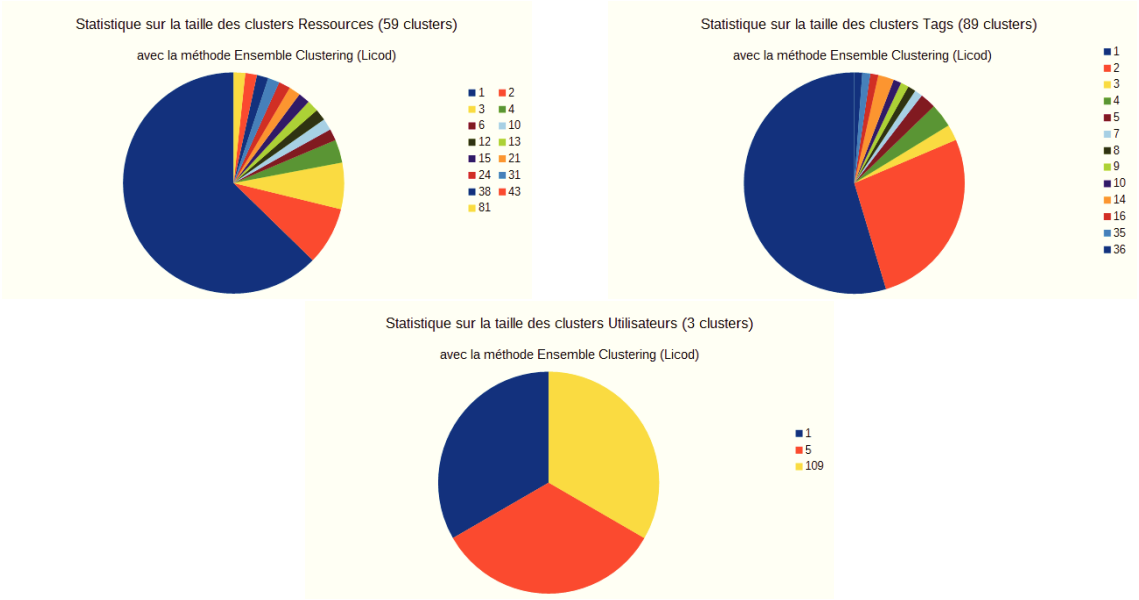

FIGURE 5.11 – Statistiques sur le nombre d'éléments par cluster avec Ensemble Clustering (Licod)

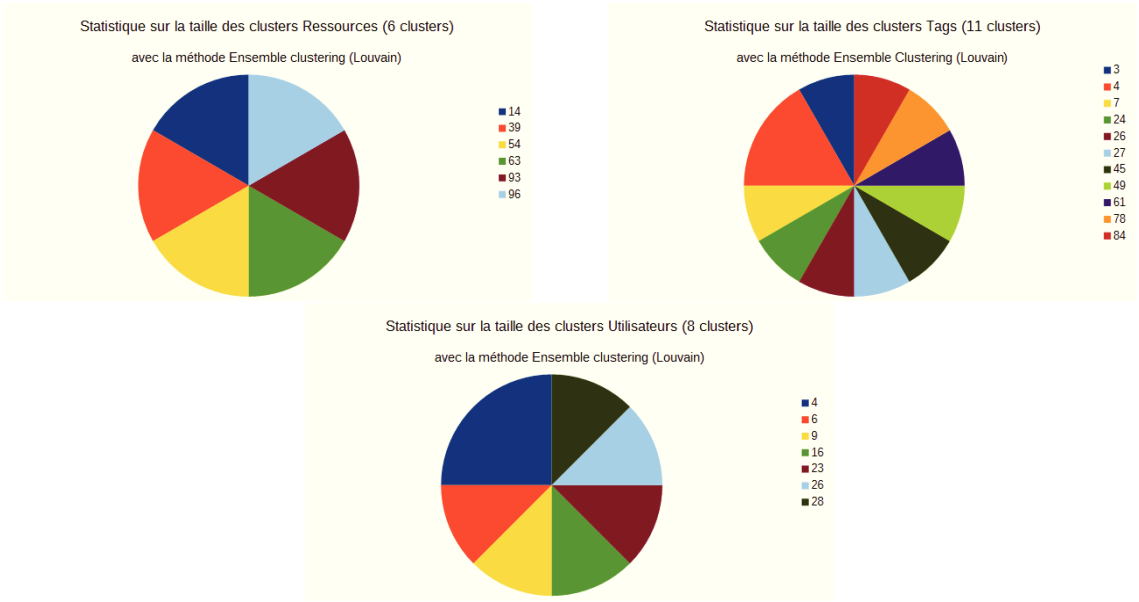

FIGURE 5.12 – Statistiques sur le nombre d'éléments par cluster avec Ensemble Clustering (Louvain)

K_t l'ensemble des tags recommandés. Pour évaluer les performances, nous utilisons la mesure de précision qui est une norme standard pour évaluer les systèmes de recommandation [Herlocker *et al.* 2004]. La précision est définie par:

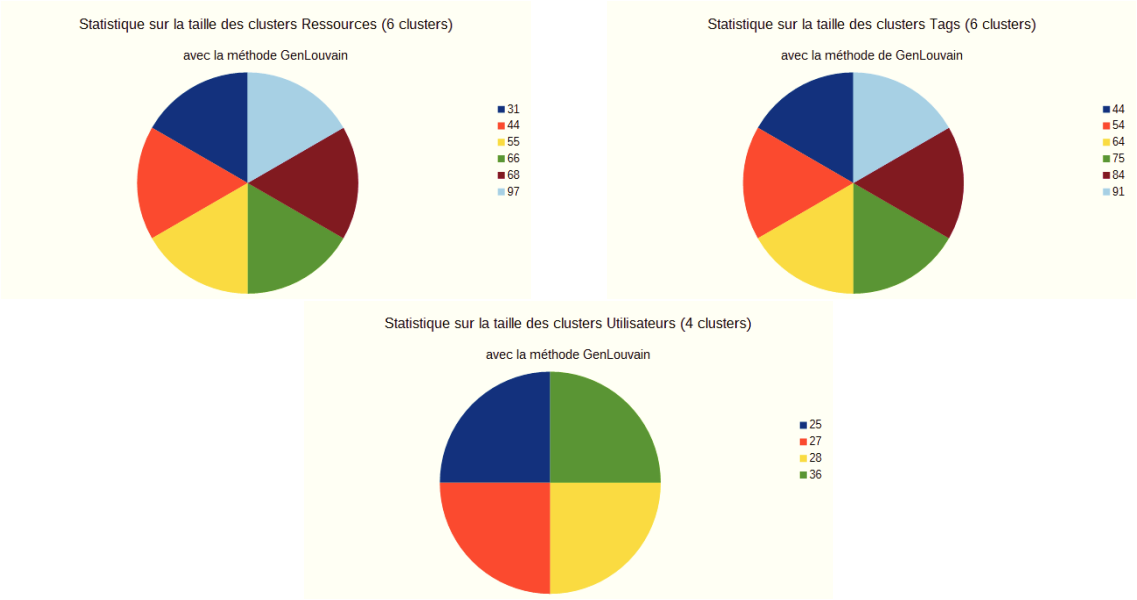

FIGURE 5.13 – Statistiques sur le nombre d'éléments par cluster avec GenLouvain

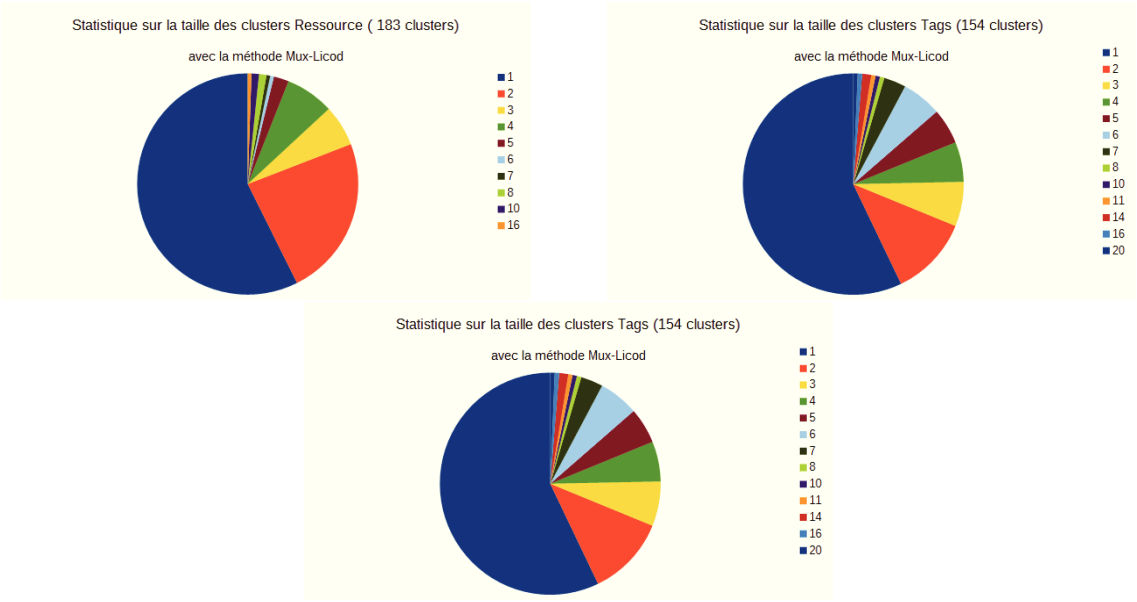

FIGURE 5.14 – Statistiques sur le nombre d'éléments par cluster avec Mux-Licod

$\text{Précision}(K_t(u,r)) = \sum_{u \in U} \frac{|T(u,r) \cap K_t(u,r)|}{|K_t(u,r)|}$.

Dans cette section, nous comparons les résultats de recommandation de l'approche de Folkrank classique par rapport à l'approche TLTR basée sur: ESC (Mux-Licod, Gen-

Louvain), EC (Licod, Louvain), AC (Licod, Louvain). Nous appliquons une procédure de 5-validation croisée pour évaluer notre système en divisant les données en cinq blocs. Nous utilisons quatre blocs pour l'apprentissage et un bloc pour le test. Les expérimentations ci-dessous présentent la moyenne des cinq expérimentations de recommandation de tags allant de 1 jusqu'à 4 tags en variant à chaque fois le nombre de clusters.

Nous rappelons que notre objectif initial était de comparer l'approche de recommandation

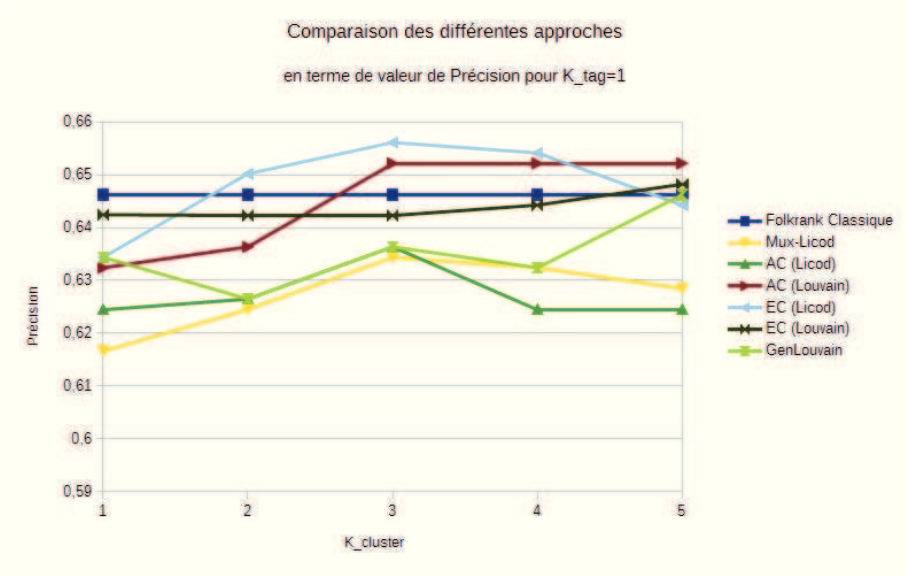

FIGURE 5.15 – Étude comparative des différentes approches de recommandation de tags en terme de précision avec $k_t = 1$

TLTR à l'approche de recommandation classique Folkrank. La méthodologie adoptée est d'étudier l'impact des trois paramètres de l'approche sur la précision et de confronter les résultats obtenus avec la précision de Folkrank. TLTR est évaluée en utilisant différents algorithmes: EC, AC, ESC (Genlouvain, Mux-Licod). La meilleure version de Mux-Licod trouvée dans la tâche de recommandation est celle du *mode*1. Pour rappel, le *mode*1 est expliqué dans la section 4.3.2 où nous avons effectué l'étude de paramétrage de Mux-Licod. Nous lançons une série d'expérimentations afin de déterminer, pour chaque configuration en variant le nombre des tags entre 1 et 4 et le nombre de clusters entre 1 à 5, la configuration optimale de TLTR, en termes d'algorithme de détection de communautés choisi et de nombre de clusters à sélectionner. Nous constatons que notre approche a obtenu de meilleurs résultats dans toutes les expérimentations (voir les figures 5.15, 5.16, 5.17, et

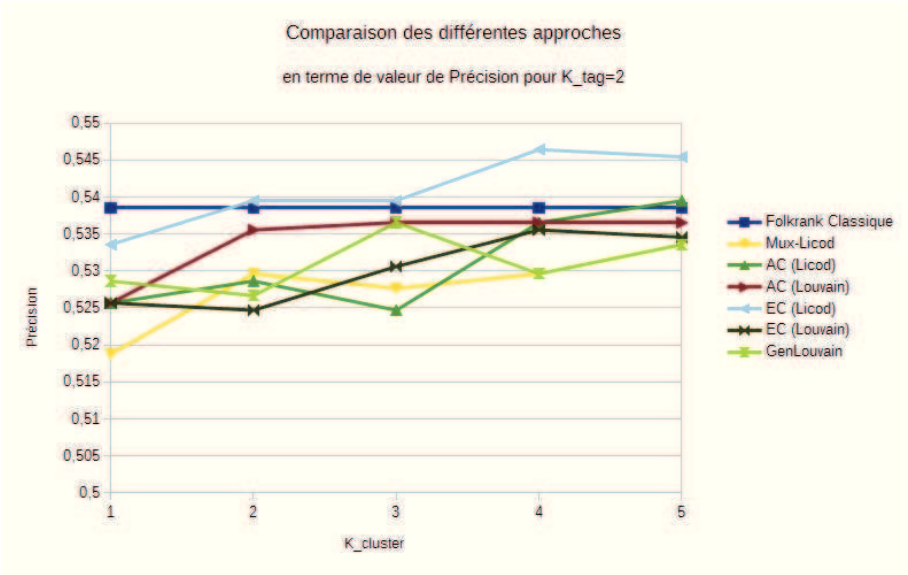

FIGURE 5.16 – Étude comparative de différentes approches de recommandation de tags en terme de précision avec $k_t = 2$

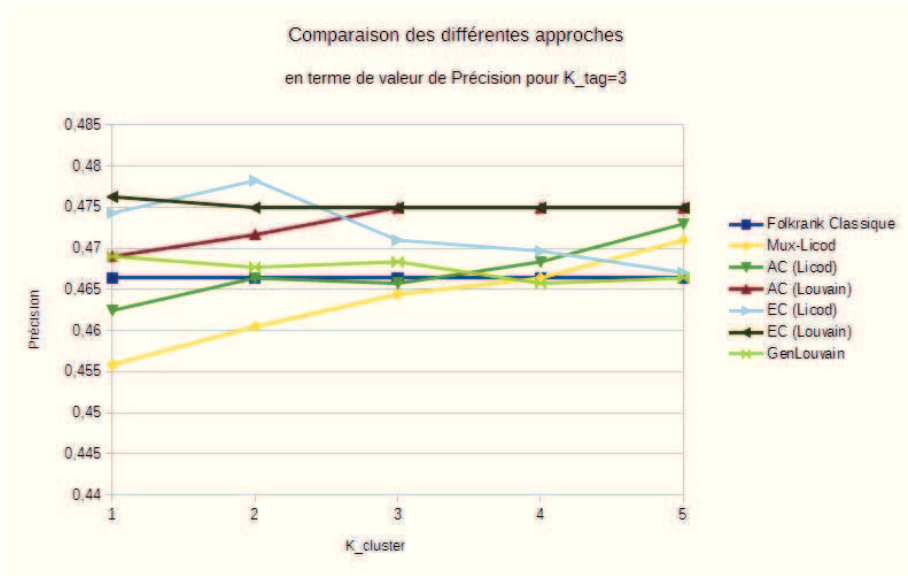

FIGURE 5.17 – Étude comparative de différentes approches de recommandation de tags en terme de précision avec $k_t = 3$

5.18). En effet, la meilleure précision obtenue par TLTR est 0.65 avec l'approche de recommandation Ensemble Clustering Licod où $K_t = 1$ avec $K_c = 3$ (voir la figure 5.15). Le résultat obtenu dépasse Folkrank qui a obtenu une précision égale à 0.64. Nous nous focalisons sur l'expérimentation qui évalue la recommandation avec $K_t = 3$. En effet, comme

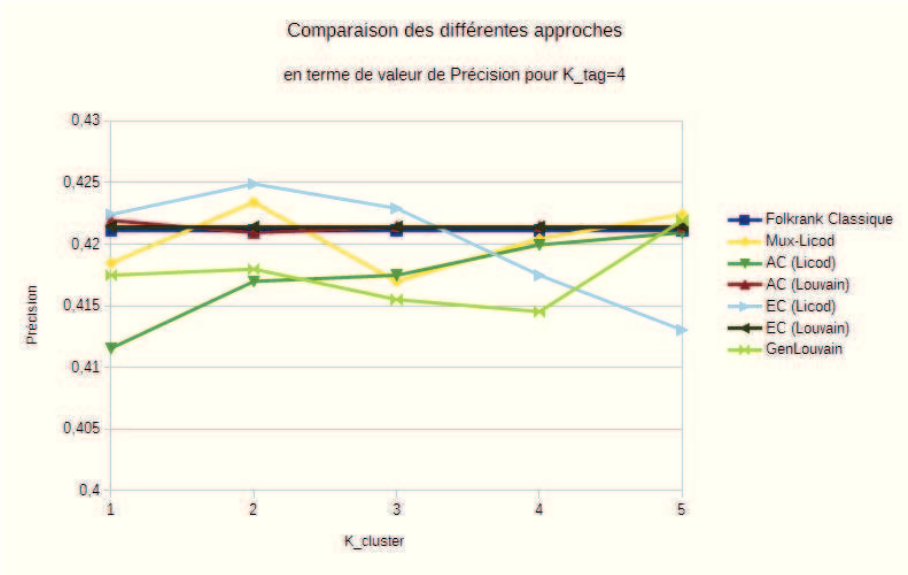

FIGURE 5.18 – Étude comparative de différentes approches de recommandation de tags en terme de précision avec $k_t = 4$

nous l'avons vu dans la figure 5.8, l'étude statistique sur les données de Bibsonomy a montré que la plupart des références bibliographiques ont été annotées par trois tags. Les résultats de la figure 5.17 montrent qu'en recommandant trois tags, les courbes de précision de TLTR, quelque soit l'algorithme de détection de communautés utilisé, ont dépassé la courbe de précision de Folkrank. Nous constatons que les résultats obtenus avec TLTR sont prometteurs permettant d'augmenter la précision de la recommandation de tags.

5.4 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle approche topologique de recommandation de tags basée sur le clustering des différentes composantes de la folksonomie (Utilisateurs, Tags et Ressources). Ce qui permet d'avoir un graphe de taille beaucoup plus réduite et d'avoir la structure à double niveau; le niveau générique composé uniquement par des clusters et le niveau spécifique présenté par les différentes composantes du graphe initial. Les résultats des expérimentations sur les données de Bibsonomy ont montré que l'approche de recommandation par niveaux, quelque soit la méthode de clustering utilisée (EC, AC, Mux-Licod, GenLouvain), est plus performante en terme de précision que celle

de l'approche classique de Folkrank.

Quatrième partie

Conclusion et Perspectives

Chapitre 6

Conclusion et Perspectives

Nous avons abordé dans cette thèse deux problématiques: la recommandation de tags dans les folksonomies et la détection de communautés dans les réseaux multiplexes.
Nous avons proposé une nouvelle approche topologique de recommandation de tags dite *par niveau*: une recommandation à un niveau général au service d'une recommandation à un niveau spécifique.
Comme l'indique son nom, la recommandation au niveau général généralise les concepts pour donner une vision macro sur les relations possibles entre les différentes entités. L'une des problématiques traitées dans cette thèse est comment définir ces entités. Nous avons donc eu recours au clustering et nous avons proposé une nouvelle approche.

Notre approche Mux-Licod est une généralisation directe de l'algorithme Licod [Yakoubi et Kanawati 2014], proposé initialement pour traiter des graphes monoplexes, au cas d'un multiplexe. Son principe est que les communautés se forment autour des nœuds appelés nœuds-leaders. Il calcule en premier temps les leaders puis il applique autour de ces leaders un processus d'expansion pour identifier les communautés dans le réseau.
L'ensemble des métriques classiques dans les réseaux simples est à redéfinir pour le cas de réseaux multiplexes. Nous redéfinissons trois métriques: le plus court chemin, le voisinage d'un noeud et le degré d'un nœud.
Nous avons étudié les différents types d'approches de détection de communautés dans le contexte multiplexe.

Notre approche, qui explore simultanément les couches d'un multiplexe, présente des performances intéressantes comparées aux approches classiques basées sur l'agrégation de couches ou l'agrégation des partitions ou même par rapport à l'algorithme générique *GenLouvain* [Mucha *et al.* 2010].

Les résultats ont montré, en utilisant les indicateurs topologiques de modularité et de redondance, que notre algorithme Mux-Licod est globalement meilleur que les autres approches de l'état de l'art. Les tests ont été effectués sur des données de taille différente.

Nous avons proposé aussi l'approche *TLTR* qui consiste à assurer une recommandation personnalisée de tags à deux niveaux.

De plus, nous exploitons la présence de la couche clusters pour ajouter un niveau de filtrage, par l'élimination dès le départ les clusters de tags non intéressants par rapport au contexte de l'utilisateur.

Par conséquent, la quantité de données traitées est nettement moins importante que celle de données de départ.

Les résultats des expérimentations montrent que la recommandation de tags sur les données de Bibsonomy par niveau, quelque soit le type d'approche de détection de communautés utilisée (Mux-Licod, Ensemble Clustering, Agrégation des couches, GenLouvain), offre un meilleur compromis en matière de taux de compression de graphe initial et de précision que l'approche classique de recommandation de tags sans niveau.

Les perspectives sont nombreuses sur chacun des axes présentés ci-dessus. Nous en présentons un certain nombre.

D'un point de vue méthodologique, nous avons utilisé pour assurer la phase de compression une approche centrée graine basée sur des calculs locaux. Prochainement, nous envisageons tester d'autres approches locales comme la propagation de label (LPA) [Raghavan *et al.* 2007]. Nous avons vu que la phase de clustering se fait hors ligne. Donc, la version actuelle nous oblige de calculer les clusters régulièrement à chaque évolution de la folksonomie en reproduisant la compression du graphe initial.

Pour accélérer les prétraitements et minimiser les coûts de calcul, nous proposons d'implémenter une version incrémentale des communautés permettant de faire la mise à jour toute en évoluant la structure communautaire d'une manière incrémentale évitant le

passage par les prétraitements de projection. Pour le traitement des très grands graphes, nous visons aussi passer par une implémentation parallèle afin d'optimiser le temps de calcul.
Le majeur problème que nous rencontrons est l'indisponibilité des données accessibles en ligne ayant la structure multiplexe où les couches sont composées par le même ensemble des nœuds pour pouvoir mieux évaluer notre approche.
En plus, le problème d'évaluation de communautés reste un problème ouvert, même pour les réseaux monoplexes. Des approches orientées tâches, à l'instar du travail présenté dans [Yakoubi et Kanawati 2014], sont aussi à l'étude. Nous avons effectué une première application des communautés dans la tâche de recommandation de tags.

Concernant les aspects liés à la recommandation, la perspective majeure est d'intégrer d'autres informations sur les utilisateurs à travers leurs interactions dans d'autres réseaux. Par exemple, inclure les informations sur les utilisateurs issues de deux réseaux disponibles en ligne comme Twitter et Instagram. Sur le plan expérimental, nous visons évaluer la recommandation de TLTR sur d'autres bases données de taille plus importante ainsi que d'utiliser d'autres algorithmes de recommandation de tags.

Bibliographie

Adamic, L. A. et Adar, E. (2003). Friends and neighbors on the web. *Social networks*, 25(3):211–230. 63

Adomavicius, G. et Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749. 28, 43

Aggarwal, C. C. et Reddy, C. K. (2013). *Data Clustering: Algorithms and Applications*. CRC Press. 62, 74

Asuero, A., Sayago, A. et Gonzalez, A. (2006). The correlation coefficient: An overview. *Critical reviews in analytical chemistry*, 36(1):41–59. 39

Au Yeung, C.-m., Gibbins, N. et Shadbolt, N. (2009). Contextualising tags in collaborative tagging systems. *In Proceedings of the 20th ACM conference on Hypertext and hypermedia*, pages 251–260. ACM. 52

Aynaud, T. et Guillaume, J.-L. (2010). Static community detection algorithms for evolving networks. *In Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 513–519. IEEE. 69

Baeza-Yates, R., Ribeiro-Neto, B. *et al.* (1999). *Modern information retrieval*, volume 463. ACM press New York. 45

Barabási, A.-L. et Albert, R. (1999). Emergence of scaling in random networks. *science*, 286(5439):509–512. 63

Battiston, F., Nicosia, V. et Latora, V. (2013). Metrics for the analysis of multiplex networks. *arXiv preprint arXiv:1308.3182*. 84, 85, 91, 95

Begelman, G., Keller, P., Smadja, F. *et al.* (2006). Automated tag clustering: Improving search and exploration in the tag space. *In Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*, pages 15–33. 51, 52

Berlingerio, M., Coscia, M. et Giannotti, F. (2011). Finding and characterizing communities in multidimensional networks. *In Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 490–494. IEEE. 16, 80, 81, 85, 89, 104

Berlingerio, M., Coscia, M., Giannotti, F., Monreale, A. et Pedreschi, D. (2013). Evolving networks: Eras and turning points. *Intell. Data Anal.*, 17(1):27–48. 89

Blondel, V. D., Guillaume, J.-l. et Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:P10008. 67, 84, 103, 104, 106

Bogers, T. et Van den Bosch, A. (2008). Recommending scientific articles using citeulike. *In Proceedings of the 2008 ACM conference on Recommender systems*, pages 287–290. ACM. 40

Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z. et Wagner, D. (2008). On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20(2):172–188. 66

Brin, S. et Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117. 47

Brodka, P. et Kazienko, P. (2014). Encyclopedia of social network analysis and mining, ch. multi-layered social networks. *Springer*. 91, 93, 94

Brooks, C. H. et Montanez, N. (2006). Improved annotation of the blogosphere via autotagging and hierarchical clustering. *In Proceedings of the 15th International Confe-*

rence on World Wide Web, WWW '06, pages 625–632, New York, NY, USA. ACM. 51, 52

Brown, G. (2010). Ensemble learning. *In Encyclopedia of Machine Learning*, pages 312–320. Springer. 89

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370. 28

Cai, D., Shao, Z., He, X., Yan, X. et Han, J. (2005). Mining hidden community in heterogeneous social networks. *In Proceedings of the 3rd international workshop on Link discovery*, pages 58–65. ACM. 82

Cai, Y., Shi, C., Dong, Y., Ke, Q. et Wu, B. (2011). A novel genetic algorithm for overlapping community detection. *In Proceedings of the 7th International Conference on Advanced Data Mining and Applications - Volume Part I*, ADMA'11, pages 97–108, Berlin, Heidelberg. Springer-Verlag. 66

Campello, R. J. (2007). A fuzzy extension of the rand index and other related indexes for clustering and classification assessment. *Pattern Recognition Letters*, 28(7):833–841. 74

Chevaleyre, Y., Endriss, U., Lang, J. et Maudet, N. (2007). A short introduction to computational social choice. *In* van Leeuwen, J., Italiano, G. F., van der Hoek, W., Meinel, C., Sack, H. et Plasil, F., éditeurs : *SOFSEM (1)*, volume 4362 de *Lecture Notes in Computer Science*, pages 51–69. Springer. 72, 155

Chirita, P.-A., Costache, S., Nejdl, W. et Handschuh, S. (2007). P-tag: large scale automatic generation of personalized annotation tags for the web. *In Proceedings of the 16th international conference on World Wide Web*, pages 845–854. ACM. 46

Coleman, J., Katz, E. et Menzel, H. (1957). The diffusion of an innovation among physicians. *Sociometry*, 20(4):253–270. 96

Cordasco, G. et Gargano, L. (2012). Label propagation algorithm: a semi-synchronous approach. *International Journal of Social Network Mining*, 1(1):3–26. 69

Corlette, D. et Shipman III, F. M. (2010). Link prediction applied to an open large-scale online social network. *In Proceedings of the 21st ACM conference on Hypertext and hypermedia*, pages 135–140. ACM. 69

Costa, L. d. F., Evuskoff, A., Mangioni, G. et Menezes, R. (2011). *Complex Networks: Second International Workshop, CompleNet 2010, Rio de Janeiro, Brazil, October 13-15, 2010, Revised Selected Papers*, volume 116. Springer. 84

Davis, D., Lichtenwalter, R. et Chawla, N. V. (2011). Multi-relational link prediction in heterogeneous information networks. *In Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 281–288. IEEE. 78

Deshpande, M. et Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177. 38

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271. 119

Donetti, L. et Munoz, M. A. (2004). Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment*, 2004(10):P10012. 67

Duch, J. et Arenas, A. (2005). Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104. 66

Dwork, C., Kumar, R., Naor, M. et Sivakumar, D. (2001). Rank aggregation methods for the web. *In Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 613–622. ACM. 51, 98, 119, 155

Eck, D., Lamere, P., Bertin-Mahieux, T. et Green, S. (2007). Automatic generation of social tags for music recommendation. *In Advances in neural information processing systems*, pages 385–392. 35

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3):75–174. 60

Fortunato, S. et Barthelemy, M. (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41. 68

Fouss, F., Pirotte, A., Renders, J.-M. et Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *Knowledge and Data Engineering, IEEE Transactions on*, 19(3):355–369. 64

Frank, E., Paynter, G. W., Witten, I. H., Gutwin, C. et Nevill-Manning, C. G. (1999). Domain-specific keyphrase extraction. 46

Garg, N. et Weber, I. (2008). Personalized, interactive tag recommendation for flickr. *In Proceedings of the 2008 ACM conference on Recommender systems*, pages 67–74. ACM. 35

Gemmell, J., Schimoler, T., Mobasher, B. et Burke, R. (2010). Hybrid tag recommendation for social annotation systems. *In Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 829–838. ACM. 36, 55

Gemmell, J., Shepitsen, A., Mobasher, B. et Burke, R. (2008). Personalizing navigation in folksonomies using hierarchical tag clustering. *In Data Warehousing and Knowledge Discovery*, pages 196–205. Springer. 51, 52, 54

Giannakidou, E., Koutsonikola, V., Vakali, A. et Kompatsiaris, I. (2008). Co-clustering tags and social data sources. *In Web-Age Information Management, 2008. WAIM'08. The Ninth International Conference on*, pages 317–324. IEEE. 52

Girvan, M. et Newman, M. E. J. (2002). Community structure in social and biological networks. *PNAS*, 99(12):7821–7826. 104

Goder, A. et Filkov, V. (2008). Consensus clustering algorithms: Comparison and refinement. *In ALENEX*, volume 8, pages 109–117. SIAM. 82

Goldberg, D., Nichols, D., Oki, B. M. et Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70. 37

Good, B. H., de Montjoye, Y.-A. et Clauset, A. (2010). Performance of modularity maximization in practical contexts. *Physical Review E*, 81(4):046106. 68

Graham, R. et Caverlee, J. (2008). Exploring feedback models in interactive tagging. *In Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 1, pages 141–147. IEEE. 46

Gregory, S. (2010). Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103–118. 70

Gruber, T. (2007). Ontology of folksonomy: A mash-up of apples and oranges. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 3(1):1–11. 52

Guimera, R., Sales-Pardo, M. et Amaral, L. A. N. (2004). Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101. 66

Herlocker, J. L., Konstan, J. A., Terveen, L. G. et Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53. 125

Hotho, A., Jäschke, R., Schmitz, C. et Stumme, G. (2006). Information retrieval in folksonomies: Search and ranking. *In The semantic web: research and applications*, pages 411–426. Springer. 23, 29, 47, 48, 49, 121, 122

Hubert, L. et Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1): 193–218. 75

Jaccard, P. (1901). *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz. 63

Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L. et Stumme, G. (2007). Tag recommendations in folksonomies. *In Knowledge Discovery in Databases: PKDD 2007*, pages 506–514. Springer. 28, 35

Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L. et Stumme, G. (2008). Tag recommendations in social bookmarking systems. *Ai Communications*, 21(4):231–247. 28, 47, 109, 111, 112, 118, 120, 121, 123

Java, A., Joshi, A. et Finin, T. (2008). Detecting commmunities via simultaneous clustering of graphs and folksonomies. *In Proceedings of WebKDD*, volume 2008. 52

Ju, S. et Hwang, K.-B. (2009). A weighting scheme for tag recommendation in social bookmarking systems. *In Proc. the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 109–118. 55

Kanawati, R. (2014). Seed-centric approaches for community detection in complex networks. *In Social Computing and Social Media*, pages 197–208. Springer. 71, 90

Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43. 64

Kazienko, P., Brodka, P. et Musial, K. (2010). Individual neighbourhood exploration in complex multi-layered social network. *In Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 3, pages 5–8. IEEE. 93

Khorasgani, R. R., Chen, J. et Zaïane, O. R. (2010). Top leaders community detection approach in information networks. *In 4th SNA-KDD Workshop on Social Network Mining and Analysis, Washington DC*. 70

Lancichinetti, A. et Fortunato, S. (2011). Limits of modularity maximization in community detection. *Physical Review E*, 84(6):066122. 68

Lancichinetti, A. et Fortunato, S. (2012). Consensus clustering in complex networks. *Scientific reports*, 2. 69

Lancichinetti, A., Fortunato, S. et Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110. 67

Lancichinetti, A., Radicchi, F., Ramasco, J. J. et Fortunato, S. (2011). Finding statistically significant communities in networks. *PloS one*, 6(4):e18961. 73

Lazega, E. (2001). *The collegial phenomenon : the social mechanisms of cooperation among peers in a corporate law partnership*. Oxford university press, Oxford. 97

Lee, S. O. K. et Chun, A. H. W. (2007). Automatic tag recommendation for the web 2.0 blogosphere using collaborative tagging and hybrid ann semantic structures. *In Proceedings of the 6th Conference on WSEAS International Conference on Applied Computer Science - Volume 6*, ACOS'07, pages 88–93, Stevens Point, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS). 28, 44

Leung, I. X., Hui, P., Lio, P. et Crowcroft, J. (2009). Towards real-time community detection in large networks. *Physical Review E*, 79(6):066107. 69

Li, J. et Song, Y. (2013). Community detection in complex networks using extended compact genetic algorithm. *Soft Computing*, 17(6):925–937. 66

Li, X., Snoek, C. G. et Worring, M. (2009). Learning social tag relevance by neighbor voting. *Multimedia, IEEE Transactions on*, 11(7):1310–1322. 52

Lipczak, M., Hu, Y., Kollet, Y. et Milios, E. (2009). Tag sources for recommendation in collaborative tagging systems. *ECML PKDD discovery challenge*, pages 157–172. 28

Liu, Z., Shi, C. et Sun, M. (2010). Folkdiffusion: A graph-based tag suggestion method for folksonomies. *In Information Retrieval Technology*, pages 231–240. Springer. 50

Lü, L. et Zhou, T. (2011). Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170. 63

Marinho, L. B., Nanopoulos, A., Schmidt-Thieme, L., Jäschke, R., Hotho, A., Stumme, G. et Symeonidis, P. (2011). Social tagging recommender systems. *In Recommender systems handbook*, pages 615–644. Springer. 15, 36, 38, 39

Medelyan, O., Frank, E. et Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. *In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1318–1327. Association for Computational Linguistics. 46

Meilă, M. (2003). Comparing clusterings by the variation of information. *In Learning theory and kernel machines*, pages 173–187. Springer. 76

Mika, P. (2005). Ontologies are us: A unified model of social networks and semantics. *In The Semantic Web-ISWC 2005*, pages 522–536. Springer. 114

Mishne, G. (2006). Autotag: a collaborative approach to automated tag assignment for weblog posts. *In Proceedings of the 15th international conference on World Wide Web*, pages 953–954. ACM. 35, 40, 45

Mucha, P. J., Richardson, T., Macon, K., Porter, M. A. et Onnela, J.-P. (2010). Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878. 83, 89, 103, 134

Musto, C., Narducci, F., de Gemmis, M., Lops, P. et Semeraro, G. (2009). Star: a social tag recommender system. *In Proceeding of ECML/PKDD 2009 Discovery Challenge Workshop*, pages 215–227. Citeseer. 55

Musto, C., Narducci, F., De Gemmis, M., Lops, P. et Semeraro, G. (2010). An "ir"-based approach for tag recommendation. *In IIR*, pages 65–69. Citeseer. 36, 45

Newman, M. E. (2004). Coauthorship networks and patterns of scientific collaboration. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5200–5205. 66, 67

Newman, M. E. (2006). Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104. 67

Newman, M. E. et Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113. 54

Papadopoulos, S., Kompatsiaris, Y. et Vakali, A. (2009). Leveraging collective intelligence through community detection in tag networks. *CKCaR, September*. 52

Papadopoulos, S., Kompatsiaris, Y., Vakali, A. et Spyridonos, P. (2012). Community detection in social media. *Data Mining and Knowledge Discovery*, 24(3):515–554. 15, 60, 62, 78

Papadopoulos, S., Vakali, A. et Kompatsiaris, Y. (2011). Community detection in collaborative tagging systems. *In Community-Built Databases*, pages 107–131. Springer. 60, 70

Parra, D. et Brusilovsky, P. (2009). Collaborative filtering for social tagging systems: an experiment with citeulike. *In Proceedings of the third ACM conference on Recommender systems*, pages 237–240. ACM. 36, 39

Peters, I. (2009). *Folksonomies: Indexing and Retrieval in the Web 2.0*, volume 1. Walter de Gruyter. 23

Pizzuti, C. (2012). Boosting the detection of modular community structure with genetic algorithms and local search. *In Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 226–231. ACM. 66

Pons, P. et Latapy, M. (2005). Computing communities in large networks using random walks. *In Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer. 67, 103, 104

Potgieter, A., April, K. A., Cooke, R. J. et Osunmakinde, I. O. (2009). Temporality in link prediction: Understanding social complexity. *Emergence: Complexity & Organization*, 11(1). 81

Pujari, M. et Kanawati, R. (2012). Tag recommendation by link prediction based on supervised machine learning. *In ICWSM*. 48, 50, 51

Pujari, M. et Kanawati, R. (2014). Link prediction in multiplex networks. *Networks and Heterogeneous Media*. Special Issue on New trends, models and applications in Complex and Multiplex Networks. 78, 96

Rae, A., Sigurbjörnsson, B. et van Zwol, R. (2010). Improving tag recommendation using social networks. *In Adaptivity, Personalization and Fusion of Heterogeneous Information*, RIAO '10, pages 92–99, Paris, France, France. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE. 47, 109

Raghavan, U. N., Albert, R. et Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036–106. 69, 70, 134

Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N. et Barabási, A.-L. (2002). Hierarchical organization of modularity in metabolic networks. *science*, 297(5586):1551–1555. 63

Reichardt, J. et Bornholdt, S. (2006). Statistical mechanics of community detection. *Physical Review E*, 74(1):016110. 66, 68

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. et Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. *In Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA. ACM. 37

Rosvall, M., Axelsson, D. et Bergstrom, C. T. (2009). The map equation. *Eur. Phys. J. Special Topics*, 13:178. 104, 105

Said, A., Berkovsky, S. et De Luca, E. W. (2010). Putting things in context: Challenge on context-aware movie recommendation. *In Proceedings of the Workshop on Context-Aware Movie Recommendation*, pages 2–6. ACM. 35

Salton, G. et McGill, M. J. (1983). Introduction to modern information retrieval. 63

Sammut, C. et Webb, G. I. (2010). *Encyclopedia of machine learning*. Springer. 82

Sarwar, B., Karypis, G., Konstan, J. et Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *In Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM. 38

Schafer, J. B., Frankowski, D., Herlocker, J. et Sen, S. (2007). Collaborative filtering recommender systems. *In The adaptive web*, pages 291–324. Springer. 39

Schein, A. I., Popescul, A., Ungar, L. H. et Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. *In Proceedings of the 25th Annual International*

ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '02, pages 253–260, New York, NY, USA. ACM. 56

Sculley, D. (2007). Rank aggregation for similar items. *In SDM*, pages 587–592. SIAM. 98, 103, 155

Seifi, M. (2012). *Coeurs stables de communautés dans les graphes de terrain.* Thèse de doctorat. 69, 82, 103

Shah, D. et Zaman, T. (2010). Community detection in networks: The leader-follower algorithm. *arXiv preprint arXiv:1011.0774.* 70

Shardanand, U. et Maes, P. (1995). Social information filtering: Algorithms for automating “word of mouth”. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 210–217, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co. 37

Shepitsen, A., Gemmell, J., Mobasher, B. et Burke, R. (2008). Personalized recommendation in social tagging systems using hierarchical clustering. *In Proceedings of the 2008 ACM conference on Recommender systems*, pages 259–266. ACM. 52, 60

Sigurbjörnsson, B. et Van Zwol, R. (2008). Flickr tag recommendation based on collective knowledge. *In Proceedings of the 17th international conference on World Wide Web*, pages 327–336. ACM. 35, 52

Song, Y., Zhang, L. et Giles, C. L. (2008a). A sparse gaussian processes classification framework for fast tag suggestions. *In Proceedings of the 17th ACM conference on Information and knowledge management*, pages 93–102. ACM. 36, 44

Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W.-C. et Giles, C. L. (2008b). Real-time automatic tag recommendation. *In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522. ACM. 35

Sood, S., Owsley, S., Hammond, K. J. et Birnbaum, L. (2007). Tagassist: Automatic tag suggestion for blog posts. *In ICWSM.* 42, 45

Sorensen, T. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biol. skr.*, 5:1–34. 63

Strehl, A. et Ghosh, J. (2003). Cluster ensembles a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617. 76, 82

Šubelj, L. et Bajec, M. (2011). Robust network community detection using balanced propagation. *The European Physical Journal B*, 81(3):353–362. 69, 70

Suthers, D. D., Fusco, J., Schank, P. K., Chu, K.-H. et Schlager, M. S. (2013). Discovery of community structures in a heterogeneous professional online network. *In HICSS*, pages 3262–3271. 81

Szomszor, M., Cattuto, C., Alani, H., O'Hara, K., Baldassarri, A., Loreto, V. et Servedio, V. D. (2007). Folksonomies, the semantic web, and movie recommendation. *In 4th European Semantic Web Conference, Bridging the Gap between Semantic Web and Web 2.0*. 35

Tang, L. et Liu, H. (2010). Community detection and mining in social media. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 2(1):1–137. 15, 16, 60, 62, 64, 65, 67, 81, 83

Tatu, M., Srikanth, M. et D'Silva, T. (2008). Rsdc08:tag recommendations using bookmark content. *ECML PKDD discovery challenge*, 2008:96–107. 54

Topchy, A., Jain, A. K. et Punch, W. (2005). Clustering ensembles: Models of consensus and weak partitions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12):1866–1881. 82

Toussaint, G. T. et Bhattacharya, B. K. (1981). Optimal algorithms for computing the minimum distance between two finite planar sets. *In Pattern Recognition Letters*, pages 79–82. 78

Tso-Sutter, K. H. L., Marinho, L. B. et Schmidt-Thieme, L. (2008). Tag-aware recommender systems by fusion of collaborative filtering algorithms. *In Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, pages 1995–1999, New York, NY, USA. ACM. 15, 37, 40, 41

Vander Wal, T. (2005). Explaining and showing broad and narrow folksonomies. *online posting, Feb*, 21. 15, 24, 25

Viana, W., Braga, R., Lemos, F., de Souza, J. O., Carmo, R., Andrade, R. et Martin, H. (2013). Mobile photo recommendation and logbook generation from context-tagged images. *IEEE Multimedia*, 99(PrePrints):1. 35

Vinh, N. X., Epps, J. et Bailey, J. (2009). Information theoretic measures for clusterings comparison: is a correction for chance necessary? *In Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073–1080. ACM. 76

Wang, J., De Vries, A. P. et Reinders, M. J. (2006). Unifying user-based and item-based collaborative filtering approaches by similarity fusion. *In Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508. ACM. 37

Weston, J., Bengio, S. et Usunier, N. (2010). Large scale image annotation: learning to rank with joint word-image embeddings. *Machine learning*, 81(1):21–35. 45

Xie, J. et Szymanski, B. K. (2011). Community detection using a neighborhood strength driven label propagation algorithm. *In Network Science Workshop (NSW), 2011 IEEE*, pages 188–195. IEEE. 70

Xu, S., Bao, S., Fei, B., Su, Z. et Yu, Y. (2008). Exploring folksonomy for personalized search. *In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 155–162. ACM. 43

Yakoubi, Z. et Kanawati, R. (2012). Classification non-supervisée par application d'un algorithme de détection de communautés dans les réseaux complexes. *SFC12 : XIX journée de la Société Française de Classification.* 78

Yakoubi, Z. et Kanawati, R. (2013). Leader-driven approach for community detection in complex network. *In proceedings of the international conference on intercations in complex systems*. 78

Yakoubi, Z. et Kanawati, R. (2014). Licod: A leader-driven algorithm for community detection in complex networks. *Vietnam Journal of Computer Science*, 1(4):241–256. 29, 71, 90, 91, 103, 104, 133, 135

Yang, J. et Leskovec, J. (2012). Defining and evaluating network communities based on ground-truth. *In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 3. ACM. 73

Young, H. P. et Levenglick, A. (1978). A consistent extension of condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2):285–300. 155

Zhang, Z.-K., Zhou, T. et Zhang, Y.-C. (2011). Tag-aware recommender systems: a state of the art survey. *Journal of Computer Science and Technology*, 26(5):767–777. 47

Zhou, T., Lü, L. et Zhang, Y.-C. (2009). Predicting missing links via local information. *The European Physical Journal B-Condensed Matter and Complex Systems*, 71(4):623–630. 63, 64

Annexes

Annexe

.1 Différentes méthodes de fusion de votes

La fusion de votes consiste à fusionner le vecteur d'appartenance de chaque nœud avec les vecteurs de ses voisins. Différentes méthodes de fusion peuvent être utilisées. Dans Mux-Licod, nous avons choisi d'utiliser des algorithmes issus de la théorie du choix social [Chevaleyre *et al.* 2007]. Plus précisément, nous appliquons les trois méthodes suivantes: la méthode de Majorité, la méthode de Borda [Sculley 2007], et la méthode Kemeny [Dwork *et al.* 2001].

Méthode de Majorité: est la méthode de fusion la plus simple. Le calcul se fonde sur la position individuelle des candidats (communautés) dans tous les votes. Pour chaque rang $r \in [1, 2, ..., k]$, on retient le candidat qui a été le plus classé dans ce rang.

Méthode de Borda: est une méthode fondée sur le positionnement absolu des communautés classées plutôt que leur classement respectif. Un score de Borda est calculé pour chaque communauté c dans les vecteurs agrégés. Pour l'ensemble des vecteurs d'appartenance L, le score d'une communauté c pour un vecteur L_i est donné par:

$$B_{L_i}(c) = \left\{ count(c') | L_i(c) > L_i(c'), c' \in L_i \right\} \tag{1}$$

Le score de Borda total de c est donné par:

$$B(c) = \sum_{t=1}^{n} B_{L_i}(c) \tag{2}$$

Méthode de Kemeny: est fondée sur l'ordre entre le rang des communautés. Elle garantit que le résultat de la fusion satisfait le principe de Condorcet [Young et Levenglick 1978]. Le principe est comme suit: s'il existe une partition D de la liste totale des

candidats (i.e. les communautés) C telle que $\forall x \in C$, $\forall y \in D$, si la majorité des électeurs (i.e. les votes des nœuds voisins) préfère x à y, alors x est classé au-dessus de y.

.2 Étude de paramétrage de Mux-Licod

.2.1 Avec la méthode de fusion Kemeny

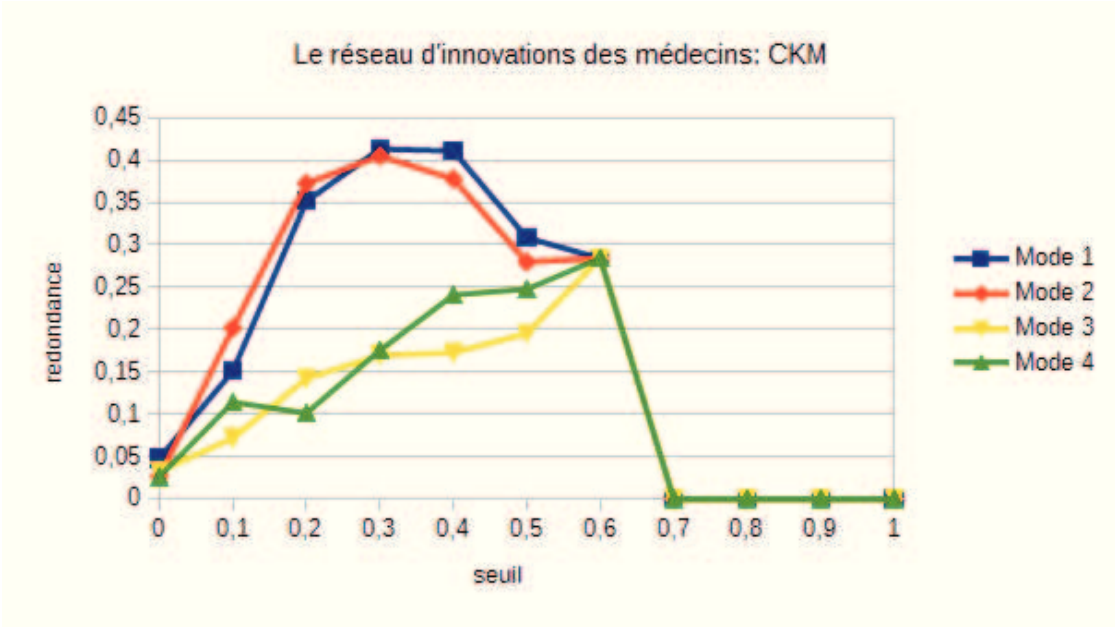

FIGURE 1 – Mesure de redondance sur le réseau d'innovations des médecins

.2.2 Avec la méthode de vote Majorité

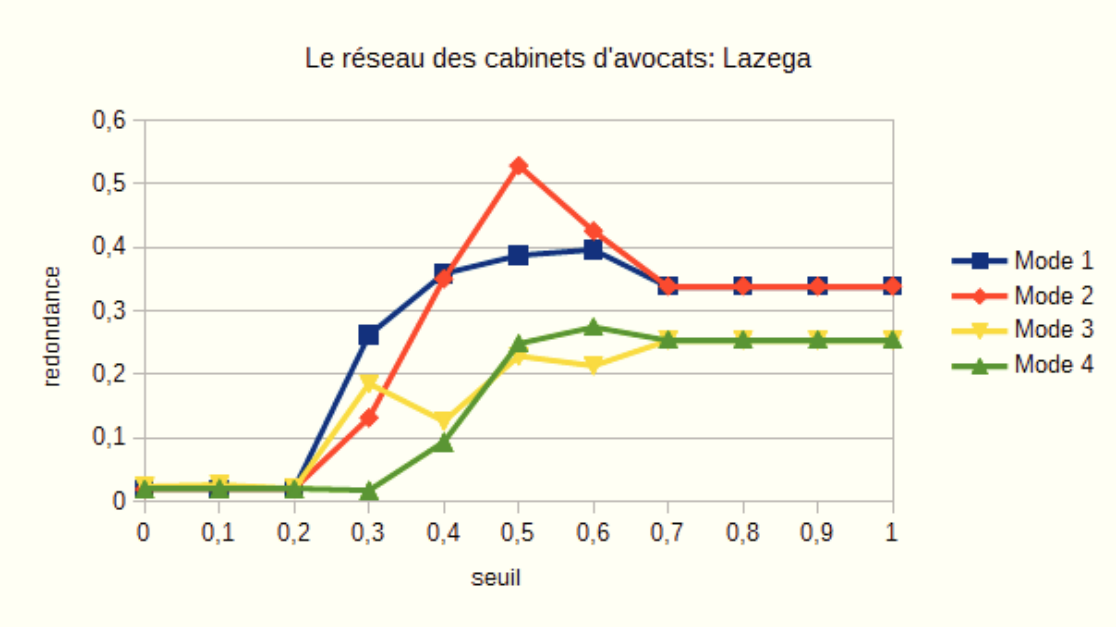

FIGURE 2 – Mesure de redondance sur le réseau de cabinet d'avocats

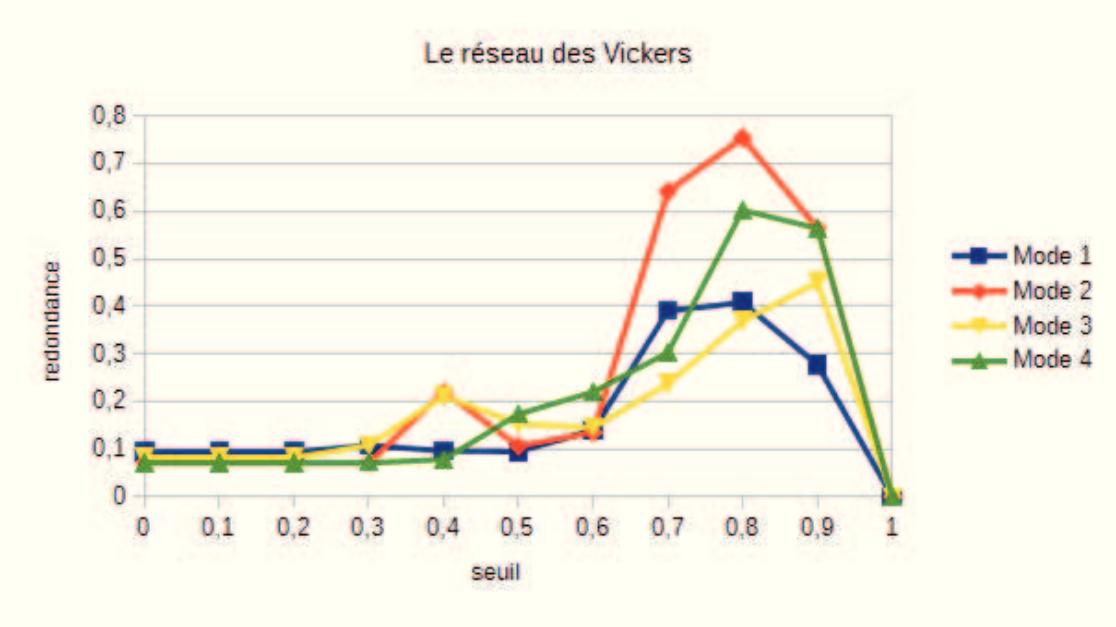

FIGURE 3 – Mesure de redondance sur le réseau des Vickers

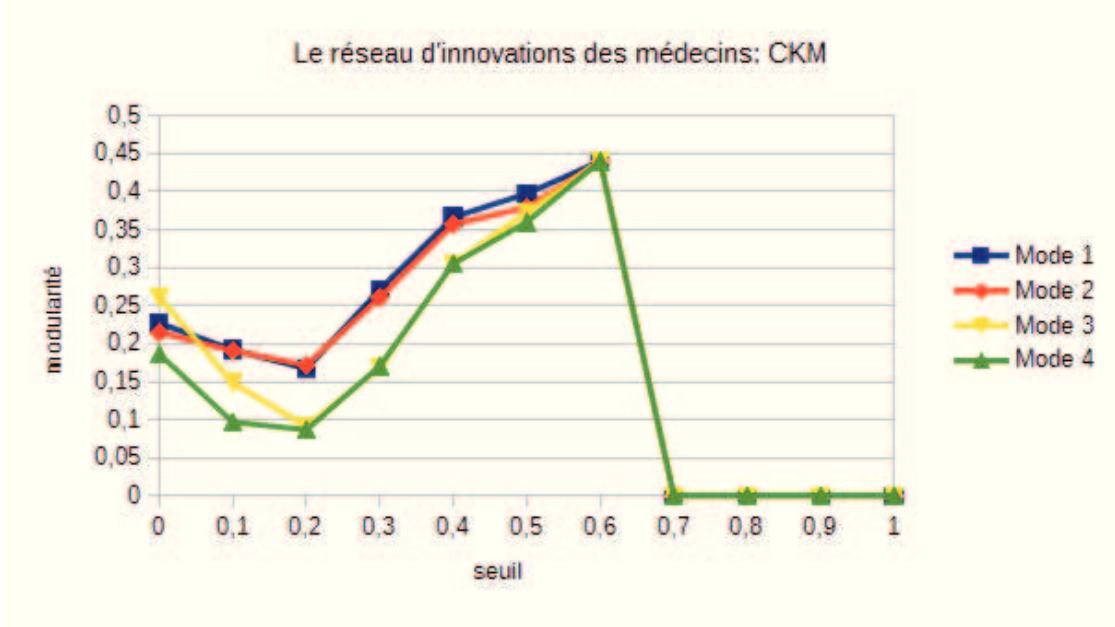

FIGURE 4 – Mesure de modularité sur le réseau d'innovations des médecins: CKM

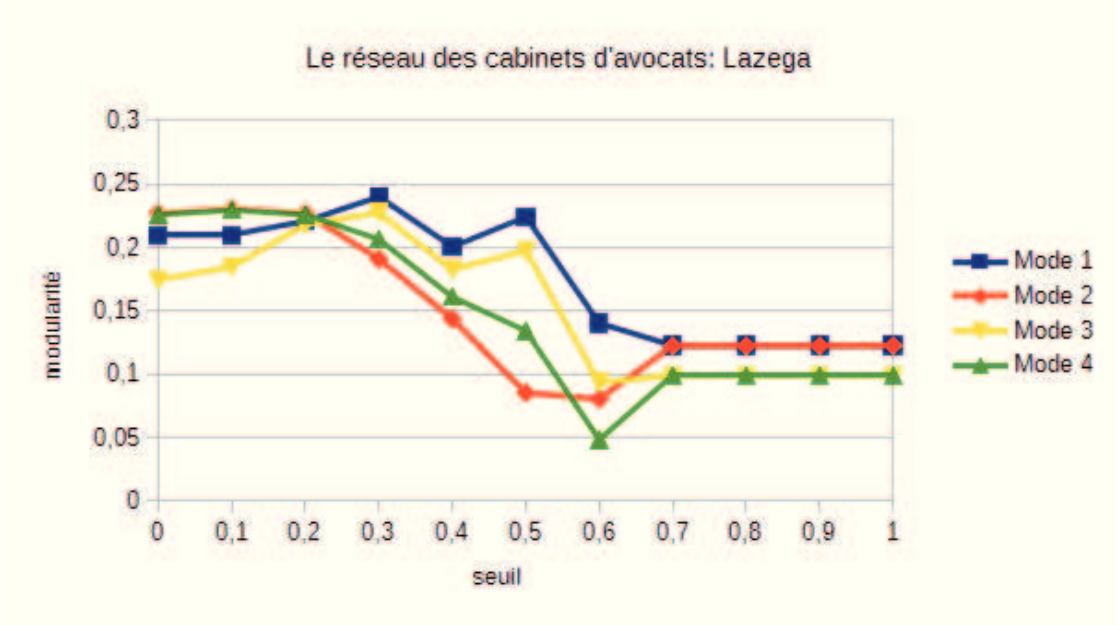

FIGURE 5 – Mesure de modularité sur le réseau de cabinet d'avocats: Lazega

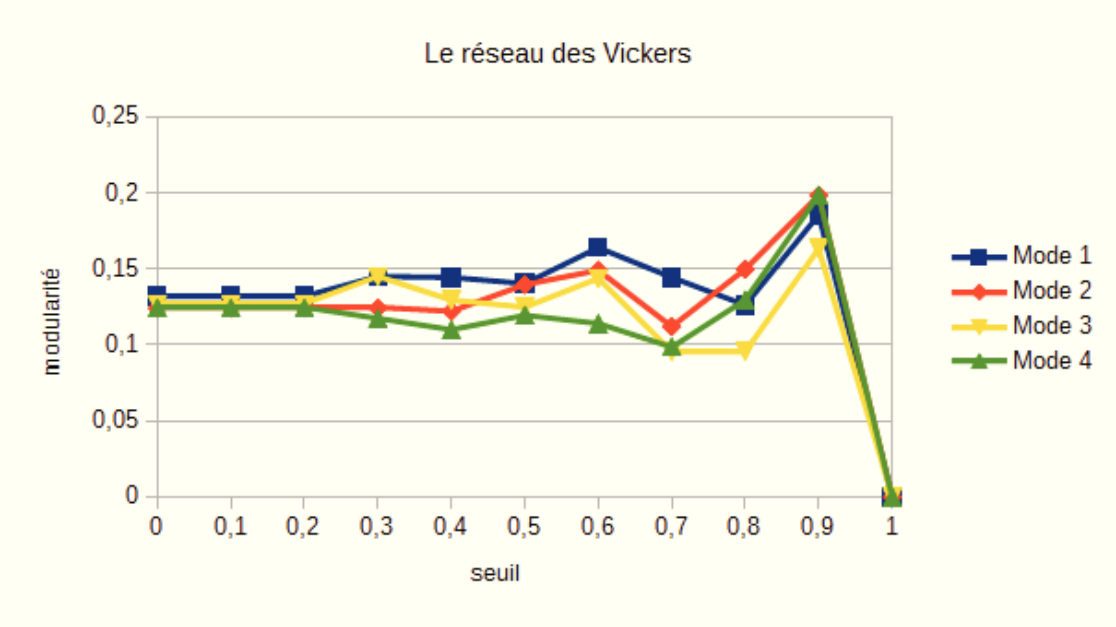

FIGURE 6 – Mesure de modularité sur le réseau des Vickers

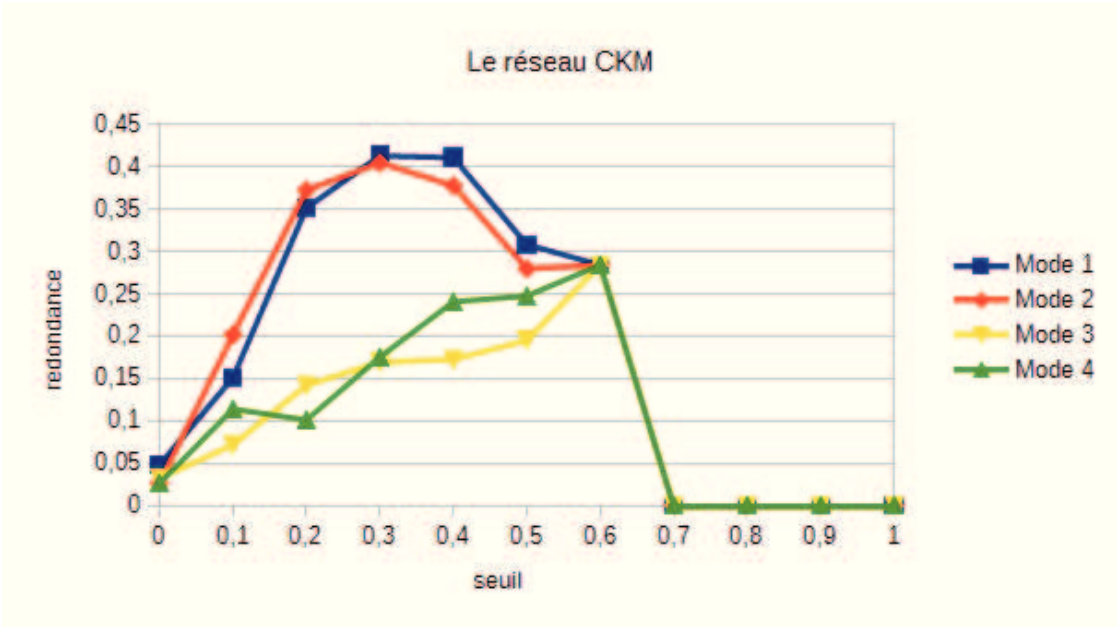

FIGURE 7 – Mesure de redondance sur le réseau d'innovations des médecins

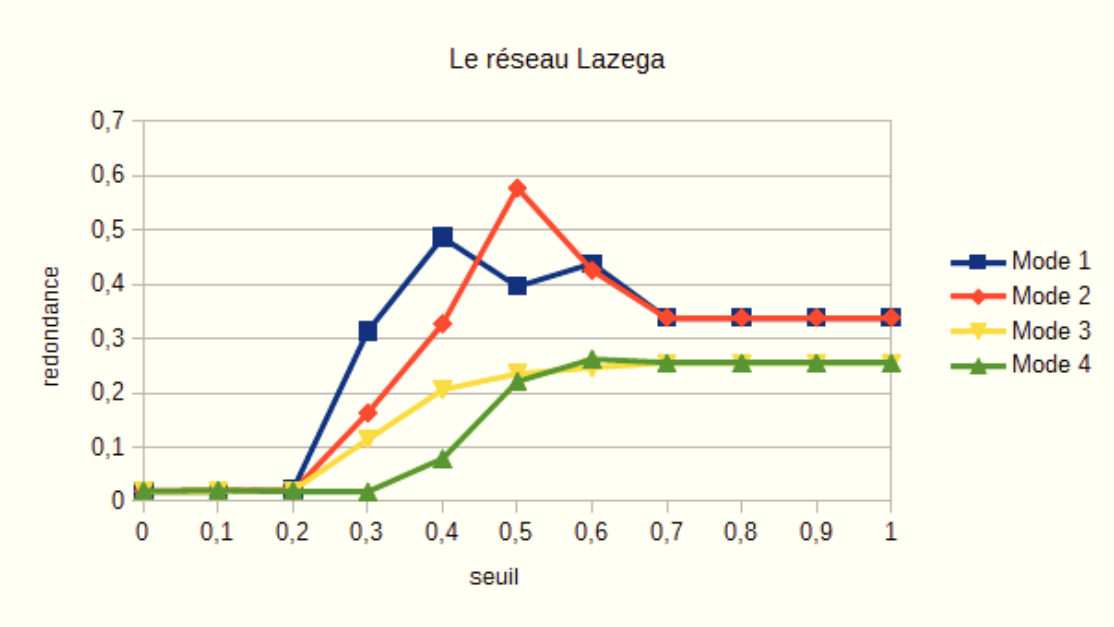

FIGURE 8 – Mesure de redondance sur le réseau de cabinet d'avocats

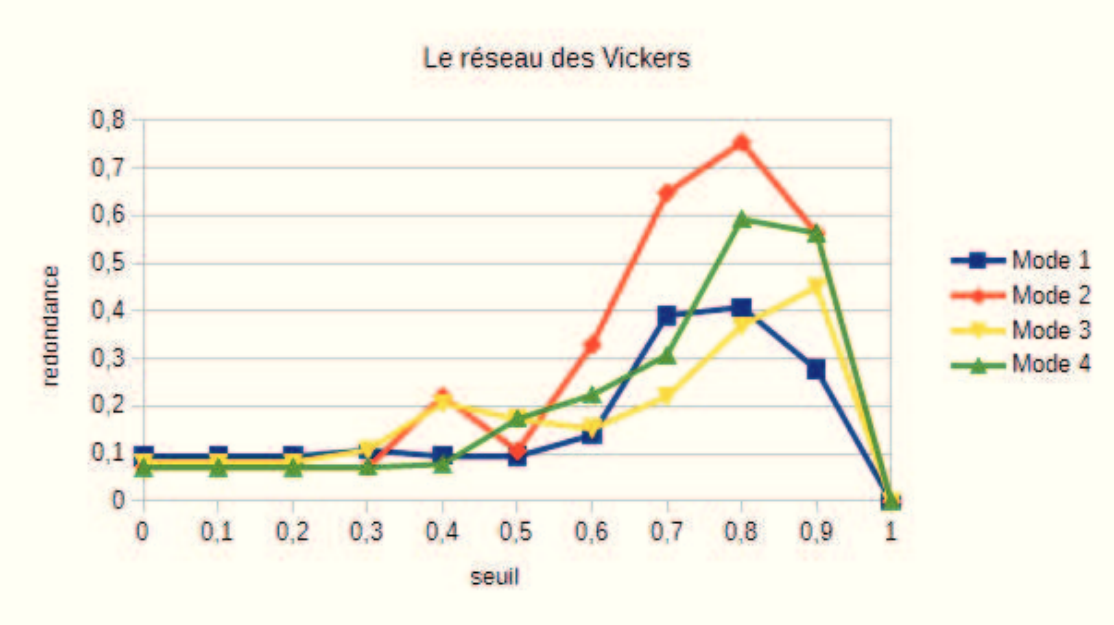

FIGURE 9 – Mesure de redondance sur le réseau des Vickers

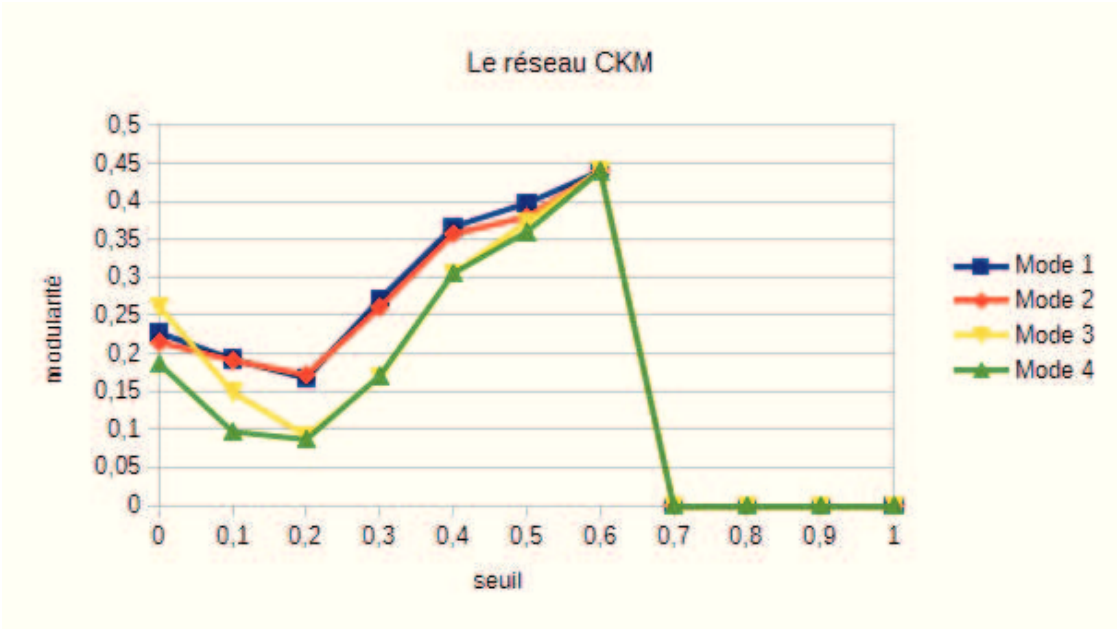

FIGURE 10 – Mesure de modularité sur le réseau d'innovations des médecins: CKM

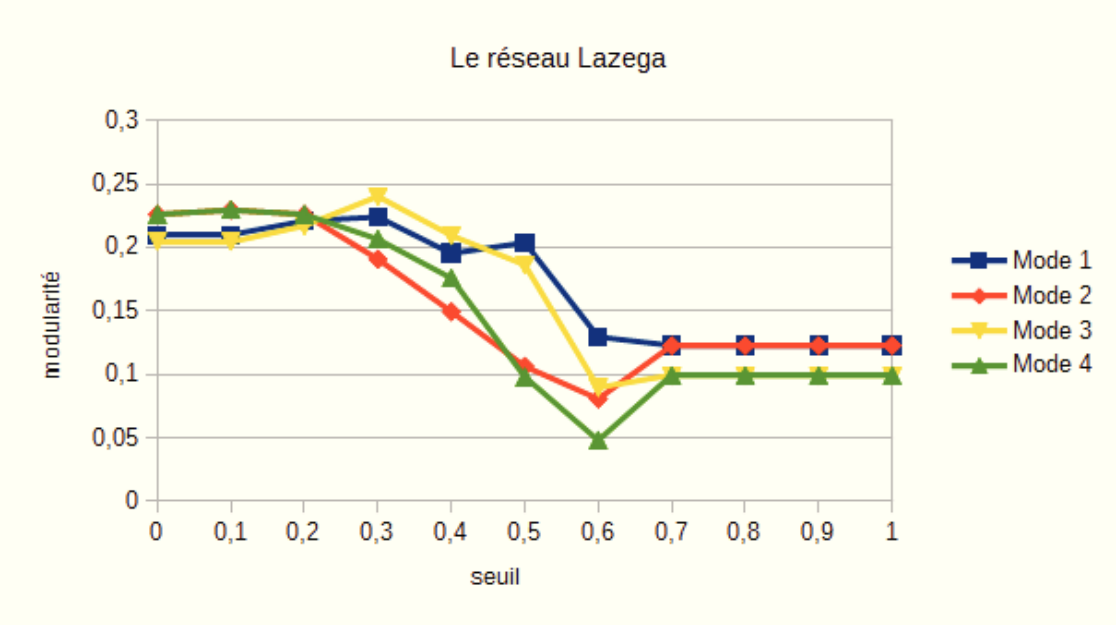

FIGURE 11 – Mesure de modularité sur le réseau de cabinet d'avocats: Lazega

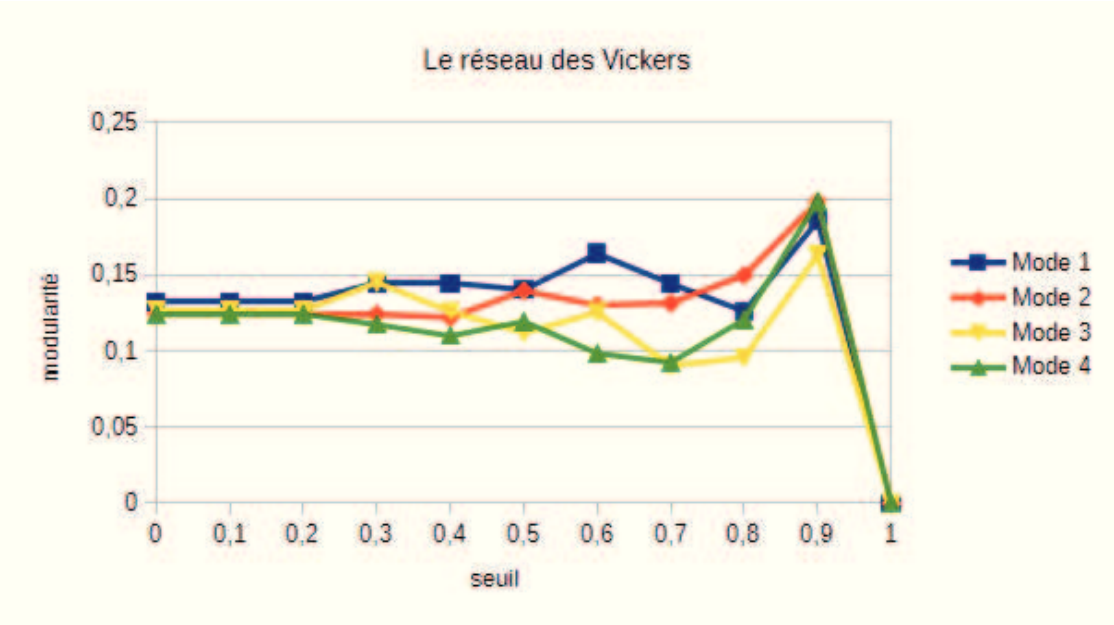

FIGURE 12 – Mesure de modularité sur le réseau des Vickers

Manel HMIMIDA

Une nouvelle approche topologique pour la recommandation de tags dans les folksonomies

Résumé:
Nous nous intéressons dans cette thèse à la problématique de recommandation de tags dans les systèmes de partage et de classification sociale des ressources, dits *folksonomies*. Les utilisateurs annotent les ressources à partager par des *tags* librement choisis. Or, la liberté de choix de tags les rend *ambigus*. Nous proposons une nouvelle approche *topologique* nommée TLTR (Two Level Tag Recommendation)pour la recommandation de tags. TLTR est basée sur une approche originale de compression des graphes. Le graphe d'une folksonomie est compressé en appliquant une méthode de clustering sur chacune des trois composantes d'une folksonomie, à savoir: l'ensemble des utilisateurs, l'ensemble des ressources et l'ensemble des tags. Nous proposons également une méthode de clustering topologique basée sur une approche centrée graine pour la détection des communautés dans les graphes multiplexes. Une approche topologique classique, en l'occurrence la méthode *Folkrank*, est appliquée sur le graphe réduit afin de sélectionner les clusters de tags les plus appropriés. Ces clusters sont ensuite utilisés pour construire un autre graphe contextuel extrait du graphe original représentant la folksonomie. La méthode *Folkrank* est à nouveau appliquée afin de calculer la liste de tags à recommander. Des expérimentations sur de grandes folksonomies, notamment, des jeux de données extraits du système de partage des références bibliographiques *Bibsonomy* montrent la pertinence de notre approche.
Mots clés:
Recommandation de tags, Réseaux complexes, Réseaux multiplexes, TLTR, Détection des communautés.

Abstract:
We focus in this thesis on the problem of tag recommendation in social sharing to classification systems called *folksonomies*. Users of a folksonomy annotate their resources with *freely* tags chosen. We propose here a new *topological* approach for tags recommendation called TLTR (Two Level Tag Recommendation). TLTR is based on an original approach of graph compression. The graph of a folksonomy is compressed by a clustering each of the three components, namely the set of users, resources and tags. A topological clustering method based on a seed-centered approach for community detection in multiplex graphs is proposed. A classical topological approach, namely *Folkrank*, is applied to the reduced graph to select the most appropriate clusters of tags. These clusters are then used to build another contextual graph extracted from the original graph representing the folksonomy. *Folkrank* method is applied again to compute the list of tags to recommend. Experiments on large folksonomy, including, data extracted from references system *Bibsonomy* show the relevance of our approach.
Keywords:
Tags Recommendation, Complex network, Multiplex network, TLTR, Community detection.