# Détection de communautés dans les réseaux d'information utilisant liens et attributs

David Combe

Université Jean Monnet
SAINT-ETIENNE

# Détection de communautés dans les réseaux d'information utilisant liens et attributs

Thèse présentée devant

**l'Université Jean Monnet**

pour obtenir le grade de

**Docteur en informatique**

par

**David COMBE**

soutenue le 15 octobre 2013 devant le jury composé de

| | | |
|---|---|---|
| M. Hamamache KHEDDOUCI | Professeur des Universités | Rapporteur |
| M. Emmanuel VIENNET | Professeur des Universités | Rapporteur |
| M. Pierre MARET | Professeur des Universités | Examinateur |
| Mme Christine LARGERON | Professeur des Universités | Directrice |
| M. Előd EGYED-ZSIGMOND | Maître de conférences | Co-directeur |
| M. Mathias GÉRY | Maître de conférences | Co-encadrant |

Année 2013 École doctorale "Sciences, Ingénierie, Santé" 488

Alors que les réseaux sociaux s'attachent à représenter des entités et les relations existant entre elles, les réseaux d'information intègrent également des attributs décrivant ces entités ; ce qui conduit à revisiter les méthodes d'analyse et de fouille de ces réseaux. Dans ces travaux, nous proposons des méthodes de classification des entités du réseau d'information qui exploitent d'une part les relations entre celles-ci et d'autre part les attributs les caractérisant. Nous nous penchons sur le cas des réseaux à vecteurs d'attributs, où les entités du réseau sont décrites par des vecteurs numériques. Ainsi nous proposons des approches basées sur des techniques reconnues pour chaque type d'information, faisant appel notamment à l'inertie pour la classification automatique et à la modularité de Newman et Girvan pour la détection de communautés. Nous évaluons nos propositions sur des réseaux issus de données bibliographiques, faisant usage en particulier d'information textuelle. Nous évaluons également nos approches face à diverses évolutions du réseau, notamment au regard d'une détérioration des informations des liens et des attributs, et nous caractérisons la robustesse de nos méthodes à celle-ci.

---

While social networks use to represent entities and relationships between them, information networks also include attributes describing these entities, leading to review the analysis and mining methods for these networks. In this work, we discuss classification of the entities in an information network. Classification operate simultaneously on the relationships and on the attributes characterizing the entities. We look at the case of attributed graphs where entities are described by numerical feature vectors. We propose approaches based on proven classification techniques for each type of information, including the inertia for machine learning and Newman and Girvan's modularity for community detection. We evaluate our proposals on networks from bibliographic data, using textual information. We also evaluate our methods against various changes in the network, such as a deterioration of the relational or vector data, mesuring the robustness of our methods to them.

Ce travail a été partiellement soutenu par la Région Rhône-Alpes (`http://www.rhonealpes.fr`).

# Remerciements

Je remercie d'abord Christine Largeron, ma directrice, sans laquelle rien n'aurait été possible. J'ai pu apprécier sa détermination dans les moments où j'ai eu le plus d'incertitudes, ainsi que son ouverture d'esprit sur le plan scientifique. Je remercie ensuite Előd Egyed-Zsigmond, co-directeur, qui n'a jamais hésité à braver les kilomètres entre Lyon et Saint-Etienne. Il a toujours été disponible, en face à face comme à distance, de bon conseil, et un soutien. Je remercie aussi Mathias Géry, co-encadrant, dont les avis et conseils ont toujours fait l'objet de beaucoup d'attention de la part de nous tous, car il ne parle jamais pour ne rien dire.

Je remercie les personnes qui m'ont fait l'honneur de prendre part au jury. Je remercie ainsi Emmanuel Viennet, Hamamache Kedhoucci pour leur lecture attentive du manuscrit et leurs remarques pertinentes. Enfin, je remercie Pierre Maret de bien avoir voulu présider ce jury.

Je tiens à remercier les très nombreuses personnes avec lesquelles j'ai partagé mes deux bureaux. D'abord Jean-Philippe, Aurélien et Fabien déjà avec moi dans les amphis de la Métare, mais aussi Émilie, Stéphanie, Christophe, Laurent et Frédéric, qui m'ont accueilli, m'ont beaucoup appris, ainsi que Chahrazed, Tung, Mattias, Vladimer, Hao, Michael, Émilie, Taygun, Nidhal, Aytaç, Stéphanie, Natacha, Adrien, Johan, Juri, Jan-Willem et Bert-Jan. Rien que ça. J'ai apprécié la rencontre de personnes d'horizons aussi variés pendant ce travail, dans le laboratoire mais aussi au cours des quelques conférences auxquelles j'ai participé.

J'ajoute un grand merci pour mes professeurs de la Faculté des Sciences, Catherine, Fabrice, Marc S., Marc B., Baptiste, François, Philippe, Thierry, car il y a certainement un petit bout de chacun d'eux dans dans le manuscrit, ainsi qu'à ceux qui sont arrivés un peu tard pour que je les connaisse devant un tableau noir, Leonor, Élisa, Rémi et Amaury. Je remercie aussi Colin de la Higuera et Jean-Christophe Janodet qui ne sont pas pour rien dans le fait que j'ai finalement réalisé ce travail. Je remercie également tout le personnel du laboratoire Hubert Curien.

Je remercie JC, Charlotte, Nicolas B., Nicolas D. et Jordan de m'avoir changé les idées, ma famille, qui a enduré certainement un peu de mon stress durant cette période, et mon papa de m'avoir fait la surprise de venir à la soutenance.

# Table des matières

# Table des figures

# Liste des tableaux

# Introduction

Nous tissons au quotidien des liens de différentes natures avec des personnes. Ces liens existent au sein de la sphère familiale ou professionnelle, les liens *forts*, ou avec des personnes avec lesquelles nous n'aurons communiqué qu'une seule fois, les liens *faibles*. Toutes ces relations, considérées collectivement, constituent des réseaux sociaux. Ces réseaux ont depuis longtemps fait l'objet d'études notamment en sciences sociales par des sociologues, des comportementalistes, des économistes, etc. Ainsi Wasserman définit un réseau social comme un ensemble d'acteurs et la donnée des relations existant entre eux (Wasserman et Faust, 1994b). On considère que ces réseaux sont le reflet d'une organisation où, à son échelle, chaque acteur du réseau est amené à créer des liens avec d'autres acteurs. L'analyse de ces liens peut permettre de prédire des caractéristiques des acteurs ou l'apparition de liens entre eux ou encore de connaître les modalités de diffusion dans le réseau. On peut aussi chercher à détecter des groupes d'acteurs fortement connectés entre eux. C'est ce sujet qui est au cœur de cette thèse où nous nous intéressons à la détection de communautés dans les réseaux.

L'avènement des réseaux sociaux en ligne a conduit à un regain d'intérêt pour leur analyse y compris en informatique. Ces réseaux de l'internet ont permis de valider à plus grande échelle des théories émises en sociologie comme par exemple la théorie des six degrés de séparation de Milgram ou la notion de réseau "petit monde" qui ont façonné la vision que nous portons sur les graphes sociaux (Milgram, 1967).

Mais cet intérêt est dû aussi à la disponibilité d'information portant non seulement sur les relations qui existent entre les acteurs, mais aussi de données permettant de décrire ou de caractériser ces derniers. Les techniques et services du Web 2.0 permettent en effet aux utilisateurs de certains sites internet de devenir producteurs ou consommateurs d'information et d'entrer en relation avec les autres internautes en déclarant par exemple leurs caractéristiques (âge, sexe, etc.). De plus en plus souvent, on dispose donc de réseaux où les acteurs sont non seulement reliés entre eux mais ont également des informations attachées, telles qu'un profil d'utilisateur ou un contenu produit. Ces données attachées peuvent être des étiquettes, des vecteurs numériques, du contenu textuel, etc. Ces réseaux enrichis, désignés par le nom de réseaux d'information, peuvent être représentés par un graphe dont les sommets sont décrits par des attributs.

De tels réseaux ne sont que partiellement analysés par les méthodes traditionnelles. En effet, si la détection de communautés dans un graphe a fait l'objet de nombreuses recherches ayant abouti à plusieurs méthodes et algorithmes, celles-ci ne prennent pas en compte en général les attributs décrivant les sommets du graphe.

Par ailleurs la classification automatique, dont l'objet est de regrouper les éléments ayant les mêmes caractéristiques au regard d'une mesure de similarité, a également donné lieu à de nombreux travaux, mais les méthodes de classification automatique ne permettent pas de tirer parti, en plus des valeurs associées aux éléments, de leurs connexions dans un graphe.

C'est la raison pour laquelle des recherches récentes ont été consacrées à la détection de communautés exploitant données relationnelles et attributs. En effet, la prise en compte conjointe des deux types de données soulève des questions nouvelles liées à la façon de tirer le meilleur parti de l'ensemble des données. Cette approche est notamment justifiée par le phénomène d'homophilie, qui traduit la tendance qu'ont les individus à se lier avec d'autres individus aux caractéristiques similaires. Nous verrons dans cette thèse dans quelle mesure nous pouvons améliorer le processus de détection de communautés en combinant les deux types d'informations.

Dans le chapitre 1, nous présenterons le contexte de ce travail en introduisant les notions de graphe, de réseau social, de réseau d'information et nous expliquerons comment des données bibliographiques peuvent être exploitées pour construire un réseau d'information.

Le chapitre 2 est dédié à l'état de l'art. Nous commencerons par étudier les méthodes classiques de classification non supervisée et détaillerons les modes d'évaluation. Nous ferons de même pour la détection de communautés dans les graphes, qui dispose de ses critères, méthodes, et modes d'évaluation propres. Nous présenterons ensuite l'état de l'art de la détection de communautés dans des réseaux où les sommets sont décrits par des attributs et distinguerons quatre familles de méthodes permettant de traiter ce problème.

Dans le chapitre 3, nous présentons notre première proposition, ToTeM, une méthode de détection de communautés qui étend la méthode de Louvain de façon à prendre en compte les attributs. La méthode de Louvain utilise comme critère d'optimisation la modularité de Newman et Girvan (Newman et Girvan, 2004), une mesure de la qualité d'une partition des sommets d'un graphe. Nous proposons d'adjoindre à cette mesure l'inertie interclasses, qui mesure la qualité d'une partition pour la classification automatique, de façon à opérer un partitionnement selon les deux critères. On verra cependant que ces deux critères ont des propriétés et des valeurs limites différentes, qui peuvent laisser penser qu'un critère peut prendre le pas sur l'autre lors de la classification. Nous testerons notre méthode sur des réseaux artificiels et un réseau réel que nous avons construit à partir d'informations bibliographiques.

Dans le chapitre 4, pour répondre au problème précédent concernant le critère d'optimisation, nous faisons notre deuxième proposition, 2Mod-Louvain. Cette méthode est aussi une extension de la méthode de Louvain reposant sur une approche

différente pour la prise en compte des attributs. Là où l'utilisation d'un critère joignant modularité de Newman et Girvan et inertie interclasses pouvait poser des problèmes de normalisation, nous proposons de remplacer l'inertie par un critère que nous avons construit, la modularité basée sur l'inertie. Celle-ci est inspirée de la modularité de Newman et Girvan mais, là où cette dernière est fondée sur des notions d'arêtes et de degrés, la modularité basée sur l'inertie utilise des distances et la notion d'inertie. Ce faisant, nous traiterons le problème de la pondération des deux informations en y apportant un éclairage nouveau.

Nous conclurons alors avec le chapitre 5, qui présentera notamment les perspectives que nous donnons à ces travaux.

Enfin figure en annexe une étude complémentaire que nous avons effectuée pour évaluer et comparer différents programmes et librairies d'analyse de réseaux sociaux qui nous ont été utiles dans le cadre de nos recherches.

CHAPITRE 1

# Du réseau social au réseau d'information

**Sommaire**



## 1.1 Introduction

Dans ce chapitre, nous présentons d'abord le contexte des réseaux sociaux en général, dans la section 1.2. Nous introduisons également les concepts de la théorie des graphes qui nous serons utiles par la suite. Nous aborderons alors la notion de réseau d'information dans la section 1.3. Nous terminerons ce chapitre par les notions propres aux bases bibliographiques, qui constituent le domaine d'application privilégié de nos travaux. Dans la section 1.4, nous décrirons aussi les étapes nécessaires pour construire un réseau d'information à partir d'une base bibliographique, qui sera utilisé à des fins d'évaluation dans le chapitre 3.

## 1.2 Réseau social et graphe

Wasserman décrit un réseau social comme un ensemble fini d'acteurs ainsi que les relations définies entre eux (Wasserman et Faust, 1994b). On pourra prendre l'exemple du réseau social des élèves d'un collège qui ont déjà été dans la même classe. Les acteurs seront alors tous les élèves du collège. La relation entre deux éléments sera alors "ont déjà été dans la même classe". Cette définition fait référence de façon directe à la notion mathématique de relation et à celle de graphe utilisée pour représenter le réseau.

### 1.2.1 Notions relatives aux graphes

On considère un graphe $G = (V, E)$ où $V$ est l'ensemble des sommets et $E \subseteq V \times V$ est l'ensemble des arêtes. On s'intéressera dans ce document, sauf indication contraire, aux graphes non orientés, lesquels décrivent une relation symétrique entre les sommets.

Les *sommets* sont les objets, au sens général du terme, qui sont en relation dans le graphe. On parle souvent aussi de *nœuds* ou d'*acteurs*. On notera $N$ le nombre de sommets de $G$, avec $N = |V|$.

Deux sommets $v$ et $v'$ sont *adjacents* si ils sont les extrémités d'une même arête du graphe, c'est-à-dire si $(v, v') \in E$.

Les *arêtes* décrivent les relations entre les sommets du graphe. Une arête relie deux sommets (éventuellement confondus) du graphe. Les arêtes peuvent être *valuées*, c'est-à-dire qu'une valeur leur est attribuée. Une valuation forte indique alors une relation de forte intensité. On dit que le graphe est *valué*. Sauf indication contraire, seuls des graphes ne comportant que des valuations positives sur leurs arêtes seront traités.

Une arête est *incidente* à un sommet si le sommet constitue une (ou deux) de ses extrémités.

La *matrice d'adjacence* $\mathcal{A}$ du graphe $G$ est la matrice carrée de côté $|V|$ dont le terme $\mathcal{A}[v, v']$ correspond à la valuation, que nous notons de façon simplifiée $\mathcal{A}_{v,v'}$, de l'arête éventuelle liant le sommet $v$ au sommet $v'$, ou 0 si $v$ et $v'$ ne sont pas adjacents.

On note $M$ la somme des valuations des arêtes de $G$ :

$$M = \sum_{(v,v') \in V \times V} \mathcal{A}_{v,v'} \tag{1.1}$$

Le *degré* $deg(v)$ d'un sommet $v \in V$ est le nombre d'arêtes adjacentes à $v$.

Dans un graphe valué, on préférera le plus souvent utiliser le *degré valué* qui tient compte de la valuation des arêtes :

$$k(v) = \sum_{v' \in V} \mathcal{A}_{vv'} \tag{1.2}$$

Une arête reliant un sommet $v$ avec lui-même est appelé *boucle*. Dans ce cas l'impact de la valuation de l'arête compte pour ses deux extrémités. La contribution de l'arête au degré du sommet est donc double dans le degré de $v$.

Un graphe est dit *complet* si tous ses sommets sont adjacents deux à deux :

$$\forall (v, v') \in V \times V, (v, v') \in E \tag{1.3}$$

Un *sous-graphe* $G' = (V', E')$ de $G$, avec $V' \subset V$, $E' \subset E$ est composé des sommets de V' et des arêtes de $E$ ayant leurs deux extrémités dans $V'$.

Une *clique* $G' = (V', E')$ est un sous-graphe de $G$ où tous les couples de sommets de $V'$ sont reliés par une arête, c'est donc un sous-graphe complet de $G$.

Un *graphe biparti* est un graphe dont l'ensemble des sommets est divisé en deux sous-ensemble disjoints $V_1$ et $V_2$ et tel que chaque arête connecte un sommet de $V_1$ à un sommet de $V_2$.

Une *composante connexe* est un ensemble maximal de sommets tel qu'entre tout couple de sommets $(v_1, v_n)$ il existe un chemin, c'est-à-dire une succession de sommets $v_2, v_3, \ldots, v_{n-1}$ de $V$ avec $(v_i, v_{i+1}) \in E$, $\forall i = 1, \ldots, n-1$.

Un graphe est *non-orienté* si $\forall (v, v') \in E$, $(v', v) \in E$, c'est-à-dire si les arêtes sont faites de paires de sommets non ordonnées. Si les arêtes sont présentes sous forme de couples de sommets, avec une origine et une destination, alors le graphe est orienté.

### 1.2.2 Distances dans un graphe

La notion de distance entre objets étant d'une importance primordiale dans toute tâche de classification, nous allons voir quelles sont les mesures applicables à des sommets d'un graphe.

Dans un graphe non valué, la longueur du plus court chemin entre deux sommets $v$ et $v'$ de $V$ correspond au nombre d'arêtes qu'il faut traverser au minimum pour joindre $v$ et $v'$. Elle est appelée distance géodésique entre $v$ et $v'$.

Dans un graphe valué, la distance du plus court chemin est la somme minimale des valuations des arêtes nécessaires pour joindre les sommets $v$ et $v'$.

Cette distance est bien plus coûteuse en calcul, *a fortiori* si l'on n'impose pas des valuations positives sur chaque arête. De plus, dans cette configuration peuvent apparaître des cycles infinis de coût négatif. En pratique, on pourra alors borner les distances minimum. Inversement, dans le cas où il est impossible de relier deux sommets, on pourra borner la distance maximum entre deux sommets.

### 1.2.3 Mesures de centralité

Nous appellerons indicateur toute mesure de nature quantitative qui peut être calculée à partir des sommets, des arêtes, des ensembles de sommets ou d'arêtes ou encore du graphe lui-même.

Parmi ces indicateurs, figurent les mesures de centralité qui visent à évaluer des propriétés souvent abstraites des entités du réseau social. On distingue la centralité de proximité, de prestige, de pouvoir, de cohésion, etc (Freeman, 1979).

De nombreux auteurs ont parlé des centralités sans qu'une définition consensuelle existe. On peut néanmoins s'intéresser à la nomenclature de Koschutzki *et al.* qui propose une typologie de ces mesures selon les axes suivants (Koschützki et al., 2005) :

- l'accessibilité (*Reachability*), qui repose sur une notion de distance entre les sommets (degré, excentricité, proximité) ;
- l'écoulement (*Amount of flow*), qui repose sur une notion de flux circulant entre les sommets du graphe. On prendra pour exemple la centralité d'intermédiarité (Freeman, 1979) et les mesures qui font usage d'une marche aléatoire (PageRank (Page et al., 1999), HITS (Kleinberg, 1999)).
- la vitalité (*Vitality*), pour déterminer l'importance d'un sommet ou d'une arête dans un graphe en faisant la différence entre $f(G)$ et $f(G \setminus v_x)$, où la fonction $f()$ est une mesure quantitative caractérisant $G$ et $G \setminus v_x$ désigne le graphe $G$ privé du sommet $v_x$.
- la réaction (*Feedback*) où le score d'un sommet dépend implicitement des scores des autres sommets dans le réseau, comme dans l'indice de Katz (Katz, 1953).

Selon le cas, on va choisir une centralité appropriée au contexte d'application. Ainsi, on fera usage d'une centralité axée sur l'accessibilité pour choisir l'emplacement des services d'urgence tels que des casernes de pompiers. On choisira une centralité basée sur l'écoulement pour placer des capteurs pour mesurer la contamination d'un réseau d'eau. On utilisera la vitalité pour mesurer l'impact de la présence d'un service d'hôpital dans son environnement. Enfin on aura recours à la réaction pour mesurer l'impact de la mise à disposition d'un serveur miroir dans un réseau de serveurs $G$ sur la qualité de service perçue par les utilisateurs d'un site web.

#### 1.2.3.1 Centralité de degré

La mesure de la centralité la plus simple est le degré. Que le graphe soit valué ou pas, on mesure la somme de l'intensité de la connexion d'un sommet avec ses voisins directs.

$$C_D(v) = \sum_{v' \in V} \mathcal{A}_{v,v'} \tag{1.4}$$

Une version normalisée est proposée par Nieminen, pour laquelle le score est de 1 pour les sommets connectés à tous les autres sommets (Nieminen, 1974).

$$C'_D(v) = \frac{deg(v)}{|V| - 1} \tag{1.5}$$

Cette mesure est indiquée dans les situations où on peut assimiler l'importance d'un sommet à son activité potentielle de communication.

#### 1.2.3.2 Centralité d'intermédiarité (*Betweenness centrality*)

La centralité d'intermédiarité est une autre mesure de centralité d'un sommet dans un graphe. L'intermédiarité d'un sommet $u \in V$ est définie par :

$$C_B(u) = \sum_{v,v' \in V, v \neq v'} \frac{\varphi(v, v'|u)}{\varphi(v, v')} \tag{1.6}$$

où $\varphi(v, v')$ est le nombre de plus courts chemins passant du sommet $v$ au sommet $v'$ et $\varphi(v, v'|u)$ est le nombre de plus courts chemins du sommet $v$ au sommet $v'$ passant par $u$.

Les sommets qui se trouvent fréquemment sur les plus courts chemins entre deux autres sommets ont une intermédiarité plus grande que les autres (Freeman, 1979; Brandes, 2008).

L'intermédiarité peut également être définie pour une arête $e$ (*edge betweenness*) :

$$C_{EB}(e) = \sum_{(v,v') \in V \times V} \frac{\varphi_E(v, v'|e)}{\varphi(v, v')} \tag{1.7}$$

où $\varphi_E(v, v'|e)$ est le nombre de plus courts chemins du sommet $v$ au sommet $v'$ passant par l'arête $e$.

#### 1.2.3.3 Centralité de proximité (*Closeness centrality*)

Pour les graphes connexes, la centralité de proximité est l'inverse de la distance moyenne à tous les autres sommets. La centralité de proximité est plus grande pour les sommets qui sont à faible distance de tous les autres sommets. Ainsi, dans le monde réel, dans un contexte où les arêtes sont des rues et les sommets des carrefours, les carrefours ayant la plus grande centralité de proximité sont les meilleurs candidats pour accueillir des services d'urgence.

La centralité de proximité est définie par :

$$C_C(v) = \frac{1}{\Sigma_{v' \in V \setminus v} dist(v, v')} \tag{1.8}$$

où $dist(v, v')$ est une distance, telle que le nombre d'arêtes dans le plus court chemin

entre deux sommets ou la somme des valuations de ces arêtes pour les graphes valués. L'inverse de la centralité de proximité est appelé indice de Shimbel.

Borgatti *et al.* proposent une étude approfondie des notions attachées à la centralité (Borgatti, 2005).

## 1.3 Réseau d'information

Avec l'émergence du Web 2.0 et des réseaux numériques, la notion de réseau social a dû être généralisée pour tenir compte de caractéristiques décrivant les acteurs du réseau et leurs relations. Ceci a conduit à la définition de la notion de réseau d'information homogènes ou hétérogènes par Han (Sun et Han, 2012), celle de graphe d'information par Moser *et al.* (Moser et al., 2007) ou encore de graphe avec attributs par Zhou (Zhou et al., 2009).

Dans la suite, nous appellerons réseau d'information un réseau où chaque sommet est décrit par des données qui peuvent être structurées ou non structurées. Il peut s'agir de données numériques, sous la forme d'un ensemble ou plus communément d'un vecteur, de données textuelles, ou plus généralement de données de n'importe quel type. Un exemple d'un tel réseau est celui d'un site de microblogs où chaque utilisateur peut se lier d'amitié avec d'autres et où il se décrit par le biais d'une courte biographie (contenu textuel), de son âge et de sa taille (vecteur numérique), ou encore de ses centres d'intérêt à choisir dans une liste (étiquettes).

Nous verrons comment un tel réseau peut être dérivé d'un graphe où les sommets sont associés à des documents de nature textuelle dans la section 1.4.4.2.

## 1.4 Réseau bibliographique

De nombreuses sources de données peuvent être modèlisés sous la forme de réseaux d'information. On citera ainsi par exemple les fichiers logs de services web (Serrour et Kheddouci, 2010) ou de réseaux d'affiliation où les liens peuvent prendre différentes natures (Zhao et Getoor, 2006). Le domaine des réseaux bibliographiques est l'application principale de ce travail, bien que plusieurs de nos travaux se veuillent parfaitement transposables à d'autres domaines.

Cette section vise à préciser le vocabulaire et les notions propres à cette application.

### 1.4.1 Base de données bibliographique d'articles et/ou méta-données associées

Une base de données bibliographique est une compilation d'informations (de méta-données) sur un ensemble d'articles de recherche. Parmi ces méta-données, les plus fréquentes sont :

- des auteurs,
- des publications,
- des journaux,
- des conférences,
- des attributs temporels comme les années de publication.

Les bases de données bibliographiques sont issues de démarches très différentes. La qualité de leurs méta-données, le nombre de documents référencés et l'exploitabilité par tout un chacun des informations varient selon le type de la base.

En particulier, on peut distinguer trois grands types de bases de données bibliographiques :

- les catalogues bibliographiques des éditeurs comme ACM Digital Library [1], Springer [2], Elsevier [3], IEEE [4], etc. Ils sont le plus souvent créés à partir d'actes (comptes-rendus) de conférences ou des méta-données issues des journaux scientifiques de l'éditeur.
- les bases créées par leurs utilisateurs, parmi lesquelles Mendeley [5], Academia.edu [6], Zotero [7], CiteULike [8], Bibsonomy [9], etc. Les utilisateurs enregistrent eux-mêmes les méta-données des articles qui les intéressent et participent ainsi à l'enrichissement d'une bibliothèque commune.
- les bases professionnelles telles que DBLP [10], ISI Web of Science [11], etc. Ici les articles sont intégrés en lot, à partir d'actes, par un petit nombre de personnes autorisées. S'il est parfois possible de demander l'insertion d'actes en particulier, leur intégration effective passe par un superviseur.

Dans le but d'éviter les confusions, voici la définition qui sera retenue pour quelques termes relatifs aux données bibliographiques.

1. http://dl.acm.org/
2. http://link.springer.com/
3. http://www.sciencedirect.com
4. http://ieeexplore.ieee.org
5. http://www.mendeley.com/
6. http://academia.edu
7. http://www.zotero.org/
8. http://www.citeulike.org
9. http://www.bibsonomy.org
10. http://dblp.uni-trier.de
11. http://thomsonreuters.com/web-of-science

Un *article* de recherche est un document composé d'un titre, d'un résumé, d'un corps et d'une bibliographie. Il est associé à la liste de ses auteurs et à un événement comme un journal ou une conférence. Il a une date de publication (l'information est le plus souvent limitée à l'année).

Un *événement* est une revue ou une conférence, garant de la qualité et hôte d'une publication. Dans la suite, on décide que deux éditions d'une conférence (de deux années différentes) ou deux numéros d'une revue correspondent à un même événement.

### 1.4.2 Un exemple : la base DBLP

DBLP (Digital Bibliography & Library Project) est une base de données que nous avons utilisée pour nos expérimentations. Elle est maintenue par l'université allemande de Trier. Ce n'est pas un site de réseautage social dans la mesure où il n'y a pas de notion d'utilisateur dans le système. Les articles scientifiques eux-mêmes sont absents de la base. Cependant, les méta-données qu'elle contient permettent de créer le réseau social sous-jacent de collaborations scientifiques des auteurs, relatif à l'ensemble des articles présents dans la base.

DBLP est téléchargeable sous la forme d'un fichier XML comprenant la liste des enregistrements qui sont utilisés sur le site. Chaque enregistrement correspond aux métadonnées associées à une publication. La figure 1.1 montre la forme des enregistrements dans le fichier XML de la base DBLP mis à disposition et la table 1.1 quelques caractéristiques de cette base. DBLP ne fournit ni le résumé ni le contenu des articles.

| | |
|---|---|
| Taille de l'archive | 720 Mo |
| Nombre d'enregistrements | 2 246 044 |
| Nombre d'auteurs | 837 047 |

TABLE 1.1 – La base de données DBLP, au 19 juillet 2010

### 1.4.3 Relations de base déductibles d'une base de données bibliographique

On parle souvent de réseau bibliographique car, à l'instar de la communauté scientifique, qui constitue un réseau de collaborateurs parmi les plus vastes, les bases de données bibliographiques témoignent de la coopération dans le monde académique.

La transposition de la base de données bibliographique vers un réseau nécessite le choix d'une ou de plusieurs relations qui traduisent l'interaction sociale des personnes, déductibles des informations bibliographiques.

```
<article mdate="2010-06-01" key="journals/corr/abs-1005-1659">
<author>Brian Karrer</author>
<author>M. E. J. Newman</author>
<title>Random graphs containing arbitrary distributions
of subgraphs</title>
<ee>http://arxiv.org/abs/1005.1659</ee>
<year>2010</year>
<journal>CoRR</journal>
<volume>abs/1005.1659</volume>
</article>

<article mdate="2009-04-22" key="journals/corr/abs-0903-0419">
<author>Gourab Ghoshal</author>
<author>Vinko Zlatic</author>
<author>Guido Caldarelli</author>
<author>M. E. J. Newman</author>
<title>Random hypergraphs and their applications</title>
<ee>http://arxiv.org/abs/0903.0419</ee>
<year>2009</year>
<journal>CoRR</journal>
<volume>abs/0903.0419</volume>
</article>
```

FIGURE 1.1 – Extrait du fichier XML de la base de données DBLP

Dans ce document nous nous attacherons à trois relations particulièrement utilisées en matière de modélisation de bases de données bibliographiques, la citation et la coparticipation.

#### 1.4.3.1 Relation de citation

Une *citation* est un renvoi dans un article vers un article antérieur. L'ensemble des citations faites dans un article est appelé bibliographie de l'article et est usuellement placée à la fin de celui-ci.

On dit qu'un article $v$ cite un article $v'$ quand les références de l'article $v'$ sont présentes dans la bibliographie de l'article $v$.

On notera qu'un nombre très restreint de citations a été intégré à la base DBLP.

#### 1.4.3.2 Relation de coparticipation

On dit qu'un auteur $v$ a coparticipé avec un auteur $v'$ quand ils sont enregistrés tous les deux comme auteurs d'articles dans au moins un même événement (journal

ou conférence) sans forcément être co-auteurs.

#### 1.4.3.3 Relation de copublication

On dit de deux auteurs qu'ils ont *copublié* si ils figurent ensemble dans la liste des auteurs d'au moins un même article.

### 1.4.4 Construction du jeu de données des 4 sessions

Afin d'évaluer les algorithmes de détection de communautés que nous avons développés, nous avons été amenés à construire un jeu de données muni de plusieurs vérités terrain. Celles-ci seront utilisées de manière à évaluer la performance de la classification en fonction de partitions qui sont significatives selon les données textuelles ou relationnelles ou encore selon les deux types de données. Dans cette section nous décrivons la façon dont ce jeu de référence a été réalisé.

Les données utilisées pour construire ce réseau sont issues de DBLP pour les informations relationnelles (participation d'un auteur à une conférence, etc.) et des sites des deux conférences SAC 2009 et IJCAI 2009 pour les données textuelles.

Ces deux conférences ont chacune une session sur un même thème : la robotique. On considérera en outre une session supplémentaire dans SAC 2009 sur la bioinformatique, ainsi qu'une quatrième session dans IJCAI 2009 sur la logique par contraintes. Ces deux dernières sessions ont été choisies pour le fait qu'elles sont *a priori* différentes dans le vocabulaire employé et donc identifiables uniquement sur la base du texte qui y est rattaché.

Les effectifs des auteurs ayant participé à ces 4 sessions sont indiqués dans le tableau 1.2.

| Session et conférence de rattachement | Effectif |
|---|---|
| A Bioinformatique (SAC) | 24 |
| B Robotique (SAC) | 16 |
| C Robotique (IJCAI) | 38 |
| D Contraintes (IJCAI) | 21 |
| Effectif du jeu de données | 99 |

TABLE 1.2 – Effectif de chaque session

À partir de ces données, le problème de partitionnement peut consister à déterminer :

– la session d'un auteur : Bioinformatique à SAC (A), Robotique à SAC (B), Robotique à IJCAI (C) ou Contraintes à IJCAI (D)

– sa conférence : SAC (A $\cup$ B) ou IJCAI (C $\cup$ D)

– sa thématique : Bioinformatique (A), Robotique (B $\cup$ C) ou Contraintes (D)

Nous allons montrer dans la suite que selon les données utilisées (textuelles, relationnelles ou les deux), l'objectif est plus ou moins facile à atteindre. Pour ce faire, nous devons à partir de ces deux sources de données construire un réseau d'information $G = (V, E)$ dans lequel chaque sommet $v$ de $V$ correspond à un des 99 auteurs ayant participé à au moins une des quatre sessions.

#### 1.4.4.1 Données relationnelles utilisées pour construire le graphe $G = (V, E)$

Les données relationnelles, concernant la coparticipation à des conférences, sont issues de la base de données bibliographique DBLP. L'instantané utilisé date de juillet 2010. Les données utilisées concernent des événements (conférences et revues) enregistrés entre 2007 et 2009. Elles vont nous permettre de définir les relations entre les auteurs. Soient $v$ et $v'$ deux auteurs appartenant à $V$. S'il existe au moins un événement $e$ tel que $v$ et $v'$ sont auteurs d'articles publiés dans $e$ (même sans être coauteurs), alors $(v, v') \in E$.

#### 1.4.4.2 Représentation de l'information textuelle comme attributs numériques continus des sommets du graphe

À partir du graphe $G = (V, E)$ précédent, on se propose d'exploiter les articles publiés dans les quatre sessions par les auteurs pour associer à chacun d'eux un vecteur d'attributs textuels. Un document est construit pour chaque auteur. Celui-ci contient le titre et le résumé de chaque article dont il est l'auteur dans ces quatre sessions. Parmi ces données, qualifiées de *contextuelles*, aucune pondération n'est appliquée entre le titre et le résumé. Pour exemple, la figure 1.2 montre le contenu textuel associé à Gert Rickheit, auteur de *A Computational Model for the Alignment of Hierarchical Scene Representations in Human-Robot Interaction*.

Ces documents nécessitent un prétraitement aboutissant à l'obtention d'un vecteur caractérisant chaque auteur. La première étape de ce traitement consiste à déterminer la liste des mots à considérer dans notre vecteur d'attributs textuels. Il s'agira donc dans un premier temps d'éliminer la ponctuation, unifier la casse, décider du traitement particulier des mots composés.

Nous considérerons qu'un mot est une suite de lettres correspondant à l'expression régulière [a-zA-Z] de longueur supérieure à 2 afin d'éliminer les noms de variables courts dans les formules mathématiques que l'on peut trouver dans la littérature scientifique.

A Computational Model for the Alignment of Hierarchical Scene Representations in Human-Robot Interaction
The ultimate goal of human-robot interaction is to enable the robot to seamlessly communicate with a human in a natural human-like fashion. Most work in this field concentrates on the speech interpretation and gesture recognition side assuming that a propositional scene representation is available. Less work was dedicated to the extraction of relevant scene structures that underlies these propositions. As a consequence, most approaches are restricted to place recognition or simple table top settings and do not generalize to more complex room setups. In this paper, we propose a hierarchical spatial model that is empirically motivated from psycholinguistic studies. Using this model the robot is able to extract scene structures from a time-of-flight depth sensor and adjust its spatial scene representation by taking verbal statements about partial scene aspects into account. Without assuming any pre-known model of the specific room, we show that the system aligns its sensor-based room representation to a semantically meaningful representation typically used by the human descriptor.

FIGURE 1.2 – Exemple de document textuel attaché à un auteur

La classification de documents textuels est d'autant plus facile que le nombre de dimensions est faible (Yang et Pedersen, 1997). On évoque d'ailleurs souvent le phénomène dit de "malédiction de la dimensionnalité". Un nombre élevé de dimensions va non seulement rendre la classification plus longue, mais aussi souvent plus imprécise, car les dimensions sont alors moins expressives. Comme chaque mot correspond à une dimension du vecteur, pour éliminer des dimensions sans perdre en expressivité des données, on va éliminer les mots les moins expressifs. Ces mots, servant souvent de connecteurs linguistiques, et donc peu utiles pour décider de la similitude sémantique de deux phrases, sont appelés *mots vides*. La liste des mots vides que nous utilisons est celle fournie par NLTK [12] (Loper et Bird, 2002).

La figure 1.3 montre le texte de l'exemple de la figure 1.2 après élimination des mots vides.

La *lemmatisation* consiste ensuite à ramener chaque mot à une racine. De cette façon, on peut diminuer la taille du lexique et éventuellement améliorer l'efficacité de traitements comme la classification de documents en considérant les mots de même

12. `http://nltk.googlecode.com/svn/trunk/nltk_data/packages/corpora/stopwords.zip`

computational model alignment hierarchical scene representations human robot interaction ultimate goal human robot interaction enable robot seamlessly communicate human natural human like fashion work field concentrates speech interpretation gesture recognition side assuming propositional scene representation available less work dedicated extraction relevant scene structures underlies propositions consequence approaches restricted place recognition simple table top settings generalize complex room setups paper propose hierarchical spatial model empirically motivated psycholinguistic studies using model robot able extract scene structures time flight depth sensor adjust spatial scene representation taking verbal statements partial scene aspects account without assuming pre known model specific room show system aligns sensor based room representation semantically meaningful representation typically used human descriptor

FIGURE 1.3 – Exemple de document textuel attaché à un auteur après élimination des mots vides

famille comme porteurs du même sens.

L'algorithme utilisé est celui de Porter[13] (Porter, 2006). Lors de ce traitement, les formes plurielles et conjuguées des mots deviennent confondues. On peut trouver un exemple de l'application de la lemmatisation en comparant les figures 1.3 et 1.4. L'algorithme de Porter est découpé en cinq étapes chacune composée d'un ensemble de règles simples, qui sont exécutées les unes à la suite des autres. Parmi celles-ci on peut trouver des substitutions relatives au pluriel de la langue anglaise telles que :

- SSES $\rightarrow$ SS : caresses $\rightarrow$ caress
- S $\rightarrow$ $\lrcorner$ : cats $\rightarrow$ cat

L'application de certaines règles est conditionnée à un nombre de syllabes minimum dans le mot, avant l'appartition du motif. Pour simplifier le calcul du nombre de syllabes, on compte le nombre de motifs "voyelle suivie d'une consonne". Ce nombre est appelé $vc$. Ainsi Porter dicte-t-il ces règles ayant trait à la grammaire. Ces règles sont de second niveau, ce qui explique que certains exemples ne viennent pas du dictionnaire :

- (vc>0) IZER $\rightarrow$ IZE : digitizer $\rightarrow$ digitize
- (vc>0) OUSLI $\rightarrow$ OUS : analogousli $\rightarrow$ analogous
- (vc>0) IZATION $\rightarrow$ IZE : vietnamization $\rightarrow$ vietnamize

L'ensemble des mots retenus après élimination des mots vides et lemmatisation

13. http://nltk.googlecode.com/svn/trunk/doc/api/nltk.stem.porter.PorterStemmer-class.html

comput model align hierarch scene represent human robot interact ultim goal human robot interact enabl robot seamlessli commun human natur human like fashion work field concentr speech interpret gestur recognit side assum proposit scene represent avail less work dedic extract relev scene structur underli proposit consequ approach restrict place recognit simpl tabl top set gener complex room setup paper propos hierarch spatial model empir motiv psycholinguist studi use model robot abl extract scene structur time flight depth sensor adjust spatial scene represent take verbal statement partial scene aspect account without assum pre known model specif room show system align sensor base room represent semant meaning represent typic use human descriptor

FIGURE 1.4 – Exemple de document textuel attaché à un auteur après lemmatisation

forme l'*index* $T$. C'est un ensemble de mots jugés utiles pour décrire la collection des documents considérés. Il faut ensuite représenter chacun des documents de cette collection.

La représentation des informations textuelles a fait l'objet de nombreux travaux. Parmi eux, trois modèles se détachent. Ils ont pour objectif de saisir la sémantique d'une collection de documents dans un objectif de recherche d'information ou de fouille de textes.

**Le modèle booléen** Dans le modèle booléen, un mot de l'index est considéré comme *présent* ou *absent* de chacun des documents du corpus.

**Modèles de langues - N-Grammes** Les modèles de langues ou N-grammes sont de type probabiliste. Ils assignent une probabilité à toute séquence de mots figurant dans un document. Les séquences de mots sont souvent courtes, deux mots pour le modèle bigram, trois mots pour le modèle trigram. Il est rare de dépasser trois mots. C'est un modèle très utilisé dans les applications liées au langage naturel, comme la reconnaissance de l'écriture manuscrite ou la reconnaissance de la parole, car son intérêt est de pouvoir mesurer la probabilité d'être face à un terme particulier compte tenu de l'historique des termes rencontrés.

**Le modèle vectoriel** Le modèle que nous avons retenu dans nos recherches est le modèle vectoriel. Pour chaque mot de l'index, il tient compte de son nombre d'apparitions dans le document.

De plus, ce modèle permet d'appréhender la distribution des mots au sens de la Loi de Zipf. Cette loi empirique dit que le mot le plus courant est utilisé environ dix fois plus souvent que le dixième mot. C'est aussi le modèle le plus utilisé dans les produits commerciaux. Dans ce modèle, on représente un document $d_i$ d'une collection $\mathcal{D}$ sous la forme d'un vecteur :

$$d_i = (v_{i,j}, j \in T) \tag{1.9}$$

où $v_{i,j}$ représente le poids attribué au terme $t_j$ de l'index $T$ dans le document $d_i$. Dans sa version la plus simple il s'agit du nombre d'occurrences du terme $t_j$ dans le document.

Il est à noter que la représentation vectorielle, issue d'un sac de mots, provoque d'abord la perte de l'information de l'ordre des termes dans les documents. De plus, la ponctuation et la proximité de termes sont aussi perdues.

Toutes les collections de documents ont des distributions de mots différentes. La nature scientifique de nos données peut provoquer des changements importants sur la distribution de certains mots par rapport à d'autres collections.

Pour tenir compte de ces spécificités, au lieu de prendre comme poids d'un mot sa fréquence, on préfère utiliser une formule *tf-idf* dans le modèle vectoriel.

L'*idf* ou *Inverse Document Frequency* décrit le pouvoir discriminant d'un terme $t_i$ de l'index. C'est le rapport entre le nombre de documents dans la collection et le nombre de documents où le terme apparaît :

$$idf(j) = log\left(\frac{|D|}{\left|d_i : t_j \in d_i\right|}\right) \tag{1.10}$$

où $|D|$ est le nombre de documents dans la collection et $\left|d_i : t_j \in d_i\right|$ est le nombre de documents où apparaît le terme $t_j$.

La mesure *tf* (ou *Term Frequency*) décrit l'importance d'un terme dans un document. C'est le rapport entre le nombre d'occurrences du terme dans le document et le nombre de mots total du document.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,k}} \tag{1.11}$$

où $n_{i,j}$ est le nombre d'occurrences du terme $t_j$ dans le document $d_i$ et $\sum_k n_{i,k}$ est la somme du nombre total d'occurrences des termes formant $d_i$, soit la longueur du document en nombre de mots.

Le poids $v_{i,j}$ associé à un document $d_i$ est alors défini par le produit des deux mesures précédentes de sorte que $d_i$ est représenté par :

$$d_i = (v_{i,j}, j \in T) \text{ avec } v_{i,j} = tf_{i,j} \cdot idf(j) \tag{1.12}$$

Dans la suite, pour comparer deux documents au moyen de distances entre leurs représentations vectorielles, nous utiliserons la distance euclidienne ou la distance du cosinus.

Rappelons que la distance euclidienne entre deux documents $d_i$ et $d_k$ est définie par :

$$d(d_i, d_k) = \sqrt{\sum_{j \in T} \left(d_{i,j} - d_{k,j}\right)^2} \tag{1.13}$$

On mesure la similarité du cosinus à l'aide de la formule suivante.

$$s_{cos}(d_i, d_k) = cos(d_i, d_k) = \frac{d_i \cdot d_k}{\|d_i\| \, \|d_k\|} \tag{1.14}$$

On peut en déduire la distance du cosinus entre $d_i$ et $d_k$, définie par :

$$d_{cos}(d_i, d_k) = 1 - \frac{d_i \cdot d_k}{\|d_i\| \, \|d_k\|} \tag{1.15}$$

Si les vecteurs sont identiques ou ont même direction et sens, alors la distance entre les deux documents est nulle. Dans les autres cas, la distance entre les deux éléments est la valeur du cosinus de l'angle $\theta$ formé par leurs vecteurs. Cette valeur est toujours positive, les fréquences de termes ou pondération *tf-idf*, étant elles-mêmes positives.

À l'issue de ce traitement des documents, on dispose d'un réseau d'information sous la forme d'un graphe $G = (V, E)$ composé de 99 sommets reliés par 2 623 arêtes et donc chaque sommet est associé à un vecteur réel ayant 1 040 composantes. Ce jeu de données sera utilisé lors de nos expérimentations.

## 1.5 Conclusion

Dans ce chapitre, les notions de réseau social et de réseau d'information ont été introduites. Les concepts de la théorie des graphes que nous exploiterons ont aussi été rappelés de même qu'un certain nombre de notions associées comme le degré et les centralités. Les approches de modélisation d'un réseau bibliographique ont également été définies. Celles-ci nous permettent de produire un graphe à partir des propriétés des entités d'un réseau bibliographique ainsi que d'associer à ces entités un contenu textuel.

Nous avons, à travers un réseau que nous exploiterons dans les chapitres 3 et 4,

exposé les différentes étapes permettant de transformer des données issues de bases bibliographiques et de sites de conférences en un réseau d'information où les attributs sont numériques.

Dans le chapitre suivant nous présenterons l'état de l'art relatif d'une part au domaine de la classification automatique et d'autre part à celui de la détection de communautés dans les graphes avant d'aborder le problème de la détection de communautés dans un réseau d'information. Nous étudierons aussi les modalités d'évaluation qui sont associées aux méthodes permettant de traiter ces tâches.

CHAPITRE 2

# Classification automatique et détection de communautés pour les réseaux d'information

**Sommaire**



## 2.1 Introduction

Dans ce chapitre, nous décrivons l'état de l'art dans les domaines de la classification automatique dans la section 2.2 et de la détection de communautés dans un graphe dans la section 2.3. Si ces deux domaines ont connu des évolutions relativement indépendantes, nous verrons aussi quels sont leurs points communs. Cette étape nous permettra de mieux comprendre les approches visant à détecter des communautés dans les réseaux d'information qui seront présentées dans la section 2.4.

## 2.2 Classification automatique

### 2.2.1 Principes et concepts de base

La classification automatique consiste à regrouper les objets au sens générique du terme (éléments, individus, documents, etc.) qui sont les plus semblables et à séparer les objets qui sont les plus différents. Cette activité de partitionnement divise ces objets en groupes appelés *classes* qui sont recherchées pour leur expressivité et/ou leur

utilité pour des tâches telles que la visualisation de grandes quantités de données, la détection de cibles pertinentes dans le domaine du marketing, le repérage de groupes d'individus ayant des caractéristiques communes, etc.

Dans la suite nous définissons formellement le problème de la classification automatique de la façon suivante.

Étant un ensemble d'éléments $V = \{v_1, \ldots, v_n\}$ décrits par leur représentation, l'objectif est de déterminer une partition $\mathscr{P} = \{C_1, \ldots, C_r\}$ de $V$ en $r$ classes de sorte que les éléments figurant dans une même classe soient proches vis-à-vis de leur représentation et d'un critère préalablement choisi tandis que des éléments différents soient affectés à des classes distinctes.

Les éléments peuvent être décrits par des attributs quantitatifs ou qualitatifs. Nous supposerons qu'ils sont représentés dans un espace vectoriel réel à $|T|$ dimensions.

L'ensemble $\mathscr{P} = \{C_1, \ldots, C_r\}$ forme une partition en $r$ classes de $V$ si les axiomes suivants sont vérifiés :

- $\bigcup_{k \in \{1,\ldots,r\}} C_k = V$
- $C_k \cap C_l = \emptyset$, $\forall\ 1 \leq k < l \leq r$
- $C_k \neq \emptyset$, $\forall k \in \{1, \ldots, r\}$

Le premier axiome implique que chaque élément de $V$ est affecté à une classe. Le second que les classes ne se recouvrent pas. Le troisième que chaque classe contient au moins un élément de $V$.

Dans certains cas, le second axiome n'est pas vérifiée et un élément peut appartenir à plusieurs classes avec un certain degré d'appartenance. On parle alors de classes empiètantes, ou encore de partitions floues en classification automatique (Banerjee et al., 2005; Ruspini, 1970). De même, en détection de communautés dans les graphes, on peut accepter qu'un sommet appartienne à plusieurs communautés formant ainsi des communautés recouvrantes (Baumes et al., 2005; Lancichinetti et al., 2009; Reichardt et Bornholdt, 2006; Sales-Pardo et al., 2007; Wang, 2012). Dans la suite de ce travail nous considérerons que les classes ou les communautés recherchées forment une partition au sens strict du terme.

Le nombre de partitions d'un ensemble de $N$ éléments se calcule comme le $N$-ième nombre de Bell. Celui-ci peut se calculer comme une somme de nombres dits de Stirling de seconde espèce :

$$B_N = \sum_{k=1}^{N} \left\{ {N \atop k} \right\} \lambda^k \tag{2.1}$$

$$= \left\{ {N \atop k} \right\} = \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{k}{j} j^N \tag{2.2}$$

ou alors par la convergence de la formule de Dobinski :

$$B_N = \frac{1}{e} \sum_{k=0}^{\infty} \frac{k^n}{k!} \tag{2.3}$$

Il est intéressant de faire le lien avec le monde des probabilités en soulignant que le $N$-ième nombre de Bell est aussi le moment d'ordre $N$ d'une loi de Poisson de paramètre $N$. Une distribution de probabilités peut en effet être vue comme la partition d'un ensemble.

Parmi toutes les partitions constructibles sur $V$, on distinguera la partition discrète $\mathscr{P}_D$ qui est la partition unique dans laquelle il y a autant de classes que d'éléments et où chaque classe contient un seul élément. De même on distingue la partition grossière $\mathscr{P}_G$ qui est la partition unique dans laquelle tous les objets font partie de la même classe.

Notons que la classification automatique, appelée aussi classement non supervisé, diffère du classement supervisé. Dans ce second cas, on connaît le nombre de classes et on dispose d'un échantillon d'éléments de la population $V$ appelé échantillon d'apprentissage, pour lesquels on connaît à la fois la représentation et la classe d'appartenance. On peut alors utiliser cette information pour élaborer une procédure permettant de déterminer la classe d'un élément quelconque de la population à partir de sa représentation.

Par contre, dans le cas de la classification non supervisé qui nous intéresse, on ne dispose pas d'un échantillon d'apprentissage et le nombre de classes est le plus souvent inconnu. Néanmoins, certaines méthodes de classification ont besoin de cette dernière information comme paramètre.

### 2.2.2 Approches méthodologiques

Deux types de résultats peuvent être produits par un algorithme de classification automatique. Le premier est une partition, qui décrit uniquement les groupes d'éléments et fournit la classe d'affectation de chacun des éléments. Le second est une hiérarchie comprenant une suite de partitions. Dans ce cas, les différents niveaux de partitionnement, de différentes finesses, sont imbriqués les uns dans les autres. L'intérêt de cette approche est qu'elle permet de choisir plusieurs solutions en fonction du degré de finesse voulu.

Une hiérarchie est une famille totalement ordonnée de partitions $\mathcal{H} = \{\mathscr{P}_1, \ldots, \mathscr{P}_n\}$

telle que $\mathscr{P}_1$ est la partition discrète, $\mathscr{P}_N$ est la partition grossière et pour $i = 1, \ldots, N-1$ on a $\mathscr{P}_i$ est plus fine que $\mathscr{P}_{i+1}$ au sens de la comparaison des partitions.

En fonction du résultat produit, on distingue donc parmi les méthodes de classification non supervisées les méthodes hiérarchiques des méthodes non hiérarchiques.

Les premières peuvent être ascendantes si à partir de la partition discrète elles aboutissent par agglomérations successives à la partition grossière ou au contraire descendantes si elles consistent à procéder par division de la partition grossière jusqu'à la partition discrète. La seconde catégorie de méthodes, dites non hiérarchiques, regroupe celles qui peuvent fournir directement une partition. Ces dernières sont souvent itératives et leur exécution demande en général la connaissance a priori du nombre de classes à produire.

Dans la suite nous détaillons uniquement les méthodes auxquelles nous avons eu recours.

#### 2.2.2.1 Classification hiérarchique

La classification hiérarchique ascendante est une méthode de classification qui consiste, à partir de la partition discrète, à regrouper les classes les plus proches, en utilisant une distance entre éléments (voir section 1.4.4.2) et une fonction que l'on appelle mesure d'agrégation permettant de comparer des groupes d'éléments entre eux.

Le principe de la classification hiérarchique ascendante est décrit dans l'algorithme 1.

**Algorithme 1** : Classification hiérarchique ascendante

**Entrées** : un ensemble d'éléments $V$

**Sorties** : un ensemble de partitions contenant de $N$ à $1$ classes

1 on calcule la matrice des distances entre les éléments de $V$ ;
2 $\mathscr{P} \leftarrow$ partition discrète ;
3 **tant que** $|\mathscr{P}| \neq 1$ **faire**
4 $\quad \mathscr{P} \leftarrow$ fusionner les deux classes les plus proches de $\mathscr{P}$ au sens de la mesure d'agrégation ;
5 $\quad$ mettre à jour la matrice des mesures d'agrégation ;

Dans le cadre de la classification hiérarchique ascendante, le choix de la distance est à la discrétion de l'utilisateur. Celle-ci dépendra de la nature de la représentation des éléments. Dans le cas de vecteurs numériques, on utilisera souvent la distance

euclidienne ou dans le cas de documents décrits par des sacs de mots la distance du cosinus.

De plus la méthode requiert le choix d'un critère d'agrégation. Plusieurs critères d'agrégation se sont imposés au fil du temps.

Le lien minimum est une mesure d'agrégation qui associe à deux classes $C_k$ et $C_l$ le minimum des distances entre paires d'éléments composées d'un élément de chaque classe. Le lien maximum associe à deux classes $C_k$ et $C_l$ le maximum de ces distances.

La première mesure consiste à agréger les deux classes ayant les deux éléments les plus proches. La seconde mesure agrège les deux classes entre lesquelles les deux éléments les plus éloignés sont les plus proches.

$$S_{min}(C_k, C_l) = min_{v \in C_k, v' \in C_l} d(v, v') \tag{2.4}$$

$$S_{max}(C_k, C_l) = max_{v \in C_k, v' \in C_l} d(v, v') \tag{2.5}$$

Le lien moyen est une mesure d'agrégation qui utilise la moyenne arithmétique des distances (Sokal et Michener, 1958) :

$$S_{moy}(C_k, C_l) = \frac{1}{|C_k| \cdot |C_l|} \sum_{v \in C_k} \sum_{v \in C_l} d(v, v') \tag{2.6}$$

La mesure de Ward est aussi connue sous le nom de *construction hiérarchique du moment d'ordre deux* (Ward, 1963). Elle est définie par :

$$S_{Ward}(C_k, C_l) = \frac{m_k \cdot m_l}{m_k + m_l} \cdot d(g_{C_k}, g_{C_l}) \tag{2.7}$$

où $m_k$ et $m_l$ sont les masses des deux classes, c'est-à-dire le nombre d'éléments qu'elles contiennent.

Cette mesure présente l'avantage de pouvoir être interprétée en terme d'optimisation d'inertie : elle conduit à maximiser l'inertie interclasses définie dans la section 2.2.3.1.

Il n'y a pas de critère d'agrégation fondamentalement meilleur que les autres. Le choix sera fait selon la nature des données ou des caractéristiques recherchées dans le résultat.

On préfère souvent la classification ascendante à la classification descendante pour une raison de complexité (Cailliez et al., 1976). En effet, pour une partition donnée, il existe moins de possibilités différentes de choisir deux classes à fusionner ($N \times (N-1)$) que de façons de scinder en deux l'une des classes.

#### 2.2.2.2 Partitionnement non-hiérarchique de type nuées dynamiques

Il est possible de classifier des éléments sans se placer dans le paradigme hiérarchique, notamment en adoptant une approche qui produit directement une partition. C'est le cas de la méthode des centres mobiles et de ses dérivés qui produisent des classes non pas par agrégation ou division, mais en partant d'une séparation initiale arbitraire des éléments en $k$ classes, qui est ensuite raffinée.

**Centres mobiles**

À partir des centres (souvent appelés *centroïdes*) de classes, le principe des centres mobiles consiste itérativement à affecter les individus au centre le plus proche puis à recalculer les centres (Forgy, 1965).

L'algorithme commence par sélectionner aléatoirement $k$ centres. Deux étapes sont ensuite répétées jusqu'à convergence :

– l'assignation de chaque élément à la classe ayant le centre le plus proche,

– la mise à jour du centre de chacune des classes.

On cherche à minimiser l'inertie intraclasses :

$$\arg\min_{\mathscr{P}} \sum_{C\in\mathscr{P}} m_k \sum_{v\in C} d\left(v - g_C\right)^2 \tag{2.8}$$

où $m_k$ représente la masse de la classe $C_k$.

On note que l'on distingue parfois l'algorithme des centres mobiles dû à Forgy de celui des K-means dû à MacQueen qui en est une variante où un seul élément est inséré à chaque itération et les centres sont recalculés à chaque insertion.

Les centres mobiles ont pour inconvénient le fait qu'il faut connaître $k$, le nombre de classes, à l'avance. De plus, l'algorithme est sensible à son initialisation, à savoir le choix des centres, ce qui le rend de plus non déterministe.

Il existe une variante plus longue, mais donnant souvent de meilleurs résultats de l'algorithme des K-Means intitulée les K-Means bissectifs (Steinbach et al., 2000). Cette adaptation consiste, pour un nombre de classes à produire supérieur à 2, à opérer récursivement la méthode des K-means, de façon à bénéficier des possibilités du modèle hiérarchique. Une classe est alors divisée en deux à chaque opération.

X-means est une autre variante des K-means qui ne nécessite pas de connaître à l'avance le nombre de classes à produire (Pelleg et Moore, 2000). Le principe est d'ajouter graduellement de nouveaux centroïdes et de mesurer si leur ajout est bénéfique pour la classification selon un critère statistique. Pour cela, deux méthodes sont proposées. La première consiste à prendre un centre, puis à introduire un nouveau centre dans son voisinage immédiat, et à regarder si le modèle produit est meilleur

que le modèle précédent. Si c'est le cas, le nouveau centre est conservé. La deuxième méthode consiste à retenir un nombre significatif (on propose alors la moitié) des centres existants, sélectionnés selon une heuristique pour bien se prêter à un "dédoublement". Si le dédoublement des centres choisis a engendré une meilleure partition que la précédente, alors elle devient la nouvelle partition de référence.

#### Nuées dynamiques

Les nuées dynamiques sont une variante des K-means où une classe n'est plus représentée par un centroïde mais par un ensemble d'individus (Diday, 1971). Le but est de mieux représenter la classe que par son seul centre, qui peut être extérieur à la population.

Cet ensemble, appelé noyau, est choisi au hasard à l'intérieur de chaque groupe. La distance entre un point et le centre de la classe que l'on calculait dans les K-means est remplacée par une distance moyenne avec les points qui composent le noyau.

### 2.2.3 Évaluation de la qualité d'un partitionnement

Après avoir construit une partition ou un ensemble de partitions à l'aide d'une méthode, il convient d'évaluer la qualité de ce partitionnement. Pour ce faire, on peut faire appel à des critères internes ou externes. Les premiers permettent d'évaluer, surtout relativement à d'autres, la qualité de la partition proposée. Les scores maximaux permis par ces mesures ne sont généralement pas atteignables dans des réseaux réels. Les seconds permettent de comparer le résultat obtenu avec un résultat attendu, dans notre cas une partition faisant office de "vérité terrain".

Les critères internes peuvent eux-mêmes être subdivisés en critères dont l'usage est spécifique à une distance particulière, ou à une méthode de classification spécifique. C'est le cas de l'inertie interclasses, présentée dans la section 2.2.3.1. Les autres critères, non spécifiques à une topologie des données, peuvent être utilisés pour une distance quelconque définie entre les objets a été définie. C'est le cas des indices de Dunn, de Davies et Bouldin, de Silhouette.

#### 2.2.3.1 Critères d'évaluation internes

##### Les mesures d'inertie

Les notions d'inertie totale, intraclasses et interclasses étant largement utilisées ensuite, nous les rappelons brièvement.

Étant donné $V = \{v_1, \ldots, v_n\}$ un ensemble fini d'éléments de $\mathbb{R}^{|T|}$ formant une partition en $r$ classes $\mathscr{P} = \{C_1, \ldots, C_r\}$ de centres de gravité ou moyennes respectifs

$g_1, \ldots, g_r$. On note $g$ le centre de gravité ou moyenne de $V$.

L'*inertie interclasses* $I_{inter}$ d'une partition $\mathscr{P}$ permet de caractériser la séparation entre les classes de $\mathscr{P}$.

Elle est définie par :

$$I_{inter}(\mathscr{P}) = \sum_{k=1}^{r} m_k \left\| g_k - g \right\|^2 \tag{2.9}$$

où $g_k$ est le centre de gravité de la classe $C_k$ et $m_k$ est la masse qui lui est associée.

Une inertie interclasses forte traduit des classes aussi distinctes que possible. Une inertie interclasses faible, 0 au minimum, correspond à des classes ayant la même moyenne, égale à la moyenne de la population.

Ainsi, pour être une bonne classification, une partition doit avoir une inertie interclasses forte.

L'*inertie intraclasses* $I_{intra}$ d'une partition $\mathscr{P}$ est la moyenne pondérée des inerties internes à chaque classe. Elle permet de mesurer la variabilité des valeurs à l'intérieur des classes. Alors on a :

$$I_{intra}(\mathscr{P}) = \sum_{k=1}^{r} m_k \cdot I(C_k) \tag{2.10}$$

où $I(C_k)$ est l'inertie de la classe définie par :

$$\sum_{v \in C_k} m_v \cdot d^2(v, g_k) \tag{2.11}$$

De même l'*inertie totale* associée à $V$ est définie par :

$$I(V) = \sum_{v \in V} m_v \cdot d^2(v, g) \tag{2.12}$$

D'après le théorème de Konig Huygens l'inertie totale de $V$ est égale à la somme de l'inertie intraclasses et de l'inertie interclasses :

$$I(V) = I_{intra}(\mathscr{P}) + I_{inter}(\mathscr{P}) \tag{2.13}$$

Il est donc équivalent de maximiser l'inertie interclasses ou de minimiser l'inertie intraclasses.

L'inertie interclasses (voir section 2.2.3.1) peut donner une indication sur la bonne séparation des classes. Cependant, pour des nombres de classes différents, son utilisation ne permettra pas de comparer deux partitions. En particulier, la partition grossière obtient toujours la plus faible inertie interclasses, égale à zéro.

### Indice de Silhouette

L'indice de Silhouette sert à caractériser la séparation des classes ainsi que le fait que celles-ci soient compactes (Rousseeuw, 1987; Elghazel, 2007).

Pour tout élément $v$ de l'ensemble $V$, Rousseeuw définit l'indice de Silhouette par :

$$silhouette(v) = \frac{b(v) - a(v)}{max(a(v), b(v))} \tag{2.14}$$

où $a(v)$ est la dissimilarité moyenne entre l'individu $v \in V$ et tous les autres individus de la classe $c(v)$ à laquelle il appartient, et $b(v)$ est le minimum des distances de $v$ aux éléments relevant d'une autre classe que la sienne.

L'indice de Silhouette d'une classe $C_k$ est égal à la moyenne des indices de silhouette des individus qui la composent :

$$\forall C_k \in \mathscr{P}, silhouette(C_k) = \frac{\sum_{v \in C_k} silhouette(v)}{|C_k|} \tag{2.15}$$

De la même façon, l'indice de silhouette de la partition $\mathscr{P}$ est la moyenne des indices de silhouette des classes qui la composent :

$$silhouette(\mathscr{P}) = \frac{\Sigma_{C_k \in \mathscr{P}} silhouette(C_k)}{|\mathscr{P}|} \tag{2.16}$$

$silhouette(v)$ varie entre -1 et 1. Une valeur proche de 1 signifie que l'élément est bien classé, 0 signifie que l'élément se situe entre deux classes et -1 que l'élément est mal classé.

Pour l'indice de Silhouette, les éléments d'une bonne classe doivent avoir entre eux une distance moyenne faible tandis que la distance moyenne doit être plus grande entre eux et les éléments d'autres classes.

C'est un indice simple qui rend bien compte de la séparation et de la cohésion des classes. En adaptant les mesures de dissimilarité, l'indice de Silhouette est applicable à un large éventail de méthodes de classification.

Cependant, le calcul de l'indice de Silhouette a une complexité importante. Almeida *et al.* soulignent aussi le fait que des problèmes apparaissent lorsque l'on manipule des partitions où existent des classes singletons (Almeida et al., 2011).

### Indice de Dunn

L'indice de Dunn s'attache à la caractérisation du fait que des classes soient compactes et bien séparées (Dunn, 1973). C'est le rapport entre la plus petite distance interclasses $d(C_k, C_l)$ et la plus grande distance intraclasses $d'(C_k)$.

$$Dunn(\mathscr{P}) = \frac{min_{k=1,..,r;l=1,..,r;k\neq l} d(C_k, C_l)}{max_{k=1,..,r} d'(C_k)} \tag{2.17}$$

Plus la valeur de l'indice est élevée, meilleure est la classification. C'est un indice qui n'utilise pas une mesure prédéfinie. Il peut donc être utilisé avec différents algorithmes. Un inconvénient inhérent à l'indice de Dunn est qu'il dépend d'un nombre très limité d'éléments. Pour cette raison, il est dit non robuste. Toujours pour cette raison, il peut s'avérer difficile de comparer des partitions avec cet indice.

**Indice de Davies et Bouldin**

L'indice de Davies et Bouldin est proche de l'indice de Dunn mais ce sont les paires de classes qui sont considérées (Davies et Bouldin, 1979). Il est défini par :

$$DB(\mathscr{P}) = \frac{1}{|\mathscr{P}|} \sum_{C_k \in \mathscr{P}} max_{C_l \in \mathscr{P}, C_l \neq C_k} \left( \frac{d'(C_k) + d'(C_l)}{d(C_k, C_l)} \right) \tag{2.18}$$

où $d(C_k, C_l)$ est une distance interclasses, par exemple la distance entre les centres de gravité, et $d'(C_k)$ est une distance intraclasse, par exemple le diamètre intraclasse.

#### 2.2.3.2 Critères d'évaluation externe, par rapport à une vérité terrain

Bien que l'on considère un cadre non supervisé, on peut parfois disposer d'une partition de référence servant de vérité terrain. Cette partition peut être produite par des experts ou par un générateur. L'évaluation du résultat d'un algorithme de classification est effectuée vis-à-vis de cette partition faisant office de vérité terrain.

Nous notons $cat_v$ la catégorie du sommet $v$ dans la partition de la vérité terrain.

Nous verrons d'abord les solutions basées sur un appariement entre les catégories réelles et les classes produites. Les indices que nous présenterons ensuite, plus évolués, reposent sur trois approches principales. La première d'entre elles est l'approche combinatoire initiée par Rand, tirant parti des informations dont on dispose sur les paires d'individus. La deuxième est une approche probabiliste axée sur l'information mutuelle et la troisième utilise la théorie de l'information, dans laquelle la notion d'entropie a une place importante. Les problématiques liées à l'évaluation touchant toutes les sciences, les travaux mentionnés ici font l'objet de publication dans des revues et conférence aux thématiques très larges.

Tous les indices décrits ici donnent un score identique en cas de permutation des classes à l'intérieur des partitions, car les partitions sont des ensembles non ordonnés de classes. Hormis l'homogénéité et la complétude, qui sont des sous-indices pour la V-mesure, ils sont également symétriques : on peut inverser l'ordre de leurs arguments.

### Taux d'éléments bien classés

Le taux d'éléments bien classés $TBC(\mathscr{P})$ est le rapport entre le nombre d'éléments bien classés et le nombre total d'éléments classés.

$$TBC(\mathscr{P}) = \frac{card(\{v \in V | cat_v = c_v\})}{|V|} \tag{2.19}$$

Ce taux peut être mesuré à partir de la matrice de coïncidence ou matrice de confusion $\mathcal{M}_c$ dont chaque ligne correspond à une catégorie, chaque colonne à une classe. Le terme $\mathcal{M}_c(i,j)$ contiendra le nombre d'éléments de la catégorie $i$ qui ont été affectés à la classe $j$. Chaque catégorie de la vérité terrain est associée à une classe produite par la classification. La matrice n'est pas forcément carrée, l'algorithme évalué pouvant produire un nombre de classes différent de celui de la vérité terrain.

Cette mesure a l'avantage de s'appuyer sur une réalité concrète.

Mais cette façon d'évaluer la qualité d'une classification a des inconvénients. D'abord, cette approche soulève le "problème d'appariement" (*matching problem*), qui consiste à faire correspondre des catégories réelles à des classes produites. Les éléments biens classés dans des classes mal appariées sont systématiquement négligés dans ce critère.

De plus, si les effectifs dans les classes sont déséquilibrés, on peut obtenir de fort taux de bien classés, uniquement en classant selon la partition grossière. Par exemple, dans le cadre d'un partitionnement en deux classes, si on a 5 éléments dans la première catégorie réelle et 95 dans la seconde catégorie, alors la comparaison avec une classe produite contenant la totalité de l'effectif produira un score de 95%.

### Pureté

Une autre mesure d'évaluation de la qualité d'un partitionnement est la pureté.

Pour la calculer, on cherche la catégorie réelle majoritaire $C_k$ qui appartient à $\mathscr{P}_1$ dans chacune des classes produites $C'_l$ de la partition $\mathscr{P}_2$ produite par l'algorithme. La pureté est définie par :

$$Pureté_{simple}(\mathscr{P}_1, \mathscr{P}_2) = \frac{1}{N} \sum_{C'_l \in \mathscr{P}_2} max_{C_k \in \mathscr{P}_1} |C_k \cap C'_l| \tag{2.20}$$

Une autre façon de mesurer la pureté est définie par la probabilité, étant donnée une catégorie de $\mathscr{P}_1$, que deux objets tirés au hasard sans remise soient de la même classe de $\mathscr{P}_2$ (Solomonoff et al., 1998; Forestier et al., 2010). Cette définition a l'avantage de prendre en compte le fait qu'un couple d'objets, mal classés car non majoritaires dans leur classe, aient néanmoins été classés ensemble.

La pureté est un indice très intuitif et facile à calculer à partir de la matrice de

coïncidence. Elle prend une valeur maximale égale à 1 lorsque toutes les classes de la partition sont pures. C'est une mesure dont l'usage doit être restreint à la comparaison de partitions ayant le même nombre de classes.

Les deux mesures précédentes sont plus faciles à utiliser dans un contexte supervisé que dans un contexte non supervisé.

### Indice de Rand

Une autre mesure de concordance entre deux partitions est l'indice de Rand. Celui-ci est simplement le taux de paires d'éléments qui sont en accord, c'est-à-dire associés à la fois dans $\mathscr{P}_1$ et $\mathscr{P}_2$ ou séparés à la fois dans $\mathscr{P}_1$ et $\mathscr{P}_2$.

Ainsi, on définit les quatre configurations possibles pour les paires :

- A : nombre de paires d'éléments classés ensemble à la fois dans $\mathscr{P}_1$ et $\mathscr{P}_2$
- B : nombre de paires d'éléments présents dans des classes différentes dans $\mathscr{P}_1$ et dans $\mathscr{P}_2$
- C : nombre de paires d'éléments présents dans des classes différentes dans $\mathscr{P}_1$ mais présents dans une même classe dans $\mathscr{P}_2$.
- D : nombre de paires d'éléments présents dans une même classe dans $\mathscr{P}_1$ mais présents dans des classes différentes dans $\mathscr{P}_2$.

L'indice de Rand associé à deux partitions $\mathscr{P}_1$ et $\mathscr{P}_2$ est défini par :

$$Rand(\mathscr{P}_1, \mathscr{P}_2) = \frac{A + B}{A + B + C + D} \tag{2.21}$$

Il varie entre 0 et 1 et prend la valeur maximale en cas de concordance des partitions.

C'est un indice qui ne demande pas d'établir un appariement entre les classes réelles et les classes prédites, car seule est prise en compte la classification commune ou différente des paires d'éléments dans les deux partitions.

Cependant, l'indice est basé sur les paires d'objets, et non sur les objets eux-mêmes. De plus, une simple inversion de classification pour deux éléments dans deux classes de petites tailles sera bien moins pénalisée que si elle se produit dans des classes de grandes tailles par exemple.

### Indice de Rand ajusté

Dans la mesure où l'indice de Rand varie de façon très importante sur deux partitions tirées au hasard, ce que Vinh appelle la "constant baseline property", une version "ajustée" a été créée dont l'espérance est nulle lorsque les partitions sont sélectionnées au hasard (Hubert et Arabie, 1985). L'indice de Rand ajusté (*Adjusted Rand Index*) ou

ARI est une adaptation de l'indice de Rand conçue pour être insensible au nombre de classes. Il a pour formule générale :

$$\text{Indice ajusté} = \frac{\text{Indice} - \text{Indice attendu}}{\text{Indice maximum} - \text{Indice attendu}} \tag{2.22}$$

La conception de la mesure suppose l'usage de la distribution hypergéométrique, qui fixe les chances de sortie de chacune des partitions avec un nombre de classes fixé. L'ARI est définie par :

$$ARI(\mathscr{P}_1, \mathscr{P}_2) = \frac{RI(\mathscr{P}_1, \mathscr{P}_2) - E(RI(\mathscr{P}_1, \mathscr{P}_2)||\mathscr{P}_1|, |\mathscr{P}_2|)}{RI(\mathscr{P}_1, \mathscr{P}_2) - E(RI(\mathscr{P}_1, \mathscr{P}_2)||\mathscr{P}_1|, |\mathscr{P}_2|)} \tag{2.23}$$

$$= \frac{\sum_{C_k, C_l \in \mathscr{P}_1 \times \mathscr{P}_2} \binom{|C_k \cap C_l|}{2} - [\sum_{C_k \in \mathscr{P}_1} \binom{|C_k|}{2} \sum_{C_l \in \mathscr{P}_2} \binom{|C_l|}{2}]/\binom{N}{2}}{\frac{1}{2}[\sum_{C_k \in \mathscr{P}_1} \binom{|C_k|}{2} + \sum_{C_l \in \mathscr{P}_2} \binom{|C_l|}{2}] - [\sum_{C_k \in \mathscr{P}_1} \binom{|C_k|}{2} \sum_{C_l \in \mathscr{P}_2} \binom{|C_l|}{2}]/\binom{N}{2}} \tag{2.24}$$

où $|C|$ est la cardinalité de $C$ et $\binom{y}{x}$ est le nombre de façons de choisir $x$ éléments parmi $y$.

Si les partitions sont identiques, le score vaut 1. L'ARI est une mesure corrigée pour la chance : l'espérance de l'ARI de deux partitions tirées aléatoirement vaut 0. Cependant certaines paires de partitions ont une valeur d'ARI négative. La correction proposée ne corrige pas les problèmes de distribution présents dans l'intervalle $[0, 1]$ (Meilă, 2007).

### Entropie

Plusieurs mesures importantes de l'état de l'art, reposant sur des fondements probabilistes, sont définies par rapport à l'entropie de Shannon (Shannon et Weaver, 1949). Celle-ci est définie pour une variable $X$ prenant $n$ valeurs $\{x_1, \ldots, x_n\}$ avec des probabilités respectives $\{p(x_1), \ldots, p(x_n)\}$ par :

$$H(X) = \sum_{i=1}^{n} p(x_i) log\left(\frac{1}{p_{x_i}}\right) \tag{2.25}$$

Pour deux variables $X$ et $Y$, où $p(x_i y_j)$ désigne la probabilité conjointe que $X$ prenne la valeur $x_i$ et $Y$ la valeur $y_j$, l'entropie conjointe est définie par :

$$H(X, Y) = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} p(x_i, y_j) log\left(\frac{1}{p(x_i, y_j)}\right) \tag{2.26}$$

C'est à partir de ces deux mesures qu'est construite l'*information mutuelle*.

Elle mesure la dépendance de deux variables aléatoires $X$ et $Y$ par :

$$MI(X,Y) = H(X) + H(Y) - H(X,Y) \tag{2.27}$$

$MI(X,Y)$ est d'autant plus élevée que $X$ et $Y$ sont liées et elle est nulle si $X$ et $Y$ sont indépendantes.

Définissons $P(C_k) = \frac{|C_k|}{N}$ la probabilité qu'un objet choisi au hasard appartienne à la classe $C_k$. Alors l'entropie associée à $\mathscr{P}$ est égale à :

$$H(\mathscr{P}) = -\sum_{C_k \in \mathscr{P}} P(C_k) log(P(C_k)) = -\sum_{C_k \in \mathscr{P}} \frac{|C_k|}{N} log\left(\frac{|C_k|}{N}\right) \tag{2.28}$$

L'entropie conjointe de deux partitions $\mathscr{P}_1$ et $\mathscr{P}_2$ est donc :

$$H(\mathscr{P}_1, \mathscr{P}_2) = -\sum_{C_k \in \mathscr{P}_1} \sum_{C'_l \in \mathscr{P}_2} P(C_k, C'_l) log\left(P(C_k, C'_l)\right) \tag{2.29}$$

$$= -\sum_{C_k \in \mathscr{P}_1} \sum_{C'_l \in \mathscr{P}_2} \frac{\left|\left\{v \in V | v \in C_k, v \in C'_l\right\}\right|}{N} log\left(\frac{\left|\left\{v \in V | v \in C_k, v \in C'_l\right\}\right|}{N}\right) \tag{2.30}$$

**Information mutuelle normalisée (NMI) et information mutuelle**

L'information mutuelle normalisée (*Normalized Mutual Information*, NMI), est une mesure dérivée de la notion d'information mutuelle (MI) (Strehl et Ghosh, 2003).

Soit l'information mutuelle définie par :

$$MI(\mathscr{P}_1, \mathscr{P}_2) = \sum_{C_k \in \mathscr{P}_1} \sum_{C'_l \in \mathscr{P}_2} \frac{|\{v \in V | v \in C_k, v \in C_l\}|}{N} log\left(\frac{\left|\left\{v \in V | v \in C_k, v \in C'_l\right\}\right| \cdot N}{|C_k| \cdot |C'_l|}\right) \tag{2.31}$$

alors la NMI est :

$$NMI(\mathscr{P}_1, \mathscr{P}_2) = \frac{MI(\mathscr{P}_1, \mathscr{P}_2)}{\sqrt{H(\mathscr{P}_1) H(\mathscr{P}_2)}} \tag{2.32}$$

où $MI$ est l'information mutuelle et $H$ est la mesure d'entropie.

Vinh *et al*. listent des variations de cette formulation où la racine carrée est remplacée par le minimum, le maximum, etc. (Vinh et al., 2010). La NMI montre cependant un biais lorsque l'on compare des partitions avec des nombres de classes différents (Vinh et al., 2009).

### Information mutuelle ajustée (AMI)

L'information mutuelle ajustée (Adjusted Mutual Information) est une mesure basée sur la notion d'information mutuelle (Vinh et al., 2010).

L'AMI est la version corrigée par Vinh *et al.* de la NMI, selon la formule de la correction pour la chance proposée par Hubert *et al.* (voir formule 2.22) (Hubert et Arabie, 1985).

$$AMI(\mathscr{P}_1, \mathscr{P}_2) = \frac{MI(\mathscr{P}_1, \mathscr{P}_2) - E\{MI(\mathscr{P}_1, \mathscr{P}_2)\}}{\max\{H(\mathscr{P}_1), H(\mathscr{P}_2)\} - E\{MI(\mathscr{P}_1, \mathscr{P}_2)\}} \tag{2.33}$$

où $MI$ est l'information mutuelle, $H$ est la mesure d'entropie et $E$ est l'espérance mathématique.

Deux partitions aléatoires selon la distribution hypergéométrique obtiennent alors des scores identiques, mais cette correction peut faire perdre aux mesures certaines propriétés.

### V-mesure

La V-mesure est une mesure basée sur la notion d'entropie et les travaux de la théorie de l'information (Rosenberg et Hirschberg, 2007). Elle est calculée à partir de deux sous-indices, l'homogénéité et la complétude. Quelles que soient deux partitions $\mathscr{P}_1$ et $\mathscr{P}_2$, l'homogénéité et la complétude sont liées par la relation :

$$homogénéité(\mathscr{P}_1, \mathscr{P}_2) = complétude(\mathscr{P}_2, \mathscr{P}_1) \tag{2.34}$$

Par la suite, on considérera $\mathscr{P}_1$ comme la partition réelle et $\mathscr{P}_2$ comme la partition produite par la méthode de classification.

### Homogénéité

Pour satisfaire le critère d'homogénéité, une classification doit assigner **seulement** les éléments appartenant à une même catégorie à une même classe. Une communauté est dite homogène, avec un score d'homogénéité égal à 1, si elle ne contient que des membres issus d'une même classe. On notera que la partition discrète est totalement homogène.

L'homogénéité est calculée comme suit :

$$homogénéité(\mathscr{P}_1|\mathscr{P}_2) = \begin{cases} 1 & \text{si } H(\mathscr{P}_1|\mathscr{P}_2) = 0 \\ 1 - \frac{H(\mathscr{P}_1|\mathscr{P}_2)}{H(\mathscr{P}_1)} & \text{sinon} \end{cases} \tag{2.35}$$

où $H(\mathscr{P}_1)$ est l'entropie de $\mathscr{P}_1$ et $H(\mathscr{P}_1|\mathscr{P}_2)$ est l'entropie conditionnelle de $\mathscr{P}_1$ sachant $\mathscr{P}_2$ définie par :

$$H(\mathscr{P}_1|\mathscr{P}_2) = -\sum_{C_l \in \mathscr{P}_2} \sum_{C_k \in \mathscr{P}_1} \frac{|C_k \cap C_l|}{N} log\left(\frac{|C_k \cap C_l|}{\sum_{C_k \in \mathscr{P}_1} |C_k \cap C_l|}\right) \tag{2.36}$$

**Complétude**

Pour satisfaire la notion de complétude, une classification doit affecter **tous** les éléments appartenant à une même catégorie réelle dans une même classe prédite. On notera que la partition grossière est totalement "complète". La complétude étant symétrique à l'homogénéité, sa définition est analogue à celle-ci.

La V-mesure est définie comme la moyenne harmonique des indices précédents. Dans nos expérimentations nous utilisons la formule suivante :

$$VM(\mathscr{P}_1, \mathscr{P}_2) = 2 \cdot \frac{homogénéité(\mathscr{P}_1, \mathscr{P}_2) \cdot complétude(\mathscr{P}_1, \mathscr{P}_2)}{homogénéité(\mathscr{P}_1, \mathscr{P}_2) + complétude(\mathscr{P}_1, \mathscr{P}_2)} \tag{2.37}$$

Rosenberg et Hirschberg exposent l'intérêt de la V-mesure sur l'exemple de la figure 2.1. La comparaison est opérée face à la F-mesure, qui est une fonction de la précision (taux d'éléments de la catégorie dans la classe) et du rappel (taux d'éléments de la catégorie correctement identifiés) (Van Rijsbergen, 1979).

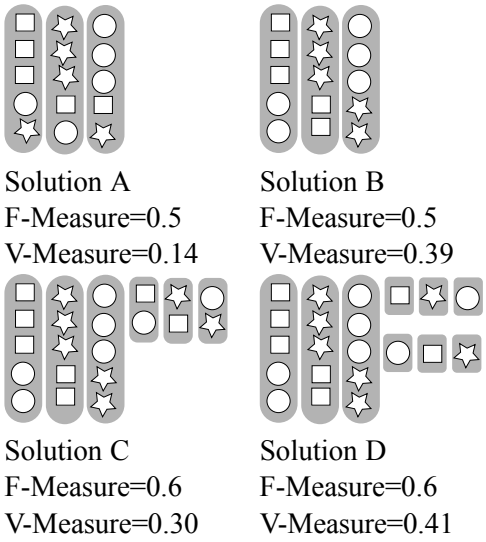



FIGURE 2.1 – Le problème d'appariement (par Rosenberg *et al.*)

Cet exemple souligne que, dans le cas où il vaut mieux éclater des classes non

homogènes, quitte à créer des classes composées d'éléments isolés, alors la V-mesure est plus pertinente que la F-mesure.

Selon Rabbany *et al.*, le choix des critères de validité dépend grandement des jeux de données et il doit être fait en fonction de l'application (Rabbany et al., 2012).

Les conclusions de Almeida *et al.*, étudiant la question à partir d'un réseau de collaboration scientifique et d'un réseau de partage de fichiers en pair à pair, vont dans le même sens (Almeida et al., 2011). Ils pointent le mauvais degré de précision apportée dans la caractérisation des communautés par ces indices qui paraissent biaisés.

L'évaluation interne est elle aussi très dépendante de la notion de bonne partition que l'on retient. L'évaluation selon un critère externe parait donner lieu à moins de débats, à condition de retenir des mesures corrigées envers les biais statistiques provoqués par les déséquilibres dans les effectifs des classes (corrections envers la "chance").

## 2.3 Détection de communautés dans les graphes

L'objectif de la détection de communautés dans les graphes, ou encore dans les réseaux sociaux, est de créer une partition des sommets, en tenant compte des relations qui existent entre les sommets dans le graphe, de telle sorte que les communautés soient composées de sommets fortement connectés et qu'elles soient peu reliées entre elles (Brandes et al., 2003; Newman, 2004; Schaeffer, 2007; Lancichinetti et Fortunato, 2009). Parmi les principales méthodes de détection de communautés proposées dans la littérature, on peut citer celles qui optimisent une fonction de qualité pour évaluer la qualité d'une partition donnée, comme la modularité, la coupe ratio, la coupe min-max ou la coupe normalisée (Kernighan et Lin, 1970; Chan et al., 1994; Shi et Malik, 2000; Ding et al., 2001; Newman et Girvan, 2004), les techniques hiérarchiques comme les algorithmes de division (Flake et al., 2003), les méthodes spectrales (Von Luxburg, 2007) ou l'algorithme de Markov et ses extensions (Satuluri et Parthasarathy, 2009). Ces techniques de partitionnement de graphes sont très utiles pour détecter des composantes fortement connectées dans un graphe.

Cette section est consacrée à la détection de telles communautés qui repose donc uniquement sur les arêtes et leur disposition dans et entre les classes. Deux sommets classés ensemble doivent être plus liés entre eux, directement ou par l'intermédiaire d'autres sommets, que vis-à-vis de sommets placés dans d'autres classes.

Nous présenterons les méthodes importantes du domaine en mettant l'accent sur celles qui optimisent un critère de qualité de la partition, notamment la modularité puisque nos propositions utilisent ce critère, en section 2.3.2. Nous évoquerons les possibilités d'évaluation des partitions obtenues en section 2.3.3 après avoir défini

plus formellement le problème.

### 2.3.1 Formalisation

Soit un graphe $G = (V, E)$. On cherche une partition $\mathscr{P}$ de $V$ telle que chaque classe $C$ de $\mathscr{P}$ renferme les deux extrémités d'un grand nombre d'arêtes, tandis que les arêtes ayant des extrémités dans deux classes différentes soient aussi rares que possible.

Dans la réalité, une partition prise au sens de la définition donnée à la section 2.2.1 correspond d'assez loin à la transposition mathématique du terme *communauté*. En effet, en général, les individus peuvent appartenir à différents groupes d'individus. On peut ainsi être chanteurs et danseurs, étudiant et actif, ligérien et rhodanien, etc. Un cadre souvent utilisé pour pallier à de tels cas de figure est celui des communautés recouvrantes. Récemment, de nombreux travaux se sont intéressés à la détection de ce type de communautés (Wang et Fleury, 2009), mais ce ne sera pas le cas dans cette thèse.

### 2.3.2 Approches méthodologiques

La classification des sommets d'un graphe est un domaine plus jeune que la classification non supervisée de données vectorielles.

Comme la nature des données considérées est différente, il n'y a pas d'équivalence d'un problème vers l'autre qui permettrait d'adapter toutes les méthodes efficaces sur les vecteurs vers les graphes et inversement.

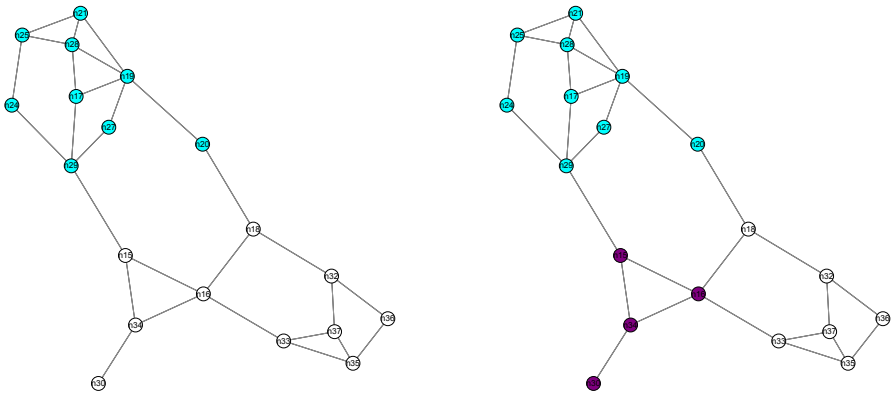
Fortunato propose un panorama large mais néanmoins détaillé des solutions qui y sont apportées (Fortunato, 2009). Porter *et al.* proposent eux aussi une étude détaillée (Porter et al., 2009).

Dans de nombreux travaux, la classification de sommets dans un graphe est considérée sous l'angle d'un problème d'optimisation. Une fonction associée à une partition des sommets du graphe procure ainsi un score qui mesure la qualité de la partition. L'un des premiers critères utilisés a été celui de la coupure minimum (Newman, 2004), dont des extensions ont été proposées ensuite, comme le ratio cut et le normalized cut (Chan et al., 1994; Shi et Malik, 2000). Selon ce critère, la meilleure partition en $k$ classes d'un graphe est celle qui est obtenue en retirant un nombre minimal d'arêtes dans le graphe d'origine de façon à obtenir $k$ composantes non connexes. Il ne sera pas question ici des méthodes génériques appliquées expressément aux graphes (méthodes hiérarchiques sur la distance du plus court chemin, etc.) mais plutôt de celles liées à nos propres travaux basés notamment sur la modularité.

Nous présentons maintenant le critère de modularité $Q_{NG}$, critère qui est au cœur de cette thèse tel qu'il a été défini par Newman et Girvan (Newman et Girvan, 2004).

#### 2.3.2.1 Modularité

Comment juger quantitativement si il est plus judicieux de partitionner un graphe d'une façon (voir figure 2.2a) ou d'une autre (voir figure 2.2b) ?



(a) Partition en 2 groupes (b) Partition en 3 groupes

FIGURE 2.2 – Pourquoi la partition (a) est-elle la plus *mauvaise* ?

Pour répondre à cette question, on peut utiliser la modularité $Q_{NG}$, proposée par Newman et Girvan (Newman et Girvan, 2004; Newman, 2006).

C'est une mesure qui a pour but de caractériser l'efficacité d'une partition des sommets d'un graphe au regard de la densité des liens à l'intérieur des groupes et du nombre de liens entre ces groupes, ainsi que vis-à-vis du degré des sommets pris en compte.

Elle compare la proportion des arêtes du réseau qui relient des sommets issus d'une même communauté à la proportion attendue dans un graphe équivalent dans le sens où les sommets ont la même distribution des degrés mais où les arêtes auraient été placées aléatoirement entre tous les sommets.

La modularité $Q_{NG}$ se calcule comme suit :

$$Q_{NG} = \frac{1}{2M} \sum_{v,v'} \left[ \left( A_{vv'} - \frac{k(v) \cdot k(v')}{2M} \right) \cdot \delta(c(v), c(v')) \right] \tag{2.38}$$

où le couple $(v, v')$ court sur toutes les paires de sommets de $V \times V$ (de façon à prendre également les boucles en compte). $M$ est la somme des valuations des arêtes, $\mathcal{A}$ est la matrice d'adjacence, $c(v)$ désigne la classe du sommet $v$ et $\delta$ est la fonction

de Kronecker qui vaut 1 si ses arguments sont égaux et 0 sinon.

La modularité prend sa valeur entre -1 (toutes les arêtes sont intercommunautaires) et 1 (toutes les arêtes sont intracommunautaires). La modularité vaut 1 dans un graphe partitionné de manière à ne pas avoir d'arête placée entre 2 sommets de communautés différentes (mais au moins une dans une communauté). La modularité vaut zéro pour la partition grossière. C'est aussi la valeur vers laquelle tend la modularité d'une partition aléatoire. D'après Newman *et al.*, les valeurs de modularité pour des réseaux ayant des structures de communauté s'étalent généralement de 0,3 à 0,7. Des valeurs supérieures sont rarement observées. Si le nombre d'arêtes intragroupes (reliant deux arêtes du même groupe) est inférieur au nombre d'arêtes attendues dans un graphe faisant l'objet d'une distribution aléatoire, alors la modularité $Q_{NG}$ est négative. Enfin la modularité est indéfinie sur un graphe sans aucun lien (graphe *vide*).

Les auteurs du concept de modularité ont envisagé d'employer différentes variations du graphe aléatoire, appelé modèle nul. Cependant, le graphe respectant la distribution des degrés s'est très vite imposé dans la littérature.

Une variante de la modularité pour les graphes orientés a été formalisée par Nicosia *et al.* (Nicosia et al., 2009). Celle-ci repose sur l'utilisation d'une matrice d'adjacence non symétrique et la prise en compte du sens des connexions dans le calcul du produit des degrés des sommets.

Si elle a eu un impact très important sur l'étude de la détection de communautés dans les graphes, il a néanmoins été montré que la modularité a des défauts. Ainsi elle présente une *limite de résolution*, qui restreint la possibilité de disposer de petites communautés qui soient bien définies (Fortunato et Barthélemy, 2007). Des variantes ont été créées pour s'affranchir de cette limite de résolution, notamment en utilisant un paramètre permettant de choisir si on désire obtenir de petites ou de grandes classes. Lancichinetti et Fortunato montrent que même en utilisant une extension de la modularité présentant un facteur de variation de la taille des communautés à produire, certaines configurations ne sont pas bien résolues (Lancichinetti et Fortunato, 2011).

Gautier *et al.* proposent de palier à ce problème avec un facteur qui contraindrait le nombre de classes à produire (Krings et Blondel, 2011).

Montgolfier *et al.* démontrent que des graphes particuliers (tores, hypercubes) qui n'ont *a priori* pas de structure de communauté ont néanmoins des partitions à modularité forte (Montgolfier et al., 2012). Keller et al. ont calculé la modularité maximale de la partition d'une grille (torique) et celle-ci atteint 0,953 dans une grille de 256 sommets de côté, ce qui est très élevé (Keller et Viennet, 2012). De plus, celle-ci tend vers 1 quand la taille de la grille augmente.

Yang *et al.*, mettent en évidence que des critères plus simples que la modularité, la conductance (qui sera présentée dans la section 2.3.3.1) et le taux d'enrôlement dans

des triades, notamment, caractérisent souvent mieux un caractère de communauté que la modularité (Yang et Leskovec, 2012). Cependant, la méthodologie utilisée faisant appel à des perturbations de partition plutôt qu'à une optimisation, les indices proposés posent en fait souvent des problèmes pour leur utilisation dans un cadre d'optimisation. Par exemple, le nombre d'arêtes internes à des communautés, qui est étudié, trouve son maximum pour la partition grossière.

Dans la méthode de Louvain, basée sur la modularité, Blondel *et al.* reconnaissent le problème posé par l'exemple du "Collier de perles" (Aynaud et al., 2010; Fortunato et Barthélemy, 2007). Une boucle de cliques reliées entre elles par des ponts formés d'une arête, au lieu d'être identifiées comme des communautés distinctes, finissent par se voir agglomérées entre elles. Ainsi il apparaît que les communautés sont regroupées si leur taille est inférieure à $\sqrt{2M}$.

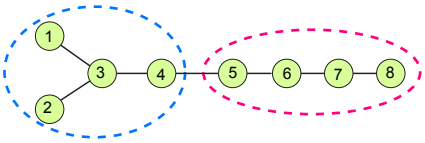



FIGURE 2.3 – Défaut de la modularité souligné par Ye *et al.*

Ye *et al.* soulèvent un défaut de l'utilisation du rapport entre le nombre d'arêtes internes et d'arêtes externes aux communautés (Ye et al., 2012). Ainsi, dans la figure 2.3, les deux communautés suggérées ont la même contribution à la modularité (elles ont le même nombre d'arêtes internes), bien que la communauté située à gauche soit manifestement meilleure.

Des travaux ont porté sur la paramétrisation de la résolution. Ainsi Arenas *et al.* proposent d'adjoindre une boucle sur tous les sommets, et de régler sa valuation, de façon à créer une résistance qui va empêcher dans une certaine mesure les petites communautés de se regrouper (Arenas et al., 2008). Reichardt *et al.* proposent eux aussi une paramétrisation au sein du critère de modularité, mais qui porte elle sur le calcul du graphe aléatoire (Reichardt et Bornholdt, 2006).

#### 2.3.2.2 Approches heuristiques visant à optimiser un critère

La recherche d'une partition optimisant un critère particulier est le plus souvent NP complète car le nombre de partitions possibles d'un ensemble explose avec le nombre de ses éléments. Ainsi, les heuristiques, qui consistent en une exploration contrainte, mais, on l'espère, plus intelligente pour ce problème particulier, ont une grande importance dans le partitionnement de graphes. L'exploration tient ainsi compte du voisinage local des sommets lorsqu'il cherche à produire une nouvelle partition qui pourrait

mieux satisfaire le critère à maximiser.

D'après Noack *et al.* (Noack et Rotta, 2009), ces méthodes peuvent être classées en deux familles.

La première famille rassemble des algorithmes de dégrossissement, strictement agglomératifs (*Coarsening algorithms*). Dans ce cas de figure, on opère uniquement par fusions. On recherche à chaque fois une fusion sûre, car elle est définitive.

- Pas à pas. (*Single-Step*). Un sommet à la fois change de classe.
- Multi-pas (*Multi-Step*). Plusieurs sommets changent de classe au même moment (Schuetz et Caflisch, 2007).
- Priorisation de fusions (*Merge Prioritizers*).

La seconde famille contient les algorithmes à raffinement. Cette fois-ci, il est possible de revenir après-coup sur une fusion.

- *Complete Greedy* est une solution qui cherche à chaque itération la meilleure modification d'affectation. Cette modification d'affectation concerne tous les sommets et toutes les classes d'affectation.
- *Fast Greedy* cherche successivement pour tous les sommets à les affecter dans la meilleure classe possible (Schuetz et Caflisch, 2007; Ye et al., 2008).
- L'adaptation de Kernighan-Lin (pour la coupure minimale) adaptée par Newman pour la modularité (Newman, 2006) vise, pour augmenter la variabilité des partitions explorées, à ne changer l'affectation de chaque sommet qu'une seule fois, mais il n'est plus nécessaire que le changement d'affectation provoque un gain par rapport à l'affectation initiale.
- Le raffinement multi-niveaux. Avec les approches précédentes, la fusion de deux sous-communautés denses moyennement connectées entre elles n'est pas possible sans effectuer plusieurs changements d'affectation qui auraient des impacts négatifs sur la modularité. Les méthodes utilisant le raffinement multi-niveaux construisent des résultats intermédiaires constitués par des graphes dont les sommets constituent des agglomérations des sommets des graphes des niveaux inférieurs. Lorsque la convergence d'un niveau est atteinte, on crée alors un nouveau graphe à partir des communautés courantes. Le processus peut ensuite être poursuivi sur ce nouveau graphe (Djidjev, 2008; Ye et al., 2008; Blondel et al., 2008).

Nous allons maintenant aborder des méthodes basées sur l'optimisation de la modularité. On s'attend à ce qu'elles renvoient un résultat identique. Cependant, celles-ci étant basées sur des heuristiques, leur efficacité en temps et en mémoire ainsi que leur faculté à renvoyer une solution proche de l'optimale est propre à chaque méthode.

#### 2.3.2.3 FastQ

Clauset, Newman et Moore introduisent en 2004 la méthode FastQ, pour *Fast Modularity* (Clauset et al., 2004). Étant entendu que tester la modularité de toutes les partitions possibles est hors d'atteinte, FastQ utilise la modularité pour guider un processus de recherche glouton (Brandes, 2008). Ce processus est un processus agglomératif donc ascendant, décrit dans l'algorithme 2. Tous les sommets sont initialement placés dans des classes dont ils sont les uniques représentants. On cherche alors les fusions successives qui produisent les meilleurs gains de modularité.

Il est fait utilisation d'une matrice dans laquelle on calcule quel est le gain de modularité apporté par les différentes fusions possibles. Un vecteur $a_i$ contient initialement les degrés de tous les sommets et un tas $h$ sert à faire ressortir l'incrément maximum que l'on peut obtenir à n'importe quel moment.

**Algorithme 2** : Algorithme FastQ

**Entrées** : Un graphe
**Sorties** : La hiérarchie de fusions
1 Calcul des valeurs initiales de $\Delta Q_{ij}$ et $a_i$, et remplir le tas-maximum avec le plus grand élément de chaque ligne de la matrice $\Delta Q$;
2 **répéter**
3 | Sélectionner le plus grand $\Delta Q_{ij}$ de $h$, unir les communautés correspondantes, mettre à jour la matrice $\Delta Q$, le tas $H$ et $a_i$ et incrémenter $Q$ de $\Delta Q_{ij}$ ;
4 **jusqu'à** *il ne reste plus qu'une seule communauté* ;

Le temps avancé pour cette solution est de $O(m \cdot d \cdot log(N))$ où $d$ est la profondeur du dendogramme décrivant la structure de communauté du réseau.

D'après Blondel *et al.* cette méthode privilégie les grandes communautés (Blondel et al., 2008).

La complexité de FastQ est, selon ses auteurs, de $O(n^3 \cdot log(n))$ dans le pire des cas, mais de $O(n \cdot log(n))$ en pratique le plus souvent.

#### 2.3.2.4 Méthode de Wakita et Tsurumi

Wakita *et al.* proposent des variantes de l'algorithme de Clauset *et al.* qui passent mieux à l'échelle et permettent de traiter des millions de sommets en moins d'une heure (Wakita et Tsurumi, 2007). Pour y parvenir, ils font usage d'un tas qui sert à mémoriser les déplacements de sommets provoquant les plus grandes augmentations $\Delta Q_{NG}$ de modularité. De plus, trois heuristiques sont comparées. Celles-ci visent à effectuer des fusions équilibrées lors du déroulement de l'algorithme. Pour cela, l'usage

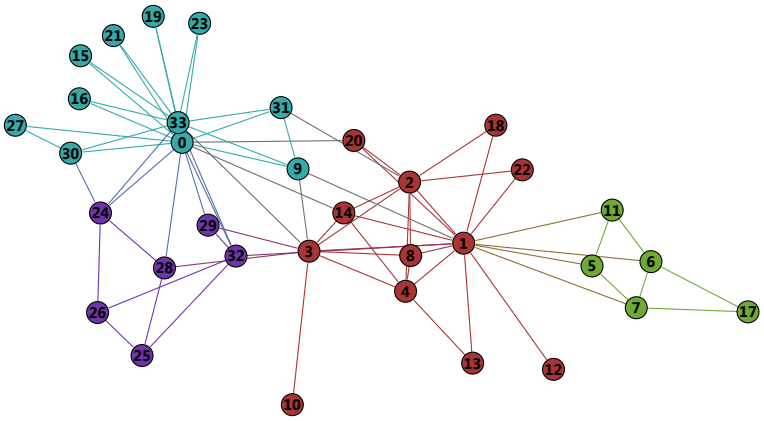



FIGURE 2.4 – Partition optimisant le score de modularité sur le réseau Karate

de différentes mesures définissant ce que doit être la taille de communautés sont proposées, parmi lesquelles le nombre de membres et le degré. L'auteur souligne que les tailles des communautés formées sont soit petites, soit grandes, sans aucune communauté de taille intermédiaire.

#### 2.3.2.5 Méthode de Louvain

La méthode de Louvain sera étudiée plus en détail dans la suite du présent document puisque notre contribution consiste à en proposer une extension adaptée aux réseaux d'information.

La méthode de Louvain est une méthode récente proposant la classification de sommets dans un graphe qui peut être valué (Blondel et al., 2008). Elle est appréciée pour sa vitesse, qui la rend utilisable sur des réseaux de très grande taille (Greene et al., 2010). La méthode de Louvain n'est pas supervisée ; le nombre de groupes à former n'est pas demandé avant l'exécution. La méthode retourne une partition en cherchant à optimiser le critère de modularité (voir section 2.3.2.1). Cependant, il est important de souligner que ce n'est pas un algorithme fournissant un maximum global, mais qu'il produit en général un optimum local.

Aynaud *et al.* décrivent ainsi le processus en deux étapes de la méthode présentée dans l'algorithme 3 (Aynaud et Guillaume, 2011). La méthode de Louvain est une méthode gloutonne conçue pour optimiser la modularité sur un graphe optionnellement valué. Elle consiste en deux phases qui sont exécutées en alternance. Initialement, chaque sommet constitue une communauté. Ensuite, durant la première phase, les

**Algorithme 3** : La méthode de Louvain

**Entrées** : Un graphe $G$
**Sorties** : Une partition $\mathscr{P}$

```
répéter
    Placer chaque sommet de G dans une unique classe;
    Sauver la modularité de cette décomposition;
    répéter
        pour tous les sommet u de G faire
            C ← communauté voisine maximisant le gain de modularité;
            si le déplacement de u vers C induit un gain strictement positif alors
                déplacer u de sa communauté vers C;
    jusqu'à ce qu'aucun sommet ne puisse plus être déplacé ;
    si le critère de qualité atteint est supérieur à sa valeur initiale alors
        fin ← faux;
        Afficher la décomposition trouvée;
        Fusionner G en le graphe entre les classes;
    sinon
        fin ← vrai ;
jusqu'à fin ;
```

sommets sont considérés un à un (le résultat final dépend de l'ordre dans lequel les sommets sont énumérés). Chacun d'eux est placé dans l'une des communautés voisines (la sienne étant incluse), choisie car elle maximise le gain de modularité. Ce processus est répété jusqu'à ce que plus aucun sommet ne puisse être déplacé (on parle alors d'optimum local). Pour éventuellement se dégager de celui-ci, la phase 2 consiste à construire un nouveau graphe entre les communautés trouvées à l'issue de la phase 1 : il y a un sommet dans le graphe pour chaque communauté et, pour deux communautés $C$ et $C$', il y a une arête de valuation $w$ où $w = \sum_{v,v' \in C \times C'} poids(v, v')$. Il y a aussi une boucle sur $C$ de poids $w = \sum_{v,v' \in C \times C} poids(v, v')^2$. L'algorithme exécute alors les phases 1 et 2 alternativement jusqu'à ce que la modularité ne s'améliore plus.

Le pseudo-code associé au processus précédent est proposé dans l'algorithme 3.

Les auteurs ont étudié l'influence de l'ordre de l'énumération des sommets lors du déroulement de l'algorithme (Aynaud et al., 2010). L'énumération des sommets dans l'ordre de leurs classes d'appartenance (tous les sommets de la première communauté puis tous les sommets de la seconde communauté, etc.) n'a amélioré ni les temps de calcul ni la modularité de la partition finale.

C'est une méthode très rapide, y compris sur des réseaux de plusieurs millions de

sommets. Elle ne nécessite aucun paramètre, elle optimise simplement le critère de modularité.

Les auteurs de la méthode de Louvain s'étant intéressés depuis à la classification de réseaux en mouvement, ceux-ci soulignent l'inconvénient de cette méthode qui réagit fortement à des modifications minimes du réseau, pouvant occasionner une modification non seulement locale, mais aussi globale de la partition.

Comme toutes les méthodes d'optimisation de la modularité, elle hérite des inconvénients de cette dernière (voir section 2.3.2.1).

#### 2.3.2.6 Travaux inspirés par la méthode de Louvain

La méthode de Louvain a inspiré de très nombreux travaux, que ce soit dans le but de corriger certains de ses inconvénients, notamment parfois ceux de la modularité elle-même, ou encore dans le but de l'adapter ou de l'étendre. C'est le cas des travaux de Collingsworth ou de ceux de Seifi (Collingsworth et Menezes, 2013; Seifi, 2012).

**Auto-organisation via l'utilisation de l'entropie de sommet**

L'entropie de sommet est une mesure qui a pour but de quantifier le fait qu'un sommet soit placé dans une communauté qui soit bien choisie vis-à-vis de la configuration de ses voisins. Le principe de l'auto-organisation donne à chaque sommet la possibilité de changer de communauté de façon à maximiser ce critère. La méthode de Louvain est utilisée parallèlement à ce processus pour optimiser la position.

Collingsworth *et al.* proposent une méthode basée sur l'entropie de sommet (Collingsworth et Menezes, 2013). Ici, on mesure l'entropie dans le cas où l'on change la communauté d'affectation d'un sommet :

$$H_S = -\sum_{i=1}^{n} \frac{p_i log_2(p_i)}{e^{\frac{k}{4}}} \tag{2.39}$$

où $n$ est le nombre de communautés que le sommet peut potentiellement rejoindre, $y$ est le nombre de triades (sous-graphes complet comportant 3 sommets) intracommunautaires formées en joignant une communauté et $p_i$ est le taux des arêtes voisines du sommet considéré qui le relient à la communauté numéro $i$. Une triade est un sous-graphe complet comportant 3 sommets.

**Cœurs de communautés**

Seifi propose une méthode de détection des cœurs de communautés. Ces cœurs sont des ensembles de sommets qui sont très souvent ou toujours classés ensemble

lors de l'application de méthodes de classification non déterministes (Seifi, 2012).

Cette approche est intéressante par le fait qu'elle s'attache à produire des communautés non complètes ; elle ne respecte pas la condition de complétude que l'on a posé pour une partition, mais les résultats produits font l'objet d'une confiance très supérieure aux autres méthodes.

### 2.3.3 Critères d'évaluation

#### 2.3.3.1 Critères d'évaluation interne

La détection de communautés dans les graphes ayant été très largement traitée sous l'angle d'un problème d'optimisation, l'éventail des critères d'évaluation interne recouvre celui des critères d'optimisation détaillés dans la section 2.3.

D'autres critères d'évaluation de structures de communauté peuvent cependant être mis en œuvre parmi lesquels on peut citer la couverture, la conductance, la performance ou le coefficient de clustering décrits ci-après.

**La couverture**

La couverture est la proportion de la somme des valuations des d'arêtes intracommunautaires par rapport aux valuations totales des arêtes de $V$, c'est-à-dire $M$ (Almeida et al., 2011) :

$$couverture(\mathscr{P}) = \frac{\psi(\mathscr{P})}{M} \tag{2.40}$$

où $\psi(\mathscr{P})$ désigne la somme des valuations des arêtes intracommunautaires, c'est-à-dire dont les deux extrémités appartiennent à la même classe :

$$\psi(\mathscr{P}) = \frac{1}{2}\sum_{k=1}^{r} \mathcal{A}_{v,v'}; v, v' \in C_k \tag{2.41}$$

La couverture prend sa valeur entre 0 et 1. Comme c'est une mesure qui trouve son maximum dans la partition grossière, elle ne pourra être utilisée que dans la comparaison de deux partitions pour lesquelles le nombre de classes est identique. C'est la raison pour laquelle il nous faut introduire maintenant des mesures qui prennent également en compte le nombre d'arêtes qui se trouvent sur la frontière entre deux communautés, les arêtes interclasses.

### La conductance

La conductance, également appelée métrique de la coupure normalisée, pour une classe, mesure le taux de valuations d'arêtes qui pointent à l'extérieur de la classe, à la classe ou à son complément, selon quel côté de la coupure la somme des valuations des arêtes est la moins importante (Leskovec et al., 2008). En effet, ce critère a d'abord été conçu pour évaluer la qualité d'une coupure, qui consiste en la scission des sommets d'un graphe en deux parties. La conductance d'une classe $C$ est donc définie par :

$$conductance(C) = \frac{\sum_{u \in C} \sum_{v \notin C} \mathcal{A}_{u,v}}{min\left(\sum_{u \in C} \sum_{v \in V} \mathcal{A}_{u,v}, \sum_{u \notin C} \sum_{v \in V} \mathcal{A}_{u,v}\right)} \tag{2.42}$$

À partir de cette mesure, on définit la conductance du graphe comme la plus petite des conductances de chacune des classes :

$$conductance(G) = min_{C_k \in \mathscr{P}}(conductance(C_k)) \tag{2.43}$$

Ainsi, une classe avec une conductance forte est une classe qui est dense en arêtes et faiblement liée avec les autres classes. Un graphe avec une conductance forte a toutes ses classes denses et faiblement liées entre elles.

### La performance

Par souci de simplification, nous décrivons la mesure sur un graphe non valué, bien qu'une version valuée de celle-ci ait été définie (Brandes et al., 2007).

La performance consiste à ajouter le nombre d'arêtes internes aux communautés au nombre d'arêtes intercommunautaires qui n'existent pas, et à diviser la somme par le nombre total d'arêtes possibles du graphe, soit $\frac{1}{2}N(N-1)$ (Van Dongen, 2000).

Soit $|E_{intra}|$ le nombre d'arêtes intraclasses, soit $|\overline{E_{inter}}|$ le nombre d'arêtes interclasses qui n'existent pas dans le graphe :

$$|\overline{E_{inter}}| = \sum_{C_k \in \mathscr{P}} \sum_{C_l \in \mathscr{P}, k>l} \left|\left\{\left\{v, v'\right\} \notin E | v \in C_k, v' \in C_l\right\}\right| \tag{2.44}$$

Alors la performance est définie par :

$$performance(C) = \frac{|E_{intra}| + |\overline{E_{inter}}|}{\frac{1}{2}N(N-1)} \tag{2.45}$$

La performance prend sa valeur entre 0 et 1, une valeur élevée décrivant une classe à la fois dense et peu liée avec d'autres classes.

Almeida *et al.* reprochent à cette mesure d'être mal adaptée à des grands graphes

éparses, dans lesquels le terme $|\overline{E_{inter}}|$ dominera largement dans le score (Almeida et al., 2011).

#### *Coefficient de clustering*

Le coefficient de clustering a été défini par Watts et Strogatz (Watts et Strogatz, 1998). Soit un sommet $v$, ayant $deg(v)$ voisins. Alors au plus $deg(v)(deg(v)-1)$ arêtes peuvent exister entre eux. Le coefficient de clustering $CC(v)$ du sommet $v$ est défini par la proportion des arêtes qui pourraient exister mais n'existent pas dans le voisinage de $v$, par rapport à l'ensemble des arêtes qui pourraient exister. Le coefficient de clustering $CC(G)$ du graphe $G$ est défini par la moyenne des coefficients de clustering des sommets qui le composent :

$$CC(G) = \frac{1}{N} \sum_{v \in V} CC(v) \tag{2.46}$$

### Comparaison des critères internes de qualité d'une partition des sommets

Le comparatif de Yang *et al.* souligne l'intérêt de la conductance mais également la pertinence du coefficient de clustering pour mesurer la structure de communauté (Yang et Leskovec, 2012).

Almeida propose aussi un panorama argumenté des différents critères d'optimisation que sont la modularité, l'indice de Silhouette, la couverture, la performance et la conductance (Almeida et al., 2011). La conclusion de ces travaux est que la modularité, la conductance et la couverture tendent à donner de meilleurs résultats quand le nombre de classes est faible, tandis que la performance et l'indice de Silhouette privilégient eux de petites communautés.

Leskovec *et al.* proposent une comparaison empirique de différentes méthodes de classification des sommets d'un graphe (Leskovec et al., 2010). Ils soulignent le fait que l'optimisation agressive d'un critère comme la conductance peut mener à des communautés trop nombreuses, tandis qu'une optimisation approximative du critère mène à des résultats plus intuitifs.

On pourra également consulter l'étude d'Artignan qui vise à montrer la proximité des résultats de différents algorithmes sur ces différents critères (Artignan et Hascoët, 2011).

On conclura qu'il n'y a pas aujourd'hui de consensus sur une mesure de qualité qui surpasserait toutes les autres. Ainsi, si on ne prend que l'indice de Silhouette et la modularité, ces deux mesures répondent à des intuitions différentes pour lesquelles il est difficile juger si l'une est supérieure à l'autre.

#### 2.3.3.2 Critères externes d'évaluation

Le but du processus de classification étant de produire une partition, les critères externes d'évaluation sont identiques à ceux décrits pour la classification de données non supervisée.

Cependant, des extensions des mesures généralistes dédiées à des domaines d'application existent. C'est par exemple le cas de la mesure proposée par Labatut qui pondère l'influence de la mauvaise classification d'un sommet dans l'indice de pureté par une notion d'importance du sommet (Labatut, 2012). La notion d'importance du sommet est choisie de manière à refléter le fait que le sommet est au cœur de sa communauté. Ainsi, un sommet mal classé pénalisera d'autant plus le critère de pureté modifié qu'il est central à une communauté, tandis que les sommets périphériques apporteront une pénalité plus faible.

### 2.3.4 Conclusion

La détection de communautés dans les graphes est une tâche qui a donné lieu à de nombreux travaux. Si la comparaison entre les méthodes est toujours d'actualité, il faudra cependant choisir entre différents paradigmes dans la définition d'une partition de bonne qualité. Ainsi, la modularité de Newman et la coupure minimale, si elles produisent des résultats différents, devront-elles être choisies selon les besoins de la tâche à effectuer. La première montrera tout son intérêt quand le nombre de classes à produire est inconnu. La seconde permet de produire des résultats hiérarchisés.

On peut voir qu'historiquement les heuristiques et les critères d'optimisation ont évolué en parallèle. Les critères apparaissent comme toujours susceptibles d'être employés dans des algorithmes plus performants et les algorithmes semblent adaptables à de nouveaux critères, bien que certaines combinaisons entre critères et algorithmes sont apparues plus pertinentes ou plus simples à mettre en œuvre.

## 2.4 Détection de communautés dans les réseaux d'information

### 2.4.1 Motivations

La détection de communautés dans des réseaux d'information est motivée par la quantité plus grande de données dont il est possible de tirer parti pour obtenir une classification de meilleure qualité, mais aussi par la complémentarité des informations. Elle est justifiée aussi par les travaux consacrés à l'homophilie (Lazarsfeld et Merton, 1954; McPherson et al., 2001; Crandall et al., 2008; Anagnostopoulos et al.,

2008). L'homophilie exprime la tendance naturelle des individus à s'associer avec des personnes aux caractéristiques proches.

Si la notion d'homophilie amène à penser que les liens apparaissent en priorité entre des éléments aux attributs similaires, d'autres réseaux peuvent avoir une logique différente que certaines méthodes de détection de communautés dans des réseaux d'information peuvent modéliser.

Dans le cas où une source de données est incomplète, l'exploitation d'une source complémentaire permet parfois de compenser les manques. De plus, en comparant des résultats de classification issus de combinaisons d'informations avec des classifications sans combinaison, on peut mettre en évidence des différences qui peuvent elles-mêmes être des connaissances utiles.

En outre, la combinaison des deux informations (relationnelles et d'attributs) peut permettre de trouver des partitions qu'il n'aurait pas été possible de produire en ne considérant que les relations ou que les attributs. C'est ce qui nous a conduit à construire un jeu de données dédié à la problématique de la combinaison d'informations (voir section 1.4.4).

Nous proposons une présentation des méthodes de détection de communautés dans des réseaux d'information en quatre familles. Nous présenterons une première famille de méthodes de classification combinée ayant pour principe de résumer les deux types de données sous la forme d'un graphe dans la section 2.4.3. La deuxième famille qui sera décrite dans la section 2.4.4 comporte des méthodes qui sont issues de l'adaptation d'algorithmes de classification automatique. La troisième famille, dans la section 2.4.5, regroupe des méthodes qui sont des extensions de méthodes de détection de communautés visant à intégrer les attributs. Enfin, dans la section 2.4.6, la quatrième famille comporte des méthodes qui cherchent plus globalement à découvrir le modèle statistique sous-jacent au réseaux d'information.

### 2.4.2 Formalisation du problème de détection de communautés dans un réseau d'information

Soit un graphe $G = (V, E)$ où $V$ est l'ensemble des sommets et $E$ est l'ensemble des arêtes. Pour chaque sommet $v \in V$, on dispose d'un vecteur d'attributs $(v_1, \ldots, v_j, \ldots v_{|T|})$ où $v_j$ est la valeur prise par l'attribut $j$ du sommet $v$.

Dans le problème de partitionnement de réseau d'information, les liens et les attributs sont considérés, de telle sorte que d'une part il doit y avoir de nombreuses arêtes entre les sommets de chaque classe et relativement peu entre elles et d'autre part, deux sommets appartenant à la même classe sont plus proches en termes d'attributs, que deux sommets appartenant à des classes différentes. Ainsi, les classes doivent être

bien séparées en termes de liens et différentes par rapport aux attributs et les sommets appartenant à un même groupe doivent être fortement connectés et homogènes vis-à-vis des attributs.

### 2.4.3 Traitement comme un problème de partitionnement dans un graphe après intégration des valeurs des attributs

La première famille d'approches opérant une combinaison de l'information relationnelle et des attributs que nous présentons passe par la création d'un nouveau graphe dans lequel les deux types d'information sont intégrés.

#### 2.4.3.1 Ajout dans le graphe de nouveaux sommets et d'arêtes représentant les attributs

La construction d'un tel graphe est employée dans les méthodes SA-Cluster et Inc-Cluster (Zhou et al., 2009, 2010). Elle consiste en l'ajout de sommets représentant les différentes valeurs prises par chacun des attributs et aboutit à la création d'un nouveau graphe appelé *Attribute augmented graph*. Il s'agit donc d'un graphe qui contient, en plus des informations du graphe original (le graphe structurel), des sommets représentant une valeur de chacun des attributs (que l'on désignera par *sommet-attribut*). Une *arête-attribut* est créée entre chaque sommet-attribut et les sommets originaux prenant la valeur que ce sommet-attribut désigne.

Par exemple, dans le jeu de données de blogs utilisé par Adamic *et al.* (Adamic et Glance, 2005), chaque sommet (un blog) peut être soit libéral, soit conservateur. Alors deux nouveaux sommets artificiels "libéral" et "conservateur" sont créés, et une arête-attribut est créée entre chaque blog libéral et le sommet-attribut "libéral", de même qu'entre les blogs conservateurs et le sommet-attribut correspondant.

Zhou *et al.* utilisent des k-médoïdes et une distance basée sur une marche aléatoire pour la classification de ce graphe tout en envisageant d'autre méthodes de classification réduites aux relations à cette étape. En 2010, Zhou *et al.* proposent une version incrémentale de SA-Cluster, *Inc-Cluster*, plus efficace sur les grands graphes (Zhou et al., 2010).

On notera que cette approche est adaptée lorsque les attributs sont des étiquettes. Elle n'est pas utilisable dans le cas d'attributs à valeurs discrètes car il faudrait introduire trop de sommets. De plus, il n'y aura plus de prise en compte du caractère ordonné des valeurs des attributs.

Yin *et al.* utilisent aussi les réseaux obtenus en intégrant de nouveaux sommets relatifs aux valeurs d'attributs qu'ils désignent sous le terme de "réseaux sociaux-attributs" (social-attribute network, SAN) mais dans la méthode LinkRec destinée à

la prédiction de liens (Yin et al., 2010). Gong *et al.* utilisent aussi cette représentation pour prédire des liens ou inférer des valeurs d'attributs, notamment en adaptant des méthodes de Yin *et al.* (Gong et al., 2012).

#### 2.4.3.2 Valuation des arêtes par la distance entre vecteurs

Une deuxième façon d'introduire les attributs dans le graphe consiste à valuer les arêtes par une distance calculée entre les vecteurs. C'est l'approche proposée par Neville *et al.* (Neville et al., 2003). La valuation de chaque arête est définie comme le nombre de valeurs d'attributs partagées par ses deux extrémités (similarité du *matching coefficient*). S'il n'y a pas d'arête entre deux sommets aucune valeur de *matching coefficient* n'est attribuée.

Steinhaeuser *et al.* proposent une méthode similaire où les arêtes sont valuées de façon à prendre en compte à la fois la similarité entre attributs et la proximité relationnelle des sommets (Steinhaeuser et Chawla, 2008).

Nous utiliserons nous-même une méthode, $TS_1$, reposant sur la valuation des arêtes par les distances entre attributs qui est décrite dans (Combe et al., 2012b) et qui sera présentée dans la section 3.8.2.3.

#### 2.4.3.3 Combinaison de partitions

Une autre démarche consiste à fusionner diverses partitions issues de la classification textuelle et de la classification relationnelle après la classification portant sur un seul type de données. Les distances entre les éléments ne sont plus utilisées, seule la classe des éléments est prise en compte pour former la partition finale. Pour désigner cette tâche on parle souvent de combinaison de classifieurs, d'*Ensemble clustering* ou de *Consensus clustering* (Ghaemi et al., 2009).

Les partitions, issues de la classification selon les relations et selon les vecteurs, sont représentées par un graphe augmenté, à la manière de ce qui a été présenté dans la section 2.4.3.1. Une solution est de créer des sommets artificiels pour représenter les différentes classes produites par toutes les méthodes de classification, et d'attacher avec ces sommets les sommets d'origine qui y ont été classés (Ghosh et al., 2002). Une détection des communautés permettra ensuite de classer les sommets d'origine. Une seconde représentation possible est celle utilisée par la méthode *Hypergraph Partitioning Algorithm* (HGPA), qui représente chaque communauté par une hyper-arête dans un graphe où les sommets correspondent aux éléments qui ont été classés par les différents classifieurs. Les partitions finales sont trouvées en utilisant des algorithmes faisant appel à la coupure minimum (Strehl et Ghosh, 2003).

L'*ensemble clustering* montre tout son intérêt pour exploiter des algorithmes instables ou lorsqu'on dispose de plusieurs partitions issues de plusieurs algorithmes différents, mais il est limité si l'on n'a que deux partitions différentes.

### 2.4.4 Traitement comme un problème de classification automatique

La deuxième famille de méthodes que nous décrirons traite le problème de détection de communautés comme un problème de classification automatique. Il s'agira alors de modifier une technique de classification afin d'intégrer une notion de distance relationnelle.

Dans cette famille figurent notamment des extensions de l'algorithme des K-means telles que NetScan ou JointClut.

#### 2.4.4.1 Le problème CkC (Connected k-Center)

Ester *et al.* traitent la question sous l'angle du "problème *CkC*", des k-centres connectés, et proposent NetScan, une version étendue de l'algorithme des K-means qui impose une contrainte de connexité interne à chaque classe (Ester et al., 2006; Ge et al., 2008). Sous cette condition, deux sommets d'une classe doivent être reliés par un chemin interne à celle-ci.

Soit un nombre de centres $k$, une contrainte de rayon $r$, une norme $\|\cdot\|$ et un graphe $G = (V, E)$ où chaque sommet de $V$ est associé à un vecteur de coordonnées $w$ de $\mathbb{R}^d$. On cherche s'il existe une partition $\mathscr{P} = \{V_1, \ldots, V_k\}$ de $V$ qui satisfasse les deux conditions suivantes :

1. Les sous-graphes induits $G[V_1], \ldots, G[V_k]$ sont connectés (contrainte de connexité interne).
2. $\forall 1 \leq i \leq k$, il existe un sommet centre $c_i \in V_i$, tel que $\forall v \in V_i, ||w(v) - w(c_i)|| \leq r$ (contrainte de rayon maximal).

On notera que le nombre de centres $k$ est ici fixé à l'avance, tout comme $r$, la distance maximale entre deux membres d'une communauté. De plus, les communautés produites seront forcément connexes. Le problème CkC est démontré comme étant NP-complet mais l'algorithme NetScan est une heuristique qui, d'après les expérimentations menées par les auteurs, permet de résoudre efficacement le problème.

#### 2.4.4.2 Le problème *CXC* (Connected X clusters)

Après l'algorithme NetScan proposé pour résoudre le problème *CkC*, Moser *et al.* introduisent une autre solution, JointClust, ne nécessitant pas de fournir $k$, le nombre de classes à retourner (Moser et al., 2007).

Au lieu de traiter le problème *CkC* de la détection de $k$ communautés connexes, ils résolvent le problème *CXC*. Le problème *CXC* consiste à trouver une partition $\mathscr{P}$ de $G$ telle que :

1. $\mathscr{P}$ remplit la contrainte de taille minimale, c'est-à-dire que chaque classe contient au moins $t$ objets, où $t$ est un paramètre préalablement fixé.
2. $\mathscr{P}$ maximise le *Coefficient de Silhouette Joint*.

Cette approche a pour inconvénient de poser une limite basse au nombre d'objets dans une classe.

JointClust procède en deux phases. La première permet d'identifier des classes atomiques qui sont fusionnées dans la seconde phase, où le résultat peut être présenté sous forme de dendogramme. JointClust maximise le coefficient de Silhouette joint, introduit par les auteurs et décrit dans la section 2.4.7, qui permet de juger de la qualité de la classification en considérant non seulement les attributs mais aussi les relations. Un des avantages de ce critère est qu'il est indépendant du nombre de classes. Ce même critère sera réutilisé par Wan *et al.* dans une méthode hiérarchique (Wan et al., 2009).

Une autre extension des K-means est introduite par Luo *et al.* qui utilise non seulement une distance entre les éléments par rapport aux attributs mais aussi le voisinage de chaque élément défini également par rapport aux attributs. Mais cette méthode pourrait cependant être appliquée en considérant un voisinage défini par rapport à des données relationnelles (Luo et al., 2009). Après avoir choisi des sommets à forte centralité de degrés, on opère des fusions dépendant du nombre de voisins communs et de la proximité des attributs. L'application des K-means est opérée ensuite sur les attributs.

Comme Netscan, cette méthode nécessite de connaître à l'avance le nombre de classes à produire.

### 2.4.5 Extension de la méthode de détection de communautés de Louvain

Si des travaux de la section précédente ont consisté à étendre l'algorithme des K-means, d'autres se sont intéressés à la méthode de Louvain. C'est le cas notamment des travaux de Dang *et al.* ou Cruz-Gomez *et al*.

#### 2.4.5.1 Score de modularité étendu par la similarité des attributs

Dang *et al.* ont étendu la modularité de Newman en ajoutant un terme pour mesurer la similarité basée sur les attributs entre les paires de sommets relevant de la même classe (Dang et Viennet, 2012). De cette façon, ils construisent une mesure de

modularité composite définie comme une combinaison linéaire de la modularité de Newman et Girvan et de la similarité vis-à-vis des attributs. C'est cette mesure composite qui est optimisée dans leur algorithme SAC1, déduit de Louvain. À la différence de la méthode de Louvain, deux sommets peuvent être affectés à une même classe même s'ils ne sont pas reliés. De plus, une des difficultés de la méthode tient au choix du coefficient dans la combinaison linéaire.

Notons que les auteurs proposent également un autre algorithme, SAC2, qui comporte un pré-traitement permettant de générer un graphe des plus proches voisins défini en fonction des liens initiaux et des attributs puis ils appliquent sur ce nouveau graphe une méthode de détection de communautés.

#### 2.4.5.2 Entropie de sommet

Cruz-Gomez *et al.* proposent aussi une méthode de détection de communautés déduite de Louvain et basée sur l'optimisation de deux critères : la modularité pour les relations et l'entropie pour les attributs. L'entropie de la partition est optimisée en suivant l'approche Monte-Carlo (Cruz-Gomez et al., 2011). L'algorithme reprend les principes de Louvain mais en optimisant successivement l'entropie et la modularité afin de minimiser la première et de maximiser la seconde. De ce fait les deux optimisations peuvent provoquer successivement des changements contradictoires.

### 2.4.6 Modèles statistiques

Dans cette quatrième famille, nous présentons les méthodes qui cherchent à modéliser le réseau, au moyen par exemple d'une distribution.

#### 2.4.6.1 Approximation de la loi qui régit la distribution des attributs et les liens

Xu *et al.* modélisent la distribution des attributs et celle des relations dans le réseau par des loi (Dirichlet, etc.) (Xu et al., 2012). Ainsi ils sont en mesure d'évaluer la probabilité pour deux sommets d'être classés ensemble compte tenu des liens et des attributs.

D'autres travaux ont également été proposés par Kim *et al.* qui, pour leur part, utilisent un modèle probabiliste pour capturer des affinités positives et négatives entre les attributs de sorte que seront classés ensemble des éléments qui prennent les mêmes valeurs pour certains attributs ou au contraire des valeurs différentes pour d'autres. Les informations sont inscrites dans une matrice d'affinité qui traite l'interaction entre attributs et relations (Kim et Leskovec, 2010, 2011).

#### 2.4.6.2 Prise en compte des informations vectorielles dans le critère de modularité

D'autres travaux cherchent non pas à intégrer les attributs dans un méthode de détection de communautés existante, mais à modifier ses hypothèses sous-jacentes. Une proposition en ce sens à été faite par Liu *et al.* qui proposent de modifier le modèle nul utilisé dans le calcul de la modularité en ajoutant au degré la prise en compte d'une similarité entre sommets (Liu et al., 2013).

Cette démarche est proche de celle qui nous a amené à proposer la modularité basée sur l'inertie dans le chapitre 4.

### 2.4.7 Évaluation

Dans la littérature, on rencontre principalement deux modes d'évaluation des méthodes développées en classification non supervisée de sommets dans les réseaux d'information. La première d'entre elles est l'évaluation à l'aide de critères *internes*. On cherche alors à obtenir une partition la meilleure possible au sens d'un critère particulier, comme la modularité.

Or face à un probléme réel, la question que l'on se pose est plutôt "La solution que m'apporte l'algorithme est-elle de qualité, a-t-elle de bonnes propriétés ?" que "La solution que m'apporte l'algorithme est-elle la meilleure ?" (ou l'une des meilleures, la meilleure réponse face à un compromis temps / résultat). En particulier, la notion de "bonne solution" varie selon le contexte d'application.

Le second mode d'évaluation est la confrontation des classes fournies par la méthode à des groupes "naturels". On dira alors que l'on évalue le résultat selon un critère *externe*. Le score se calcule alors en terme de précision et de rappel, ou encore de F-mesure.

Si plusieurs mesures hybrides ont été proposées pour opérer la classification dans des réseaux d'information, elles ne peuvent pas réellement être employées en tant que critères d'évaluation, en tout cas si elles sont également utilisées pour construire les classes car l'évaluation sera biaisée. Dans l'état de l'art figure cependant le critère de Joint Silhouette Coefficient proposé par Moser *et al.* et qui permet de quantifier la qualité d'une partition construite sur un réseau d'information en tenant compte des deux types de données : attributs et relations (Moser et al., 2007). Cet indice, inspiré de l'indice de Silhouette de Rousseeuw (Rousseeuw, 1987), est défini par :

$$JSC(\mathscr{P}) = \frac{1}{|V|} \sum_{v \in V} \frac{b_E(v) - a(v)}{max\left\{b_E(v), a(v)\right\}} \tag{2.47}$$

où $a(v)$ est la distance entre l'élément $v$ et le centre de sa classe et $b_E(v)$ est la distance moyenne à toutes les autres classes connectées à la classe $c(v)$.

Cette mesure est donc hybride. Elle est d'autant plus grande que les classes connectées entre elles sont éloignées les unes des autres ($b_E(v)$) et que la dispersion à l'intérieur de chacune des classes est limitée ($a(v)$).

Nous proposons une synthèse des mesures d'évaluation dans la Table 2.1 qui reprend les critères utilisés pour la classification de vecteurs d'attributs, ceux utilisés pour la détection de communautés dans un graphe, ainsi que les critères externes. Ces critères sont aussi utilisables pour la détection de communautés dans un réseau d'information.

| | **Vecteurs** | **Graphe** |
|---|---|---|
| **Critères internes** | Indice de Silhouette (distance euclidienne) (Rousseeuw, 1987)<br>Inertie interclasses<br><br><br><br><br>Davies & Bouldin (Davis et Leinhardt, 1967)<br>Dunn (Dunn, 1973) | Indice de Silhouette (distance géodésique)<br><br>Couverture<br>Conductance<br>Modularité<br>Entropie de sommet (Kantardzic, 2011) |
| **Critères externes** | Taux de bien classés<br>Pureté<br>Indice de Rand (Rand, 1971)<br>ARI (Adjusted Rand Index) (Hubert et Arabie, 1985)<br>AMI (Adjusted Mutual Information) (Vinh et al., 2010)<br>NMI (Normalized Mutual Information) (Strehl et Ghosh, 2003)<br>V-mesure (Rosenberg et Hirschberg, 2007) | |
| **Critères hybrides** | Coefficient de Silhouette Joint (Moser et al., 2007) | |

TABLE 2.1 – Synthèse des critères d'évaluation

### Génération de réseaux d'information

Confrontés au faible nombre de réseaux d'information réels exploitables pour l'évaluation des algorithmes de détection de communautés, quelques travaux ont porté sur leur génération. Nous commençons par un court historique sur la génération de graphes.

**Génération de graphes** L'analyse des réseaux sociaux et de leurs propriétés a provoqué le besoin de générer des réseaux soumis aux mêmes lois. Il existe plusieurs modèles pour générer des réseaux sans attributs. L'un des plus anciens est celui d'Erdős et Rényi (Erdős et Rényi, 1959). Les connexions entre les sommets y sont équiprobables et indépendantes des autres connexions.

Le modèle du petit monde ("small world") de Watts et Strogatz intègre les éléments de la théorie de Milgram des six degrés de séparation dans la mesure où la distance moyenne entre deux individus est faible dans les réseaux à caractères sociaux (Watts et Strogatz, 1998). De plus, le modèle de Watts et Strogatz amène à un *clustering coefficient*, tel que défini dans la section 2.3.3.1, plus élevé que dans les deux modèles abordés précédemment.

Barabási et Albert introduisent le principe d'attachement préférentiel (Barabási et Albert, 1999). Dans ce modèle, chaque nouveau sommet $v$ introduit est attaché avec les sommets précédents selon une probabilité $p_v$ qui dépend du degré de ceux-ci :

$$p_v = \frac{k(v)}{\sum_{v'} k(v')} \tag{2.48}$$

Il est alors possible d'utiliser le principe de la sélection à la roulette (*roulette wheel selection*) pour définir le sommet de rattachement.

C'est un modèle plus réaliste que les précédents, en particulier par le fait qu'il prend en considération la loi de puissance, fréquemment observée sur les réseaux sociaux. On parle aussi de réseaux *sans échelle* (scale-free networks).

Ces modèles de génération sont prévus pour construire des graphes et non des réseaux d'information puisqu'ils ne prennent pas en compte des valeurs d'attributs.

**Génération de réseaux** Nous introduisons maintenant les méthodes qui ajoutent des attributs lors de la génération du graphe.

Kim *et al.* proposent le modèle MAG (*Multiplicative attribute graph*) de génération de réseaux d'information qui considère les sommets décrits par des attributs catégoriels. Dans ce modèle, une matrice permet de modéliser l'interaction entre la valeur d'un attribut et particulier et la probabilité d'existence d'un lien entre une paire de sommets (Kim et Leskovec, 2010).

Les mêmes auteurs proposent ensuite MAGFIT, une méthode d'estimation des paramètres de MAG (Kim et Leskovec, 2011).

Un autre modèle, introduit par Dang *et al.*, génère aussi un réseau en fonction de la similarité des attributs décrivant chacun des sommets. Le principe de cette méthode est le suivant. On commence par générer quelques sommets de chaque catégorie qui joueront le rôle de graines. Lors de l'ajout des sommets suivants, la probabilité qu'une

arête prenne place entre un nouveau sommet $v$ et un sommet $v'$ existant est définie par :

$$p_{v,v'} = \frac{deg(v') \cdot s(v,v') \cdot s_{centres}\left(c_v, c_{v'}\right)}{\sum_{v'' \in V} deg(v'') \cdot s(v,v'') \cdot s_{centres}\left(c_v, c_{v''}\right)} \tag{2.49}$$

où $s(v,v')$ est la similarité entre $v$ et $v'$ définie par :

$$s(v,v') = \frac{1}{|v - v'|} \tag{2.50}$$

où $|v - v'|$ donne la valeur absolue de la différence entre $v$ et $v'$ et où $s_{centres}(c_v, c_{v'})$ est égal à $s(g_{c_v}, g_{c_{v'}})$, où $g_{c_v}$ désigne le centre de gravité de la classe du sommet $v$.

## 2.5 Conclusion

Dans ce chapitre, nous avons d'abord présenté les algorithmes de classification automatique auxquels nous avons eu recours ainsi que les critères qu'ils exploitent, tels que l'inertie interclasses. Nous avons décrit les problèmes posés par la détermination du nombre de classes et l'évaluation de la qualité des partitions, notamment en utilisant des critères internes comme l'indice de Silhouette, de Dunn ou de Davies et Bouldin.

Dans un deuxième temps, nous avons décrit le critère de modularité. Bien que le problème de l'optimisation de la modularité se soit révélé NP-complet, différentes heuristiques d'optimisation du critère que nous avons rappelées permettent de proposer une solution satisfaisante dans un temps raisonnable.

Dans un troisième temps, nous avons abordé la détection de communauté dans des réseaux d'information. Au travers d'une typologie en quatre familles, nous avons soulevé différents inconvénients des méthodes existantes. D'abord, les méthodes procédant par production d'un graphe résumant les deux informations peuvent se révéler inadaptées à des données continues, ignorer certaines distances entre éléments pendant la classification ou combiner les informations en reposant sur des classifications intermédiaires sans ne plus pouvoir prendre en compte les distances entre éléments. Les méthodes issues de la deuxième famille, reposant sur des méthodes de classification automatique, semblent poser moins de problèmes méthodologiques mais posent le problème de la pondération des deux informations dans la partition finale et nécessitent souvent des paramètres comme le nombre de classes à retourner, la taille minimale d'une classe ou la distance maximale entre attributs de deux éléments d'une classe.

La troisième famille, étendant la méthode de Louvain, proposent des méthodes

optimisant des critères soit locaux soit menant à des choix contradictoires. Enfin, la quatrième et dernière famille est composée de méthodes statistiques souvent coûteuses en temps de calcul ou faisant des hypothèses qui pourraient ne pas se révéler pertinentes pour tous les réseaux d'information.

L'état de l'art nous montre que la classification automatique et la classification des sommets d'un graphe sont des domaines bien distincts qui ont chacun leurs approches et leurs critères d'optimisation. Ces domaines ont évolué parallèlement, mais largement indépendamment.

Nous avons vu comment la modularité et l'inertie interclasses pouvaient permettre respectivement de classer des éléments selon les relations entre eux et selon leurs valeurs d'attributs.

Que ce soit pour la classification automatique ou la détection de communautés, il nous semble important d'insister sur le fait que l'évaluation reste aujourd'hui complexe. Le nombre de jeux de données d'évaluation de type benchmark est faible et soulève souvent des problèmes de pertinence ou de confiance dans la manière où la vérité terrain a été construite. Dans le cas où l'évaluation est faite par rapport à une partition précise, on a vu que tout un éventail de mesures, des plus simples aux plus complexes, sont disponibles. Les critères internes posent d'autres questions, qui rejoignent celles rencontrées dans l'évaluation de la détection de communautés dans un graphe ou de classification non supervisée. C'est la raison pour laquelle nous avons généré nous-même un jeu de données qui sera employé dans le chapitre suivant et avons eu recours à un générateur.

Dans le chapitre suivant, nous proposons une nouvelle méthode, ToTeM. À la différence de la méthode de Cruz-Gomez *et al.*, ToTeM optimise un critère tenant compte de la qualité globale de la partition par rapport aux attributs et aux relations, plutôt que d'optimiser successivement deux critères. En ce sens elle se rapproche d'avantage de la méthode SAC1 de Dang *et al.*, mais, alors que SAC1 considère la similarité entre les paires de sommets relevant de la même classe, ToTeM optimise une mesure d'inertie interclasses.

CHAPITRE 3

# ToTeM, une méthode de détection de communautés utilisant modularité et inertie

**Sommaire**



## 3.1 Introduction

Dans le chapitre 2 nous avons d'abord vu différentes manières de classifier des éléments décrits par des vecteurs de façon non supervisée. Nous avons ensuite décrit différentes façons de classifier les sommets dans un graphe, puis nous avons présenté des méthodes de détection de communautés dans des réseaux d'information combinant les deux types de données relationnelles et d'attributs.

Parmi ces approches, certaines sont dédiées à la classification de sommets pour lesquels les attributs sont discrets, comme SA-Cluster (Zhou et al., 2009). D'autres sont des approches où la combinaison effective des données relationnelles et des données d'attributs est effectuée tardivement dans le processus de classification, ce qui peut apporter un déséquilibre dans la prise en compte des deux types de données (Lu et

Getoor, 2003). Certaines de ces approches font de plus usage de critères d'optimisation locaux, qui apportent peu de garanties formelles sur le résultat final, concernant les partitions qui auront été mises de côté (Lu et Getoor, 2003; Cruz-Gomez, 2012). Dans ce chapitre nous présentons notre première proposition, la méthode ToTeM, conçue pour répondre à ces critiques.

La méthode a plusieurs objectifs. Elle doit tout d'abord produire des classes qui sont denses en liens, et où les vecteurs associés aux sommets sont similaires. En complément, ces classes doivent être peu liées entre elles par des arêtes, et les vecteurs associés à deux sommets de classes différentes doivent être aussi éloignés que possible. Le contexte d'application de réseaux bibliographiques qui est le notre nécessite de pouvoir manipuler des attributs à valeurs continues, en particulier les poids *tf-idf* des termes présents dans les documents. Nous voulons aussi proposer une méthode qui soit utilisable quelle que soit la densité des graphes et quelles que soient les moyennes et les écarts types des attributs manipulés. L'influence des deux sources d'information doit être équilibrée. De plus, cette méthode doit permettre de classifier les sommets sans effectuer une fusion prématurée des deux types d'informations, comme préconisé en conclusion du chapitre précédent. Enfin, nous basons notre méthode sur deux critères reconnus qui n'ont à notre connaissance jamais été rapprochés : l'inertie et la modularité.

Alors que la méthode de Louvain ne classifie les sommets que sur la base des relations, ToTeM prend également en compte les attributs continus durant la classification, tout en proposant une simplification des calculs relatifs à ces attributs, à la manière de ce que Newman a initié et Blondel a étendu pour le critère de modularité.

Après avoir formalisé le problème dans la section suivante, la méthode ToTeM sera décrite dans la section 3.3. Les optimisations implantées dans la méthode seront développées dans la section 3.4. La complexité de la méthode sera analysée dans la section 3.5. Les différents critères globaux de qualité que nous envisageons sont décrits dans la section 3.6. Nous évaluerons les performances de ToTeM sur des réseaux artificiels dans la section 3.7 puis sur des réseaux d'information d'origine bibliographique dans la section 3.8 et dans la section 3.9. Nous conclurons dans la section 3.10.

## 3.2 Formalisation

Étant donné un graphe $G = (V, E)$ où $V$ est l'ensemble des sommets et $E \subset V \times V$ est l'ensemble des arêtes éventuellement valuées. Dans la suite, on notera $\mathcal{A}$ la matrice d'adjacence de $G$ telle que $\mathcal{A}_{vv'}$ indique la valuation de l'arête entre $v$ et $v'$ si elle existe et vaut 0 sinon. Le degré du sommet $v$, noté $k(v)$, est égal à $\sum_{v' \in V} \mathcal{A}_{vv'}$.

On considère les sommets de l'ensemble $V$ comme des éléments d'un espace vec-

toriel à $|T|$ dimensions. Chaque élément $v$ de $V$ est associé à un vecteur d'attributs $\vec{v}$, que par souci de simplification des notations nous noterons simplement $v$ :

$$v = (v_1, \ldots, v_{|T|}) \tag{3.1}$$

De plus, chaque sommet est associé à une masse $m_v$. Dans le graphe initial, la masse de chaque sommet est égale à 1.

Nous nous limiterons au cas où le processus de détection de communautés consiste à partitionner l'ensemble $V$ des sommets en $r$ classes distinctes $\mathscr{P} = \{C_1, \ldots, C_r\}$ où $r$ est a priori inconnu. Les classes à produire réalisent une partition de $V$ : elles ne se recouvrent pas et chaque classe comporte au moins un élément, selon les définitions proposées dans la section 2.2.1.

- $\bigcup_{k\in\{1,\ldots,r\}} C_k = V$
- $C_k \cap C_l = \emptyset$, $\forall\ 1 \leq k < l \leq r$
- $C_k \neq \emptyset$, $\forall k \in \{1, \ldots, r\}$

Enfin, on note $c_v$ la classe d'appartenance du sommet $v$ dans la partition $\mathscr{P}$.

Dans certains algorithmes de détection de communautés, comme par exemple celui de Newman *et al.* basé sur l'intermédiarité, les classes recherchées doivent être connexes (Newman et Girvan, 2004). On dit qu'une classe $C$ est connexe au sein du graphe $G$ si entre toute paire de sommets de cette classe on peut trouver un chemin composé de sommets de cette même classe.

$$\begin{aligned}&\forall (v, v') \in C \times C\ \exists v_1, v_2, \ldots, v_n \in C \mid (v, v_1) \in E,\ (v_1, v_2) \in E,\\ &\ldots,\ (v_{n-1}, v_n) \in E,\ (v_n, v') \in E\end{aligned} \tag{3.2}$$

Nous adoptons le principe selon lequel les communautés à découvrir sont sinon toujours connexes, le plus souvent proche de la connexité, et que ceci est utile dans leur construction. C'est une hypothèse forte dans le sens où elle empêchera de rassembler des individus identiques mais isolés parce qu'ils ne sont pas les extrémités d'un chemin d'individus appartenant à la même classe. Une telle restriction à des communautés connexes est présente dans les travaux d'Ester *et al.* (Ester et al., 2006) alors qu'elle ne l'est pas dans ceux de Dang *et al.* (Dang et Viennet, 2012).

Nous verrons néanmoins dans la section 3.6.3 que l'on ne garantira pas strictement la connexité des classes.

## 3.3 La méthode ToTeM

La méthode ToTeM vise à construire une partition des sommets du réseau en considérant à la fois les relations qui existent entre les sommets et leurs attributs.

Notre proposition est une extension d'une méthode de partitionnement de graphe qui a provoqué un intérêt fort dans la communauté de l'analyse des réseaux sociaux : la méthode dite de Louvain (Blondel et al., 2008). La méthode de Louvain a pour avantages sa vitesse, sa simplicité et le fait qu'elle repose à la fois sur un critère (la modularité) lui-même très étudié et sur un principe intuitif comme exposé dans la section 2.3.2.5. ToTeM conserve la même logique exploratoire mais repose sur l'utilisation d'un critère de qualité hybride. Ce critère permet de classer les sommets en se souciant à la fois de la qualité des classes d'un point de vue relationnel mais également du point de vue des attributs.

La méthode fait appel aux notions de modularité et d'inertie interclasses, notions importantes dans l'état de l'art, respectivement en détection de communautés dans les graphes (voir section 2.3.2.2) et en apprentissage non supervisé (voir section 2.2.3.1).

Un exemple de critère que la méthode pourra optimiser est le suivant :

$$CG1 = \frac{I_{inter}(\mathscr{P})}{I(V)} \cdot Q_{NG}(\mathscr{P}) \tag{3.3}$$

où $I_{inter}(\mathscr{P})$ est l'inertie interclasses associée à la partition $\mathscr{P}$.

ToTeM comprend deux phases principales. La phase itérative consiste à optimiser un critère de qualité jusqu'à convergence vers un optimum local tandis que la seconde a pour but de synthétiser les données du réseau d'information dans le but d'échapper à des optima locaux. Les différentes étapes de cette méthode, décrites ci-après, seront présentées sur un exemple décrivant un réseau d'information à 9 sommets caractérisés chacun par une valeur réelle indiquée entre crochets sur le sommet à la figure 3.1.

### 3.3.1 Initialisation

Initialement, chaque sommet est affecté à une classe qui lui est propre et il est décrit par un vecteur à valeurs réelles. L'algorithme démarre donc avec la partition discrète.

### 3.3.2 Phase itérative

Après l'initialisation arrive la phase itérative. Celle-ci consiste pour chaque sommet à mesurer l'amélioration du critère global obtenue en le plaçant dans la classe d'un de ses sommets voisins. Si il y a une amélioration effective du critère, le sommet

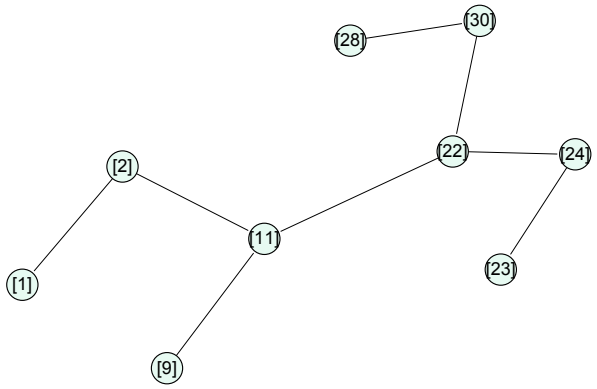



FIGURE 3.1 – Réseau d'information d'illutration

est affecté à la classe pour laquelle cette amélioration est la plus grande. Si aucune amélioration n'est possible, le sommet reste dans sa classe initiale. Le critère global est basé à la fois sur la modularité et sur l'inertie interclasses. On itère ainsi sur l'ensemble des sommets jusqu'à ce qu'aucun déplacement ne soit plus observé sur une itération entière.

Dans l'exemple illustratif, la phase itérative consistera ainsi à prendre un premier sommet, dans la figure 3.2 par exemple le sommet d'attribut 11, et à calculer la valeur du critère global lorsque ce sommet est placé avec le sommet d'attribut 2 ou 9 ou 22. Si le critère a une valeur supérieure quand le sommet d'attribut 11 est placé avec le sommet d'attribut 9, alors on place ces deux sommets dans une communauté conjointe. Si le sommet d'attribut 11 avait eu une valeur de critère supérieure en étant placé seul, alors il serait resté dans sa propre classe. Un exemple d'état du réseau à la fin de cette phase est donné par la figure 3.3

### 3.3.3 Phase de fusion des sommets

Lors de cette seconde étape, qui est exécutée lorsque plus aucun sommet ne peut être déplacé en provoquant un gain, on construit un nouveau graphe en fusionnant tous les sommets d'une classe en un nouveau sommet. Les sommets ainsi associés le seront alors définitivement. La fusion concerne d'une part les données relationnelles et d'autre part les données d'attributs et elle est détaillée dans les sous-sections suivantes.

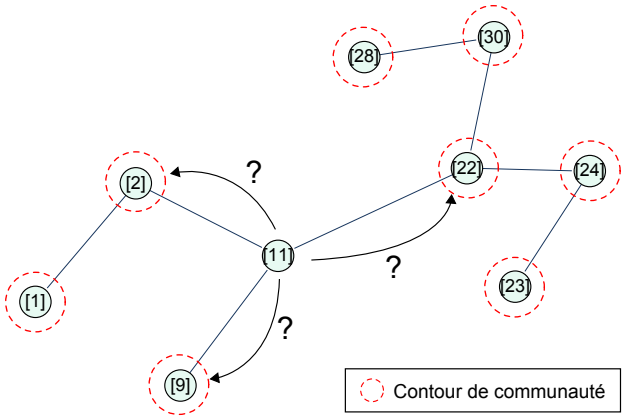



FIGURE 3.2 – Phase itérative

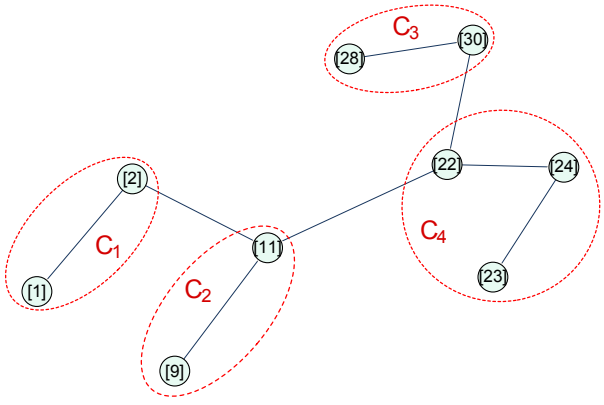



FIGURE 3.3 – Partition obtenue à la fin de la phase itérative

#### 3.3.3.1 Synthèse des informations relationnelles

L'opération de synthèse des informations du graphe consiste, à l'instar de ce qui est opéré dans la méthode de Louvain, à fusionner les sommets affectés à une même classe de façon à n'en faire qu'un seul sommet. Ainsi, à partir de la partition $\mathscr{P}' = (C_1, \ldots, C_r)$ obtenue à l'issue de la phase d'optimisation, un nouveau graphe $G' = (V', E')$ est crée. Ce graphe comporte autant de sommets qu'il y a de classes dans $\mathscr{P}'$ et chaque sommet $v'_l$ de $V'$ incarne une classe $C_l$ de $\mathscr{P}'$. La valuation de l'arête éventuellement présente entre les sommets $v'_y$ et $v'_z$ de $V'$ est égale à la somme des valuations des arêtes présentes entre des sommets de $G$ appartenant aux classes $C_y$ et $C_z$ de $\mathscr{P}'$.

Soit $\tau$ la fonction qui indique, pour un sommet $v$ de $V$, par quel sommet $v'$ de $V'$ il est représenté, alors la valuation associée à une arête est calculée de la façon suivante :

$$\mathcal{A'}_{v'_y,v'_z} = \sum_{(v_a,v_b)\in V\times V} \mathcal{A}_{v_a,v_b} \cdot \delta(\tau(v_a), v'_y) \cdot \delta(\tau(v_b), v'_z) \tag{3.4}$$

où $\mathcal{A}$ est la matrice d'adjacence de $G$, $\mathcal{A}'$ est la matrice d'adjacence de $G'$ et $\delta$ est la fonction de Kronecker.

Il est à souligner que ceci est aussi applicable aux arêtes internes aux classes de $\mathscr{P}$ ; celles-ci deviennent des boucles dans $G'$.

Cette opération est illustrée par la figure 3.4 où une boucle est créée sur les sommets représentant les communautés $C_1$, $C_2$ et $C_3$, car ces dernières contenaient chacune une arête interne. La communauté $C_4$ contenant deux arêtes internes, deux boucles sont créées sur le sommet correspondant. Chaque nouveau sommet est lié aux sommets voisins de la même façon que la communauté qu'il représente était liée aux autres communautés.

#### 3.3.3.2 Synthèse des informations des attributs

Après avoir opéré la fusion des éléments d'un point de vue relationnel, comme exposé dans la section précédente, il est nécessaire de transférer les informations relatives aux attributs sur le nouveau graphe $G'$.

Pour cela, on affecte les masses $m_{C_l}$ des classes d'origine aux sommets correspondants dans $G'$. De plus, le centre de gravité de la classe $C_l$ devient le vecteur d'attributs du sommet $v'_l$ de $V'$ correspondant à $C_l$. Ainsi, pour tout sommet $v'_l$ de $V'$ résultant de la classe $C_l$ de $\mathscr{P}'$ on a :

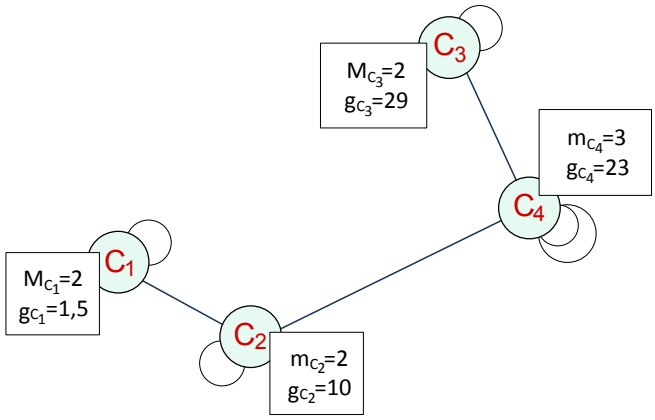



FIGURE 3.4 – Fin de la phase de fusion des sommets

$$m_{v'_l} = m_{C_l} \tag{3.5}$$

$$v'_l = g_{C_l} \tag{3.6}$$

Sur la figure 3.4, on constate que dans le graphe des communautés $G'$ image du graphe $G$ les sommets sont associés aux masses et aux centres de gravité qui caractérisaient chaque communauté de la partition des sommets de $G$.

La méthode ToTeM est détaillée dans l'algorithme 4.

## 3.4 Optimisation du calcul de la modularité et de l'inertie

De même que dans l'algorithme de Louvain, le calcul de la modularité peut être optimisé en tenant compte uniquement des changements induits localement par le déplacement d'un sommet. Nous montrons dans cette section qu'il en va de même pour l'inertie interclasses.

D'après Blondel *et al.*, le gain de modularité induit par le déplacement d'un sommet isolé $u$ vers une classe $B$ est égal à (Blondel et al., 2008) :

$$\Delta Q = \left[\frac{\sum_{in}+2k_{u,in}}{2M} - \left(\frac{\sum_{tot}+k(u)}{2M}\right)^2\right] - \left[\frac{\sum_{in}}{2M} - \left(\frac{\sum_{tot}}{2M}\right)^2 - \left(\frac{k(u)}{2M}\right)^2\right] \tag{3.7}$$

où $k(u)$ et le degré valué du sommet $u$, $M$ est la somme des valuations de toutes les arêtes du graphe, $\sum_{in}$ est la somme des poids des arêtes ayant leurs deux extrémités dans la classe $B$, $\sum_{tot}$ est la somme des poids des arêtes adjacentes aux sommets de

**Algorithme 4** : ToTeM

```
Entrées : Réseau d'information G
Sorties : Partition 𝒫
répéter
    fin ← faux;
    𝒫 ← partition discrète des sommets de V;
    CG_antérieur ← valeur du critère global de la partition 𝒫;
    répéter
        pour chaque sommet u de G faire
            B ← communauté voisine maximisant le gain du critère de qualité;
            si le placement de u dans B induit un gain strictement positif alors
                Mettre à jour la partition 𝒫 suite au transfert de u dans B
    jusqu'à ce qu'aucun sommet ne puisse plus être déplacé ;
    si le critère de qualité global de 𝒫 est supérieur à CG_antérieur alors
        Fusionner G en un réseau d'information entre les classes;
    sinon
        fin ← vrai ;
jusqu'à fin ;
𝒫 ← 𝒫 partition des sommets de V
```

$B$, $k_{u,in}$ est la somme des poids des arêtes reliant $u$ aux sommets de $B$ (Blondel et al., 2008).

De même, dans ToTeM, l'efficience de l'algorithme peut être améliorée en remarquant que la variation d'inertie interclasses induite par le déplacement d'un sommet de sa classe vers celle de l'un de ses voisins peut se calculer avec uniquement une information locale. Ainsi, comme pour la modularité, il est possible de calculer le gain de inertie interclasses induite par le déplacement d'un sommet $u$ d'une classe $A$ vers une classe $B$.

Considérons deux partitions $\mathscr{P}$ et $\mathscr{P}'$ telles que : $\mathscr{P} = (A, B, C_1, \ldots, C_r)$ et $\mathscr{P}' = (A \setminus \{u\}, B \cup \{u\}, C_1, \ldots, C_r)$.

Par la suite, $A \setminus \{u\}$ désigne la classe $A$ privée du sommet $u$ et $B \cup \{u\}$ la classe $B$ augmentée du sommet $u$.

L'inertie interclasses $I_{inter}(\mathscr{P})$ associée à la partition $\mathscr{P}$ est égale à :

$$I_{inter}(\mathscr{P}) = m_A \|g_A - g\|^2 + m_B \|g_B - g\|^2 + \sum_{l=1,\ldots,r} m_l \|g_l - g\|^2 \qquad (3.8)$$

où $g_A$ est le centre de gravité de $A$, $g_B$ est le centre de gravité de $B$, $m_A$ est le poids

de $A$ et $m_B$ est le poids de $B$.

L'inertie interclasses de la partition $\mathscr{P}'$ obtenue en retirant $u$ de sa classe $A$ et en l'affectant à la classe $B$ vaut :

$$I_{inter}(\mathscr{P}') = m_{A\setminus\{u\}} \left\| g_{A\setminus\{u\}} - g \right\|^2 + m_{B\cup\{u\}} \left\| g_{B\cup\{u\}} - g \right\|^2 + \sum_{l=1,\dots,r} m_l \left\| g_l - g \right\|^2 \tag{3.9}$$

La variation d'inertie interclasses induite par le déplacement du sommet $u$ de la classe $A$ vers la classe $B$ est donnée par :

$$\begin{aligned}
\Delta I_{inter} =& I_{inter}(\mathscr{P}') - I_{inter}(\mathscr{P}) && (3.10)\\
=& m_{A\setminus u} \cdot \left\| g_{A\setminus\{u\}} - g \right\|^2 + m_{B\cup u} \cdot \left\| g_{B\cup\{u\}} - g \right\|^2 + \sum m_l \left\| g_l - g \right\|^2 \\
& - m_A \cdot \left\| g_A - g \right\|^2 - m_B \cdot \left\| g_B - g \right\|^2 - \sum m_l \left\| g_l - g \right\|^2 && (3.11)\\
=& (m_A - m_u) \cdot \left\| g_{A\setminus\{u\}} - g \right\|^2 + (m_B + m_u) \cdot \left\| g_{B\cup\{u\}} - g \right\|^2 \\
& - m_A \cdot \left\| g_A - g \right\|^2 - m_B \cdot \left\| g_B - g \right\|^2 && (3.12)
\end{aligned}$$

$g_{A\setminus\{u\}}$ et $g_{B\cup\{u\}}$ sont eux aussi calculés facilement en considérant le sommet $u$ et les classes $A$ et $B$ ainsi que leurs centres de gravités $g_A$, $g_B$ :

$$\begin{aligned}
g_{A\setminus\{u\}} &= \frac{1}{m_{A\setminus\{u\}}} \sum_{v\in A\setminus\{u\}} m_v v && (3.13)\\
&= \frac{1}{m_A - m_u}(m_A \cdot g_A - m_u \cdot u) && (3.14)\\
g_{B\cup\{u\}} &= \frac{1}{m_B + m_u}(m_B \cdot g_B + m_u \cdot u) && (3.15)
\end{aligned}$$

On notera qu'à la suite du déplacement, la classe $A$ peut se retrouver vide et donc disparaître. Sa contribution à l'inertie interclasses devient alors nulle.

Les valeurs des masses associés aux classes peuvent aussi être recalculées à l'aide de l'information locale :

$$\begin{aligned}
m_{A\setminus\{u\}} &= m_A - m_u && (3.16)\\
m_{B\cup\{u\}} &= m_B + m_u && (3.17)
\end{aligned}$$

## 3.5 Complexité

Brandes *et al.* ont prouvé que l'optimisation de la modularité était un problème NP-complet (Brandes et al., 2007). Les auteurs de la méthode de Louvain reconnaissent à celle-ci une complexité théorique en $\mathcal{O}(|E|^3)$ dans le pire des cas. Cependant, dans la pratique, les temps d'exécution sont plutôt ceux d'un processus en temps linéaire (Aynaud et al., 2010; Seifi, 2012). Ceci s'explique par la convergence très rapide du critère de modularité, qui ne rencontre pas de cas aussi défavorable que l'algorithme peut le laisser supposer.

Si pour la méthode de Louvain le déplacement d'un sommet d'une classe à l'autre est une opération élémentaire, cette opération devient linéaire avec la prise en compte des attributs ($\mathcal{O}(|T|)$). On a vu dans la section 3.3.3.2 que la synthèse des informations des attributs lors de la phase 2 était elle immédiate. La vitesse de calcul est aussi affectée par les calculs liés au calcul du critère global, qui pour le critère $CG2$ que nous préconisons et qui sera décrit ultérieurement, ont une complexité élémentaire, donc sans incidence sur la complexité théorique. On a donc une complexité, pour un déplacement et dans le pire des cas, de $\mathcal{O}(|\mathcal{T}|)$. Ceci nous amène à une complexité dans le cas le plus défavorable en $O(|T| \times |E|^3)$. Comme pour la méthode de Louvain, ceci doit être pris en compte comme une borne théorique qui n'est pas rencontrée dans la pratique. Une amélioration possible est la mise en mémoire des valeurs de critères déjà calculées, car l'heuristique d'exploration calcule la qualité de certaines partitions de multiples fois, surtout à la fin de la phase itérative. Cependant ceci nécessite la mise en place de structures de stockage adéquates permettant de reconnaître des partitions, ou des classes, qui ont déjà été rencontrées.

## 3.6 Critères globaux de qualité

Si l'utilisation de la modularité, couplée à l'heuristique de la méthode de Louvain, est efficace pour détecter les communautés structurelles dans les graphes, le critère que nous devons utiliser doit aussi séparer les sommets quand leurs valeurs d'attributs divergent.

Ce critère de qualité globale intervenant dans ToTeM doit donc être une fonction d'une mesure de la qualité de la partition par rapport aux relations et d'une mesure de sa qualité par rapport aux attributs. La modularité $Q_{NG}$ (définie par l'équation 2.38) peut être utilisée comme mesure de la qualité par rapport aux relations. Pour ce qui est de la qualité par rapport aux attributs, une première solution envisageable peut consister à prendre le taux d'inertie interclasses. Ce qui conduit à une première mesure de qualité globale définie par :

$$CG1 = \frac{I_{inter}(\mathscr{P})}{I(V)} \cdot Q_{NG}(\mathscr{P}) \tag{3.18}$$

où $I_{inter}(\mathscr{P})$ désigne l'inertie interclasses de $\mathscr{P}$ et $I(V)$ désigne l'inertie de $V$.

On justifie un tel critère par le besoin de créer des classes aussi distinctes que possible du point de vue des attributs (donc avec une inertie interclasses forte). Comme l'inertie interclasses est bornée par l'inertie totale et que la modularité est elle même bornée par la valeur 1, cet indice est lui aussi borné par 1. Cependant, l'inconvénient de ce critère global est que l'inertie interclasses n'est pas conçue pour comparer des partitions ayant un nombre de classes différent. En effet, le taux d'inertie interclasses varie structurellement avec le nombre de classes de la partition de sorte qu'il est maximum pour la partition discrète. Une solution simple visant à palier ce biais structurel consiste à tenir compte du nombre de classes de la partition pour définir un critère global. Ceci nous mène à la définition d'un deuxième critère :

$$CG2 = \frac{I_{inter}(\mathscr{P})}{|\mathscr{P}| \cdot I(V)} \cdot Q_{NG}(\mathscr{P}) \tag{3.19}$$

où $|\mathscr{P}|$ désigne le nombre de classes de $\mathscr{P}$. Contrairement au précédent, ce critère donne un avantage aux partitions à faible nombre de classes, celles que l'inertie interclasses a en pratique une forte tendance à pénaliser. Une autre alternative pour palier le biais structurel consiste à avoir recours à des indices conçus dans le but d'optimiser le nombre de classes dans le cas du partitionnement de données vectorielles, comme par exemple l'indice de Calinski-Harabasz.

### 3.6.1 Indice de Calinski-Harabasz

Ainsi l'indice de Calinski-Harabasz $CH$ peut-il être introduit au sein du critère global à la place de l'inertie interclasses (Calinski et Harabasz, 1974). Il est défini par :

$$CH(\mathscr{P}) = \frac{I_{inter}(\mathscr{P})/(|\mathscr{P}| - 1)}{I_{intra}(\mathscr{P})/(|V| - |\mathscr{P}|)} \tag{3.20}$$

où $I_{intra}$ est l'inertie intraclasses. Ce critère permet de comparer deux partitions quelque soit leurs nombres de classes, tandis que l'inertie interclasses ne permet que de comparer des partitions avec un même nombre de classes. On peut également avoir recours à d'autres indices de validation internes de détection de communauté comme l'indice de Dunn ou de Davies-Bouldin (voir section 2.3.3.1) (Davies et Bouldin, 1979; Duda et Hart, 1973; Baker et Hubert, 1975; Milligan et Cooper, 1985).

### 3.6.2 Probabilité critique

Une autre solution pour comparer deux partitions $\mathscr{P}$ et $\mathscr{P}'$ consiste à faire appel à des tests statistiques. En effet, sous l'hypothèse nulle selon laquelle les éléments sont répartis aléatoirement au sein de la partition $\mathscr{P}$, la statistique $F(\mathscr{P})$ définie par :

$$F_{\mathscr{P}} = \frac{I_{inter}/(|\mathscr{P}| - 1)}{(I(V) - I_{inter})/(|V| - |\mathscr{P}|))} \tag{3.21}$$

suit une loi de Fisher-Snedecor $F(|\mathscr{P}| - 1, |V| - |\mathscr{P}|)$ à $(|\mathscr{P}| - 1, |V| - |\mathscr{P}|)$ degrés de liberté.

On peut donc calculer la probabilité critique $PC$ associée :

$$PC = P(F(|\mathscr{P}| - 1, |V| - |\mathscr{P}|) > F_{\mathscr{P}}) \tag{3.22}$$

De même, la statistique $F(\mathscr{P}')$ peut être déterminée sur la partition $\mathscr{P}'$ et on peut en déduire $PC'$ :

$$PC' = P(F(\left|\mathscr{P}'\right| - 1, |V| - \left|\mathscr{P}'\right|) > F_{\mathscr{P}'}) \tag{3.23}$$

La comparaison des probabilités critiques associées à $\mathscr{P}$ et à $\mathscr{P}'$ conduira à préférer la partition pour laquelle cette probabilité est la plus faible.

De la même façon que dans le contexte de la classification automatique les mesures d'agrégation et de proximité sont difficiles à choisir, les critères globaux pour la classification des sommets d'un réseau d'information ne font pas l'objet d'un consensus. Leur choix dépend de la sémantique des informations, de la dominance éventuelle des attributs sur les relations ou inversement, ainsi que des problématiques de contrôle de la taille des classes produites les unes par rapport aux autres.

### 3.6.3 Remarque sur l'utilisation de Louvain pour l'optimisation d'un score différent de la modularité

L'heuristique de Louvain a inspiré de nombreux travaux. Certains, dont notre proposition, modifient le critère à optimiser. Au cours de l'heuristique, les fusions de communautés ne sont opérées qu'entre communautés reliées au moins par une arête. Il est nécessaire de garder à l'esprit que si l'optimisation de la modularité donne des communautés finales connexes, l'optimisation d'un autre critère ne garantit pas un tel résultat. Cette mise au point est faite par exemple par Martelot et Hankin (Le Martelot et Hankin, 2013). Utilisant divers critères dans leur tâche de détection de communautés relationnelles, ils suggèrent, lorsqu'un changement d'affectation de sommet a été jugé préférable, de vérifier auparavant si celui-ci casse une communauté existante. En effet, si le sommet à transférer était le seul pont entre deux sous-ensembles

de sommets de la communauté, alors cette dernière se retrouvera scindée. Les deux sous-groupes de sommets conserveront malgré tout une information enregistrée les affectant dans la même communauté mais cette dernière ne sera pas connexe.

Martelot et Hankin préfèrent ne pas opérer le changement d'affectation si le cas se présente. La technique de vérification utilise un parcours en largeur, au sein de la communauté, afin de savoir si tous les sommets sont atteignables depuis un point de la communauté, malgré le changement d'affectation. Ils expliquent que la modularité offre une garantie contre ce phénomène. Mais, en toute généralité, les critères ne fournissent pas cette garantie structurelle.

Pour notre part, nous préférons ne pas garantir la connexité des classes par souci d'efficience. Il appartiendra à l'utilisateur de modifier l'algorithme de ToTeM en suivant la démarche de Martelot et Hankin pour assurer cette connexité si elle est nécessaire.

Un autre point à soulever concernant le choix du critère à optimiser concerne sa cohérence face à la fusion des sommets dans la seconde phase. Afin que l'exploration des partitions soit cohérente, et surtout respecter l'unicité du score pour une partition donnée, le critère doit respecter l'égalité entre les scores de la partition des deux graphes entre le début et la fin de la phase de fusion. L'utilisation des masses garantit ce comportement dans notre usage de l'inertie interclasses.

## 3.7 Évaluation sur des réseaux artificiels

Afin d'étudier le comportement de l'algorithme face à des méthodes prenant en compte uniquement les attributs ou les relations, nous avons généré automatiquement des graphes. À partir d'un graphe de référence pour lequel les partitionnements basés sur les attributs ou les relations sont très proches, nous avons dégradé tantôt les attributs, tantôt les relations. Notre objectif est d'évaluer l'impact de cette dégradation. Nous avons appliqué les trois algorithmes que sont les K-means, pour un partitionnement sur les attributs, Louvain, pour un partitionnement selon les arêtes et ToTeM, notre méthode hybride, pour les réseaux d'information.

L'implémentation de la méthode de Louvain que nous utilisons est celle de Thomas Aynaud réalisée en langage Python, proposée en 2009 [1] qui prend en compte la valuation des arêtes pour la détection des communautés. On précise que le critère qui est utilisé pour ToTeM est :

$$CG2(\mathscr{P}) = \frac{I_{inter}(\mathscr{P}) \times Q_{NG}}{|\mathscr{P}| \times I(V)} \tag{3.24}$$

1. `http://perso.crans.org/aynaud/communities/`

Dans la suite, nous modifions plusieurs paramètres : le nombre de sommets, la densité du graphe, la pertinence du placement des arêtes au regard de la catégorie d'origine des sommets ainsi que l'écart-type des distributions des attributs à l'intérieur des classes. La "détérioration" des attributs consiste à rendre les valeurs de différents sommets de moins en moins caractéristiques d'une classe particulière. La détérioration des relations changera des arêtes intraclasses en arêtes interclasses. Selon notre hypothèse, la méthode ToTeM préservera la partition d'origine au-delà de ce que pourront faire la méthode de Louvain ou les K-means, lors de l'application des dégradations du réseau.

Pour notre étude, nous partirons d'un graphe dont le découpage en partitions par une méthode aussi bien basée sur les relations que sur les attributs donne de bons résultats avec peu d'ambiguïté. Nous créons un graphe de départ ayant trois classes composées de 33 sommets chacune. À partir de ce réseau de référence noté R dans la suite, nous avons construit les familles de réseaux :

- R.1.x dans lesquels l'information relationnelle est dégradée par rapport à R,
- R.2.x dans lesquels les valeurs des attributs sont moins représentatives de chaque classe,
- R.3.x qui comportent plus de sommets que R,
- R.4.x qui comportent plus d'arêtes que R.

Nous comparons pour chaque graphe ainsi généré les partitions données par ToTeM, la méthode de Louvain et les K-means.

L'évaluation sera faite sur le nombre de classes trouvées, le taux de sommets bien classés (TBC), l'information mutuelle normalisée (NMI), la modularité ($Q_{NG}$), l'inertie interclasses ($I_{inter}$) et deux indices de silhouette utilisant pour le premier la distance géodésique (silhouette-Liens) et pour le second la distance entre les vecteurs associés à deux sommets (silhouette-Attributs) (voir section 2.2.3.1). Ces indices, sur critères internes et externes, nous permettront de savoir si la partition retournée est bien proche de la vérité terrain (TBC, NMI) et si elle a des classes compactes du point de vue des liens comme des attributs ($Q_{NG}$, $I_{inter}$, silhouette-Liens et silhouette-Attributs).

Les résultats obtenus sur le réseau de référence et ses caractéristiques sont décrits dans la prochaine section, les suivantes sont consacrées aux autres réseaux.

### 3.7.1 Réseau de référence (R)

On génère, à l'aide du modèle proposé par Dang (Dang, 2012) et décrit dans la section 2.4.7, un réseau de référence R.

Pour construire ce graphe de référence, nous introduisons 99 sommets uniformé-

ment répartis entre 3 catégories. Chaque sommet est décrit par une valeur d'attribut réelle qui suit une loi normale d'écart-type 7, centrée autour d'une valeur propre à sa classe d'origine. Ainsi la première classe a un centre $\mu_1 = 10$, la deuxième un centre $\mu_2 = 40$ et la troisième un centre $\mu_3 = 70$. La classe d'origine du sommet sert de vérité terrain pour l'évaluation. Enfin, durant la génération du graphe de référence, nous avons choisi que le processus de génération des arêtes crée au maximum deux arêtes à chaque fois qu'un nouveau sommet est introduit.

Ce réseau de référence R comporte donc 99 sommets et 168 arêtes. Le graphe de référence R est représenté dans la figure 3.6 et la distribution des valeurs des attributs attachés aux sommets de chaque classe dans la figure 3.5.

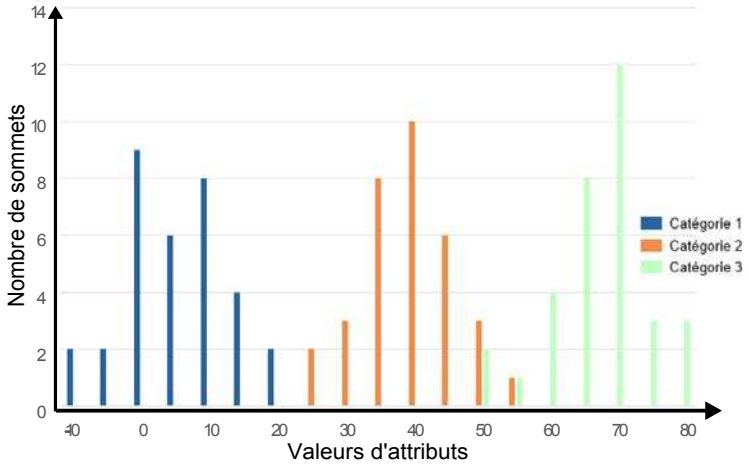



FIGURE 3.5 – Distribution des attributs des sommets du réseau R (écart-type de 7)

La table 3.1 montre la répartition des arêtes entre les classes dans ce graphe R.

L'évaluation de la méthode ToTeM sera effectuée sur le graphe de référence R et les résultats seront comparés à ceux produits par la méthode de Louvain et par les K-means en fixant le paramètre à 3.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 55 | | |
| Catégorie 2 | 2 | 53 | |
| Catégorie 3 | 1 | 7 | 50 |

TABLE 3.1 – Répartition des extrémités des liens du graphe de référence R

### Résultats sur le réseau de référence

Les résultats de l'application des trois méthodes sont présentés dans la table 3.2.

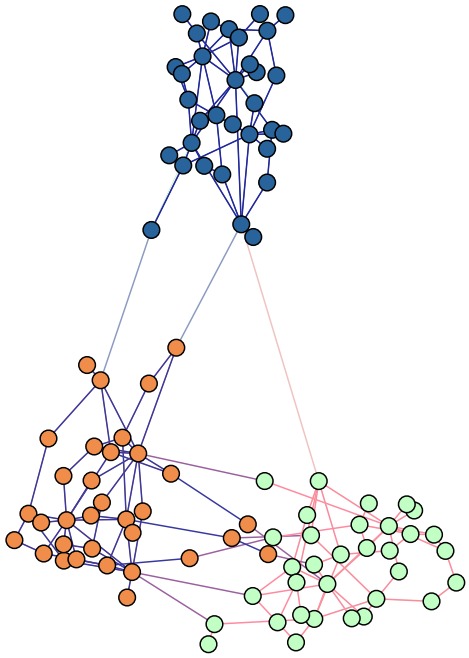


FIGURE 3.6 – Catégories de la vérité terrain du jeu de données synthétique de référence

On constate que ToTeM surpasse la méthode de Louvain, mais ne fait pas mieux que les K-means qui parviennent à un très bon résultat en terme de taux de bien classés. ToTeM suit de près les K-means, pour lesquels le nombre de classes est fixé par avance. La méthode de Louvain provoque néanmoins la scission de l'une des classes. Sur ce jeu, les taux de biens classés sont élevés pour les 3 méthodes.

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 3 | 4 | 3 |
| Taux de bien classés | 0,960 | 0,838 | 0,970 |
| NMI | 0,86 | 0,78 | 0,91 |
| $Q_{NG}$ | 0,61 | 0,62 | 0,60 |
| $I_{inter}$ | 658 | 651 | 659 |
| silhouette-Liens | 0,47 | 0,48 | 0,46 |
| silhouette-Attributs | 0,80 | 0,78 | 0,80 |

TABLE 3.2 – Résultats sur le réseau R

### 3.7.2 Dégradation de l'information relationnelle (réseaux R.1.1 et R.1.2)

On veut d'abord savoir si la méthode parvient à maintenir ses résultats dans la situation où l'information relationnelle est dégradée. Un nombre important d'arêtes intraclasses aide à la fois la méthode de Louvain et ToTeM à retrouver les communautés. On veut réduire le nombre d'arêtes intraclasses et introduire à la place des arêtes interclasses, qui compliquent la tâche de classification.

Pour ce faire, on introduit l'algorithme 5 dans lequel un paramètre détermine la proportion d'arêtes intraclasses à remplacer par une arête interclasses.

**Algorithme 5** : Algorithme de dégradation de l'information relationnelle

**Entrées** : le graphe $G$, la proportion d'arêtes intraclasses à transformer en arêtes interclasses $ratio_{arêtes}$

**Sorties** : le graphe $G$ modifié avec une information relationnelle dégradée

1 on fait la liste des arêtes intraclasses (celles dont les deux extrémités sont de la même classe de la partition vérité terrain);

2 **pour tous les** *$i$ de 0 à $ratio_{arêtes} \times nb\_arêtes$* **faire**

3 on choisi une arête de la liste;

4 on choisi une de ses extrémités;

5 on choisi un sommet $v'$ au hasard dans le graphe tel qu'il n'est pas issu de la même classe ;

6 on remplace le sommet incident de l'extrémité choisie par $v'$;

On propose de mesurer les résultats sur les graphes R.1.1 et R.1.2, dégradés respectivement avec des valeurs du paramètre de 0,25 et 0,5.

On ne retient la dégradation appliquée au graphe de référence que si le nouveau graphe est connexe. La répartition des liens entre les classes, pour un facteur de dégradation de 0,25, est présentée dans le tableau 3.3. Les résultats sont présentés dans le tableau 3.4.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 43 | | |
| Catégorie 2 | 12 | 39 | |
| Catégorie 3 | 18 | 18 | 35 |

TABLE 3.3 – Répartition des extrémités des liens du graphe R.1.1

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 30 | 9 | 3 |
| Taux de bien classés | 0,18 | 0,24 | 0,97 |
| NMI | 0,49 | 0,22 | 0,91 |
| $Q_{NG}$ | 0,40 | 0,54 | 0,60 |
| $I_{inter}$ | 654 | 175 | 659 |
| silhouette-Liens | 0,67 | 0,40 | 0,47 |
| silhouette-Attributs | 0,78 | 0,22 | 0,80 |

TABLE 3.4 – Résultats sur le graphe R.1.1

La répartition des liens entre les classes pour un facteur de dégradation de 0,5 est présentée dans le tableau 3.5. Les résultats sont présentés dans le tableau 3.6.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 30 | | |
| Catégorie 2 | 28 | 24 | |
| Catégorie 3 | 25 | 32 | 24 |

TABLE 3.5 – Répartition des extrémités des liens du graphe R.1.2

La dégradation de l'information relationnelle provoque une augmentation du nombre de classes retournées par ToTeM. La NMI de ToTeM se montre néanmoins sensiblement supérieure à celle de la méthode de Louvain pour les deux valeurs de paramètres testées. Cela traduit le fait que les classes trouvées par ToTeM recoupent mieux la partition de la vérité terrain que les classes produites par la méthode de Louvain notamment lorsque la dégradation est forte.

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 36 | 10 | 3 |
| Taux de bien classés | 0,141 | 0,141 | 0,970 |
| NMI | 0,38 | 0,12 | 0,91 |
| $Q_{NG}$ | 0,42 | 0,52 | 0,60 |
| $I_{inter}$ | 544 | 169 | 659 |
| silhouette-Liens | 0,64 | 0,41 | 0,46 |
| silhouette-Attributs | 0,62 | 0,20 | 0,80 |

TABLE 3.6 – Résultats sur le graphe R.1.2

### 3.7.3 Dégradation des attributs (réseaux R.2.1 et R.2.2)

On teste ensuite la robustesse de la méthode sur des réseaux d'information où les vecteurs d'attributs sont moins caractéristiques des classes de la vérité terrain. Pour cela, on se propose d'augmenter l'écart-type des attributs générés pour chacun des sommets. Le but est d'obtenir des distributions des valeurs d'attributs qui se chevauchent de plus en plus. Nous avons évalué les conséquences d'une augmentation de l'écart-type à des valeurs de 10 et 12 correspondant respectivement aux réseaux d'information notés R.2.1 et R.2.2. Les figures 3.7 et 3.8 présentent respectivement les distributions des attributs par classe pour ces réseaux.

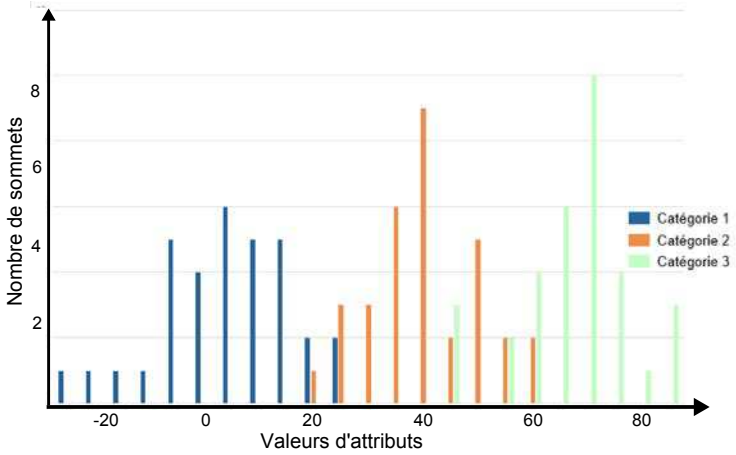



FIGURE 3.7 – Distribution des attributs par classe sur R.2.1 (écart-type de 10)

Pour un écart-type de 10, les résultats sont à consulter dans le tableau 3.7 et le taux de biens classés s'élève à 95% tandis que pour un écart-type de 12, ils sont présentés dans le tableau 3.8 et le taux de biens classés s'élève à 19% avec la méthode ToTeM.

On remarque que les résultats restent bons pour un écart-type de 10 et que ToTeM

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 3 | 4 | 3 |
| Taux de bien classés | 0,949 | 0,838 | 0,919 |
| NMI | 0,82 | 0,78 | 0,75 |
| $Q_{NG}$ | 0,61 | 0,62 | 0,56 |
| $I_{inter}$ | 703 | 690 | 712 |
| silhouette-Liens | 0,46 | 0,48 | 0,44 |
| silhouette-Attributs | 0,74 | 0,71 | 0,76 |

TABLE 3.7 – Résultats sur le graphe R.2.1

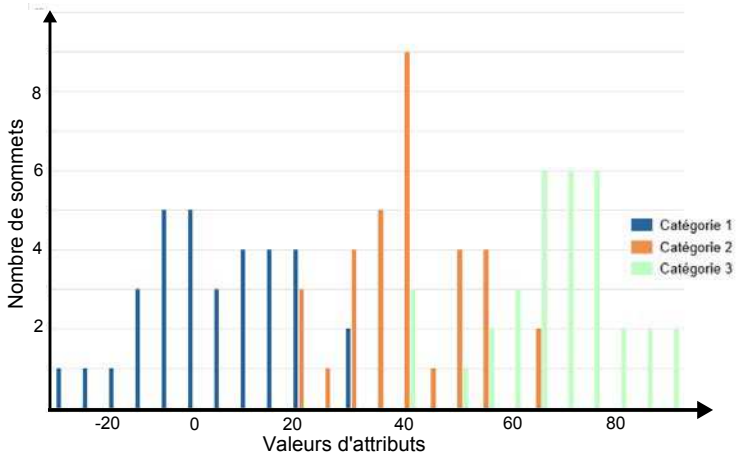



FIGURE 3.8 – Distribution des attributs par classe sur le réseau R.2.2 (écart-type de 12)

supporte mieux la dégradation d'information que les K-means d'après la NMI. Avec un écart-type de 12, les résultats de ToTeM sont très dégradés. Ceci est une conséquence du grand nombre de classes produites (26). Ceci affecte directement le taux de biens classés et, dans une moindre mesure, les autres critères tels que la NMI qui reste néanmoins proche de celle de la partition fournie par les K-means pour laquelle le nombre de classes attendues a été donné par l'utilisateur.

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 26 | 4 | 3 |
| Taux de bien classés | 0,192 | 0,838 | 0,828 |
| NMI | 0,57 | 0,78 | 0,59 |
| $Q_{NG}$ | 0,45 | 0,62 | 0,43 |
| $I_{inter}$ | 820 | 718 | 759 |
| silhouette-Liens | 0,69 | 0,48 | 0,35 |
| silhouette-Attributs | 0,81 | 0,67 | 0,74 |

TABLE 3.8 – Résultats sur le graphe R.2.2

### 3.7.4 Augmentation de la taille du réseau (réseaux R.3.1 et R.3.2)

On cherche à déterminer l'influence du nombre de sommets sur les résultats de la classification. On propose d'ajouter des sommets dans le graphe de référence et de mesurer les résultats sur de nouveaux graphes de 999 et 5 001 sommets.

Pour un réseau R.3.1 à 999 sommets, la répartition des liens entre les classes est présentée dans le tableau 3.9. Les résultats sont présentés dans le tableau 3.10. Le taux de bien classés est de 96,2% pour ToTeM.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 599 | | |
| Catégorie 2 | 36 | 573 | |
| Catégorie 3 | 5 | 56 | 570 |

TABLE 3.9 – Répartition des extrémités des liens du graphe R.3.1

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 3 | 12 | 3 |
| Taux de bien classés | 0,962 | 0,502 | 0,972 |
| NMI | 0,85 | 0,60 | 0,88 |
| $Q_{NG}$ | 0,63 | 0,68 | 0,62 |
| $I_{inter}$ | 604 | 601 | 608 |
| silhouette-Liens | 0,30 | 0,32 | 0,30 |
| silhouette-Attributs | 0,80 | 0,80 | 0,81 |

TABLE 3.10 – Résultats sur le graphe R.3.1

Pour un réseau R.3.2 à 5001 sommets, la répartition des liens entre les classes est présentée dans le tableau 3.11. Les résultats sont présentés dans le tableau 3.12. Le taux de bien classés est de 0,45%. Si ToTeM se comporte bien pour une petite augmentation du nombre de sommets, une augmentation plus forte provoque la multipli-

cation des classes trouvées par ToTeM, et une dégradation importante des résultats. Le nombre de classes découvertes par la méthode de Louvain reste, lui, raisonnable. Les K-means ne sont pas affecté par cette transformation du réseau.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 3033 | | |
| Catégorie 2 | 214 | 2934 | |
| Catégorie 3 | 25 | 244 | 2990 |

TABLE 3.11 – Répartition des extrémités des liens du graphe R.3.2

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 1518 | 10 | 3 |
| Taux de bien classés | 0,005 | 0,400 | 0,976 |
| NMI | 0,38 | 0,58 | 0,89 |
| $Q_{NG}$ | 0,38 | 0,70 | 0,63 |
| $I_{inter}$ | 640 | 588 | 598 |
| silhouette-Liens | 0,79 | 0,24 | 0,24 |
| silhouette-Attributs | 0,97 | 0,81 | 0,81 |

TABLE 3.12 – Résultats sur le graphe R.3.2

### 3.7.5 Augmentation du nombre d'arêtes (réseaux R.4.1 et R.4.2)

On cherche ici à déterminer si la densité des arêtes dans le graphe a une influence sur le résultat. Pour cela, on augmente le nombre d'arêtes ajoutées dans le graphe lors de l'introduction de chaque nouveau sommet, à l'image de ce qui se fait dans le modèle d'Albert et Barabási (Albert et Barabási, 2002).

Le jeu de référence R utilise une valeur de 2 arêtes par sommet introduit. On propose de tester les résultats avec des valeurs de 5 et 10 arêtes.

On considère le réseau R.4.1 où 5 arêtes au maximum sont introduites à chaque insertion d'un sommet dans le réseau. La répartition des liens entre les classes est présentée dans le tableau 3.13 et les résultats dans le tableau 3.14.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 95 | | |
| Catégorie 2 | 20 | 92 | |
| Catégorie 3 | 2 | 16 | 90 |

TABLE 3.13 – Répartition des extrémités des liens du graphe R.4.1

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 3 | 3 | 3 |
| Taux de bien classés | 0,949 | 0,960 | 0,970 |
| NMI | 0,81 | 0,85 | 0,91 |
| $Q_{NG}$ | 0,57 | 0,56 | 0,60 |
| $I_{inter}$ | 581 | 581 | 659 |
| silhouette-Liens | 0,39 | 0,38 | 0,46 |
| silhouette-Attributs | 0,79 | 0,79 | 0,80 |

TABLE 3.14 – Résultats sur le graphe R.4.1

Pour le graphe le plus dense R.4.2 où un maximum de 10 arêtes sont introduites pour chaque ajout d'un sommet dans le réseau, la répartition des liens entre les classes est présentée dans le tableau 3.15. Les résultats sont présentés dans le tableau 3.16.

| | Catégorie 1 | Catégorie 2 | Catégorie 3 |
|---|---|---|---|
| Catégorie 1 | 147 | | |
| Catégorie 2 | 30 | 142 | |
| Catégorie 3 | 4 | 27 | 158 |

TABLE 3.15 – Répartition des extrémités des liens du graphe R.4.2

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| Nombre de classes | 3 | 3 | 3 |
| Taux de bien classés | 0,979 | 0,969 | 0,969 |
| NMI | 0,92 | 0,88 | 0,91 |
| $Q_{NG}$ | 0,55 | 0,55 | 0,60 |
| $I_{inter}$ | 587 | 586 | 659 |
| silhouette-Liens | 0,39 | 0,39 | 0,46 |
| silhouette-Attributs | 0,81 | 0,80 | 0,80 |

TABLE 3.16 – Résultats sur le graphe R.4.2

On remarque que l'augmentation du nombre d'arêtes par sommet a plutôt un effet positif sur les résultats de ToTeM, en particulier au niveau de la NMI. Si les K-means ne sont pas affectés par cette modification, on ne s'étonne pas du fait que la méthode de Louvain bénéficie elle aussi d'une amélioration lors de la détection des communautés.

### 3.7.6 Conclusion sur l'évaluation après dégradation de l'information

Les résultats en fonction de la NMI sont récapitulés dans le tableau 3.17. On précise que les valeurs entre parenthèses indiquent les scores qui n'ont pas été affectés

par la dégradation appliquée. Les valeurs en gras indiquent, parmi les méthodes affectées, laquelle a obtenu le meilleur score de NMI. Les résultats en terme de taux de bien classés ne sont en effet pas raisonnablement calculables dans toutes les configurations.

| | ToTeM | Louvain | K-means |
|---|---|---|---|
| **Graphe de référence** | | | |
| R | 0,86 | 0,78 | **0,91** |
| **Dégradation de l'information relationnelle** | | | |
| R.1.1 | **0,49** | 0,22 | (0,91) |
| R.1.2 | **0,38** | 0,12 | (0,91) |
| **Dégradation des attributs** | | | |
| R.2.1 | **0,82** | (0,78) | 0,75 |
| R.2.2 | 0,57 | (0,78) | **0,59** |
| **Augmentation de la taille du réseau** | | | |
| R.3.1 | 0,85 | 0,60 | **0,88** |
| R.3.2 | 0,38 | 0,58 | **0,89** |
| **Augmentation du nombre d'arêtes** | | | |
| R.4.1 | 0,81 | **0,85** | (0,91) |
| R.4.2 | **0,92** | 0,88 | (0,91) |

TABLE 3.17 – Bilan de l'expérimentation, selon le score de NMI entre la partition terrain et la partition réelle)

ToTeM se comporte bien face à une augmentation importante de la densité des liens. De plus, on constate que ToTeM montre une forte robustesse par rapport à Louvain face à la dégradation de l'information relationnelle et par rapport aux K-means face à une dégradation raisonnable de l'information apportée par les attributs. Ceci démontre l'intérêt de la combinaison des deux informations. Par contre, l'augmentation de la taille du graphe provoque une augmentation très rapide du nombre de classes produites, et une forte dégradation des résultats.

### 3.7.7 Dégradation simultanée de l'information relationnelle et des valeurs des attributs sur un réseau de taille supérieure

On veut tester la performance de la méthode ToTeM dans le cas où on dégrade à la fois les relations et les valeurs des attributs. Pour cela on utilise un nouveau graphe de référence. Les attributs restent centrés sur $\mu_1 = 10$, $\mu_2 = 40$ et $\mu_3 = 70$. Les jeux sont générés de façon à comporter chacun 3 classes de 999 sommets. Le nombre de liens par lesquels un sommet est attaché à des sommets introduits précédemment est cette fois-ci au maximum de 4, afin de ne pas affecter la densité de façon trop significative

par rapport aux expérimentations sur le réseau R. Il s'agit donc d'un graphe à la fois plus grand et plus dense que celui utilisé dans la section 3.7.1.

Les attributs sont dégradés selon les modalités de l'algorithme 5 en considérant des valeurs d'écart-type $\sigma$ variant de 7 à 50. Le taux d'arêtes dégradées variera de 0 à 0,3. Les résultats sont présentés dans le tableau 3.18.

| Ratio d'arêtes dégradées | $\sigma$ | ToTeM NMI | ToTeM Nb de classes | ToTeM Bien classés | Louvain NMI | Louvain Nb de classes | Louvain Bien classés | K-means NMI | K-means Nb de classes | K-means Bien classés |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | | | | | | | | | | |
| | 7 | 0,89 | 3 | 0,97 | 0,76 | 5 | 0,85 | **0,90** | (3) | 0,98 |
| | 10 | **0,92** | 3 | 0,98 | 0,76 | 5 | 0,85 | 0,73 | (3) | 0,92 |
| | 20 | **0,84** | 3 | 0,96 | 0,76 | 5 | 0,85 | 0,34 | (3) | 0,69 |
| | 30 | 0,70 | 5 | 0,89 | **0,76** | 5 | 0,85 | 0,19 | (3) | 0,57 |
| | 50 | 0,47 | 4 | 0,79 | **0,76** | 5 | 0,85 | 0,09 | (3) | 0,47 |
| 0,1 | | | | | | | | | | |
| | 7 | 0,82 | 3 | 0,94 | 0,60 | 6 | 0,63 | **0,90** | (3) | 0,98 |
| | 10 | **0,87** | 3 | 0,97 | 0,60 | 6 | 0,63 | 0,73 | (3) | 0,92 |
| | 20 | **0,60** | 3 | 0,87 | 0,60 | 6 | 0,63 | 0,34 | (3) | 0,69 |
| | 30 | 0,40 | 5 | 0,52 | **0,60** | 6 | 0,63 | 0,19 | (3) | 0,57 |
| | 50 | 0,22 | 8 | 0,42 | **0,60** | 6 | 0,63 | 0,09 | (3) | 0,47 |
| 0,2 | | | | | | | | | | |
| | 7 | 0,74 | 3 | 0,92 | 0,33 | 8 | 0,42 | **0,90** | (3) | 0,98 |
| | 10 | **0,82** | 5 | 0,96 | 0,33 | 8 | 0,42 | 0,73 | (3) | 0,92 |
| | 20 | **0,52** | 3 | 0,83 | 0,33 | 8 | 0,42 | 0,34 | (3) | 0,69 |
| | 30 | 0,27 | 4 | 0,68 | **0,33** | 8 | 0,42 | 0,19 | (3) | 0,57 |
| | 50 | 0,18 | 8 | 0,60 | **0,33** | 8 | 0,42 | 0,09 | (3) | 0,47 |
| 0,3 | | | | | | | | | | |
| | 7 | 0,78 | 3 | 0,94 | 0,15 | 14 | 0,16 | **0,90** | (3) | 0,98 |
| | 10 | **0,74** | 3 | 0,93 | 0,15 | 14 | 0,16 | 0,73 | (3) | 0,92 |
| | 20 | **0,49** | 3 | 0,80 | 0,15 | 14 | 0,16 | 0,34 | (3) | 0,69 |
| | 30 | **0,27** | 3 | 0,66 | 0,15 | 14 | 0,16 | 0,19 | (3) | 0,57 |
| | 50 | 0,12 | 10 | 0,54 | **0,15** | 14 | 0,16 | 0,09 | (3) | 0,47 |

TABLE 3.18 – Dégradation simultanée des relations et des attributs

On voit que tant que la dégradation appliquée sur les attributs est faible, l'application des K-means en trois classes est la plus efficace (écart-type à 7), mais nécessite de connaître le nombre de classes. On constate aussi que pour une dégradation moyenne des valeurs d'attributs ($\sigma = 10$ ou $\sigma = 20$), les K-means et la méthode de Louvain sont dépassés par ToTeM en terme de NMI. En cas de forte dégradation des attributs, la méthode de Louvain parvient évidemment à reprendre la tête car elle n'est pas affectée par la baisse de qualité des attributs. Par ailleurs, lorsque le nombre de liens dégradés est important, ToTeM va rester plus longtemps compétitif face à Louvain. De plus, le nombre de classes découvertes par ToTeM est généralement plus faible que celui des classes découvertes par Louvain, notamment en cas de forte dégradation des liens (ratio = 0,3).

### 3.7.8 Conclusion sur l'évaluation sur des réseaux artificiels

Nous avons présenté des expérimentations testant l'apport effectif de ToTeM dans des contextes de classification de réseaux d'information.

La première expérimentation a montré que la méthode était robuste face à une dégradation de l'information relationnelle. En revanche, avec l'augmentation de la taille du graphe, on a vu que le nombre de classes produites par ToTeM a tendance à augmenter. La méthode de Louvain est également touchée par ce phénomène. On rappelle que dans le comparatif, la méthode des K-means a bénéficié de la connaissance du nombre de classes à produire, alors que ce n'était pas le cas pour la méthode de Louvain et ToTeM.

La deuxième expérimentation a montré que lorsque le réseau est dégradé dans une mesure raisonnable à la fois au niveau des arêtes et au niveau des attributs des sommets, notre méthode donne de bons résultats qui confirment tout l'intérêt de la prise en compte simultanée des deux informations. On constate également dans cette seconde expérimentation menée sur un graphe plus dense que précédemment que le nombre des communautés retournées parvient cette fois à être plus faible qu'avec la méthode de Louvain.

Ces expérimentations effectuées sur des réseaux artificiels montrent que ToTeM apporte un bénéfice dans la classification combinée de réseaux d'information, mais le problème du passage à l'échelle reste posé.

## 3.8 Évaluation de ToTeM sur le jeu des 4 sessions

Dans cette section, nous évaluons la méthode ToTeM sur un réseau bibliographique. Cette évaluation sera effectuée sur le jeu de données des 4 sessions construit

par nos soins dans le but de mesurer l'apport de la prise en compte de la combinaison des sources de données (Combe et al., 2012a,b). Ce jeu est est décrit dans la section 1.4.4. Rappelons que ce réseau comporte 99 auteurs et 2 623 relations de coparticipation à des conférences.

### 3.8.1 Hypothèses et scénarios

L'évaluation sur le jeu de données des sessions cherche à étudier le comportement des algorithmes par rapport à trois scénarios :

- Quelle est la thématique d'un auteur ?
- Dans quelle conférence a-t-il publié ?
- Dans quelle session a-t-il publié ?

On suppose que selon le scénario choisi, la solution pourra être plus ou moins facilement trouvée selon que l'on exploite l'information textuelle, relationnelle ou les deux.

Nous définissons ci-dessous les différents scénarios de classification. Nous considérons quatre sous-ensembles $A$ (Bioinformatique), $B$ (Robotique-SAC), $C$ (Robotique-IJCAI) et $D$ (Contraintes), rassemblant les auteurs publiant dans les quatre sessions considérées (voir tableau 3.19).

| Session et conférence de rattachement | Effectif |
|---|---|
| A Bioinformatique (SAC) | 24 |
| B Robotique (SAC) | 16 |
| C Robotique (IJCAI) | 38 |
| D Contraintes (IJCAI) | 21 |
| Effectif du jeu de données | 99 |

TABLE 3.19 – Effectif de chaque session

- **Scénario 1 : Identification du domaine de recherche : 3 catégories ($P_T$)**
  L'hypothèse qui fonde cette première expérience est que l'information textuelle devrait permettre de retrouver les trois domaines de recherche : robotique, bioinformatique et programmation par contraintes ; ceci revient à prendre comme vérité terrain la partition en trois groupes contenant les auteurs rattachés à chaque thématique de recherche : $P_T = \{A, B \cup C, D\}$.
- **Scénario 2 : Identification de la conférence : 2 catégories ($P_S$)**
  La prise en compte des données relationnelles seules devrait permettre d'identifier deux groupes correspondant aux auteurs qui participent à chaque conférence et qui correspondent à la vérité terrain de la partition $P_S = \{A \cup B, C \cup D\}$.

– **Scénario 3 : identification de la session : 4 catégories ($P_{TS}$)**
Enfin, si nous voulons identifier les auteurs rattachés à chaque session, les informations textuelles et relationnelles doivent alors être utilisées. Dans ce cas, la partition associée à la vérité terrain est $P_{TS} = \{A, B, C, D\}$.

Afin d'évaluer les algorithmes, nous serons donc en mesure pour chaque scénario de comparer le résultat produit à la partition attendue ($P_S$, $P_T$ ou $P_{TS}$) et de calculer un pourcentage de bien classés. De plus, les affectations des auteurs sont présentées sous la forme de matrices de coïncidences. Ces matrices permettent à la fois de savoir quelles sont les classes réelles qui sont les plus difficiles à retrouver, mais également quelle est la répartition des auteurs des différentes sessions dans les partitions inférées par la méthode.

Afin de comparer ToTeM à des méthodes exploitant à la fois des données relationnelles et d'attributs, nous considérons trois méthodes de référence à confronter à notre algorithme. Ces méthodes, dénommées $TS_1$, $TS_2$ et $TS_3$, sont décrites ci-après.

### 3.8.2 Méthodes comparées

Les méthodes dont les résultats, selon les différentes hypothèses, ont été comparés sont décrites dans cette section. Deux méthodes de référence mesurent les résultats obtenus en ne prenant en compte qu'un seul type de données. De plus, trois méthodes de détection de communautés dans des réseaux d'information $TS_1$, $TS_2$ et $TS_3$, simples à mettre en œuvre, sont considérées dans notre expérimentation. Ces méthodes ont été introduites dans (Combe et al., 2012b).

#### 3.8.2.1 Méthode relationnelle de référence : $S$

La méthode Louvain, de Blondel *et al.* qui exploite seulement les données structurelles (le graphe $G = (V, E)$) sera utilisé comme méthode relationnelle de référence (Blondel et al., 2008). Elle sera appliquée sur le graphe des auteurs muni de la relation de coparticipation à des conférences. Elle sera notée $S$ dans les sections suivantes.

#### 3.8.2.2 Méthode textuelle de référence : $T$

La classification en fonction des attributs textuels ne prend en compte que les documents $\{d_i, \forall v_i \in V\}$. Elle a été réalisée avec la distance euclidienne ainsi qu'avec la distance du cosinus calculée sur la description *tf-idf*, et avec la méthode des K-means bissectif, puis avec la distance du cosinus et l'algorithme du lien moyen. Comme le lien moyen donne de meilleurs résultats, c'est la seule méthode qui sera présentée ici comme référence pour nos expérimentations.

#### 3.8.2.3 $TS_1$ : Détection de communautés relationnelles sur le graphe valué à l'aide de distances entre attributs

Cette première méthode combinante notée $TS_1$ s'apparente à celle décrite par Steinhaeuser (Steinhaeuser et Chawla, 2008). Les attributs sont utilisés pour obtenir un graphe valué. Nous définissons une distance portant sur les attributs $dis_T$, par exemple la distance euclidienne ou la dissimilarité du cosinus, bien adaptée aux attributs textuels. Nous avons retenu la distance du cosinus appliquée sur les vecteurs *tf-idf*. La valeur $dis_T(v, v')$ est associée à chaque arête $(v, v')$ de $E$. Puis, une méthode de détection de communautés dans un graphe, compatible avec les graphes valués, est utilisée pour partitionner l'ensemble des sommets $V$, par exemple un algorithme qui optimise une fonction de qualité comme l'algorithme de Kernighan-Lin ou ceux basés sur la modularité (Kernighan et Lin, 1970). Dans les expérimentations, nous utilisons la méthode de Louvain sur le graphe valué. La figure 3.9 montre le déroulement de cette méthode.

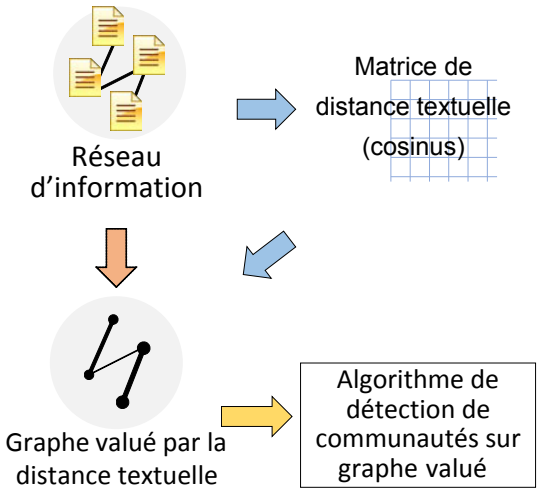



FIGURE 3.9 – Déroulement de la méthode $TS_1$

Ce traitement a pour avantage de permettre de limiter le calcul des distances aux paires de sommets reliées par les arêtes.

Cependant, un inconvénient de cette méthode est qu'il n'y a pas de prise en compte des distances par rapport aux attributs entre des sommets qui ne sont pas directement reliés dans le graphe.

#### 3.8.2.4 $TS_2$ : Classification automatique basée sur la matrice des distances géodésiques déduites du graphe valué à partir des attributs

Dans cette méthode, dite $TS_2$, les informations relationnelles sont utilisées pour définir une mesure de dissimilarité $dis_S(v, v')$ entre chaque paire de sommets $(v, v')$ dans le graphe. Dans la pratique, la longueur du plus court chemin entre $v$ et $v'$ peut être utilisée comme $dis_S(v, v')$, où le chemin le plus court entre $v$ et $v'$ est le chemin qui comporte le plus petit nombre d'arêtes. Dans le cas où les arêtes sont valuées, la longueur du chemin entre $v$ et $v'$ est la somme des valuations des arêtes du chemin et le plus court chemin entre deux sommets est celui pour lequel cette somme est minimale. Les longueurs des chemins minimaux définissent des dissimilarités et toute technique d'apprentissage non supervisée peut être appliquée sur la matrice de dissimilarités ainsi obtenue. La figure 3.10 montre le déroulement de cette méthode.

Dans nos expérimentations, nous avons utilisé la distance géodésique sur le graphe valué où à chaque arête on associe la distance du cosinus définie sur les attributs textuels (*tf-idf*) relatifs aux sommets de cette arête. On applique ensuite la classification hiérarchique avec plusieurs critères d'agrégation (les liens simple, moyen, complet, des centres de gravité) sans constater de différence dans le résultat de la classification.

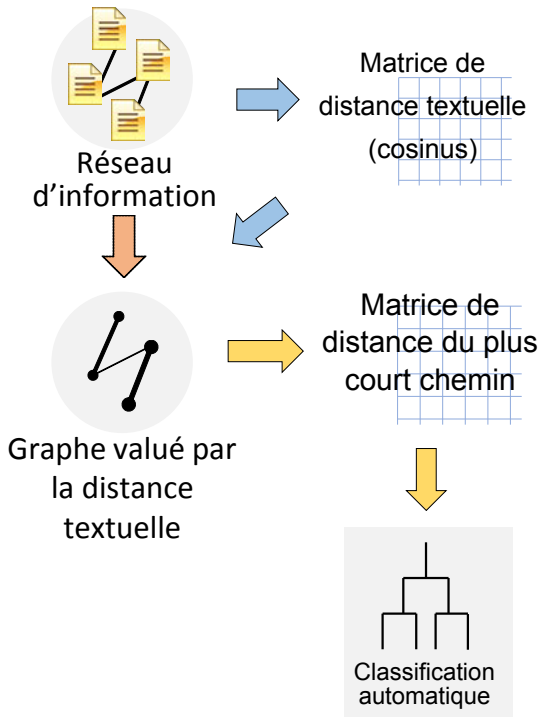



FIGURE 3.10 – Déroulement de la méthode $TS_2$

Une limite de la méthode $TS_2$ est que, comme précédemment, il n'y a pas de prise en compte des distances textuelles directes entre des sommets qui ne sont pas directement reliés dans le graphe.

#### 3.8.2.5 $TS_3$ : Combinaison linéaire de distances

Dans la troisième méthode, $TS_3$, une dissimilarité globale $dis_{TS}(v, v')$ entre deux sommets $v$ et $v'$ est définie comme une combinaison linéaire de deux mesures de dissimilarité correspondant respectivement à chaque type d'information :

$$dis_{TS}(v, v') = \alpha\, dis_T(v, v') + (1 - \alpha)\, dis_S(v, v') \tag{3.25}$$

où $dis_T(d_i, d_j)$ est une dissimilarité définie sur les attributs, $dis_S(v, v')$ est définie directement à partir du graphe et $\alpha$ est un paramètre compris entre $0$ et $1$.

Comme précédemment, la longueur d'un plus court chemin entre $v$ et $v'$ peut être utilisée pour $dis_S(v, v')$, et la distance euclidienne ou la distance du cosinus calculées sur les attributs pour $dis_T(d_v, d_{v'})$. Ensuite, la partition peut être construite soit avec un algorithme de partitionnement de graphe appliqué sur le graphe étendu et valué par la dissimilarité globale, soit par une technique non supervisée d'apprentissage utilisant la dissimilarité globale. La figure 3.11 montre le déroulement de cette méthode.

Ce scénario repose sur l'hypothèse que les deux dissimilarités se complètent ou se renforcent.

La combinaison linéaire est très facile à mettre en œuvre. Elle est facilement généralisable à un plus grand nombre de matrices et donc de distances. Cependant, un inconvénient de cette approche est qu'il n'est pas aisé de choisir la valeur du coefficient $\alpha$.

Dans nos expérimentations, nous avons utilisé la distance du cosinus entre les vecteurs d'attributs représentés selon le modèle *tf-idf* et la distance géodésique sur le graphe non valué pour exploiter l'information des relations, puis la classification hiérarchique avec le critère d'agrégation du lien moyen.

### 3.8.3 Résultats expérimentaux

#### 3.8.3.1 Identification du domaine de recherche (scénario 1)

Nous avons supposé que pour déterminer le domaine de recherche, seule l'information textuelle était utile. Cette section décrit donc les résultats obtenus à l'aide de la méthode textuelle de référence $T$.

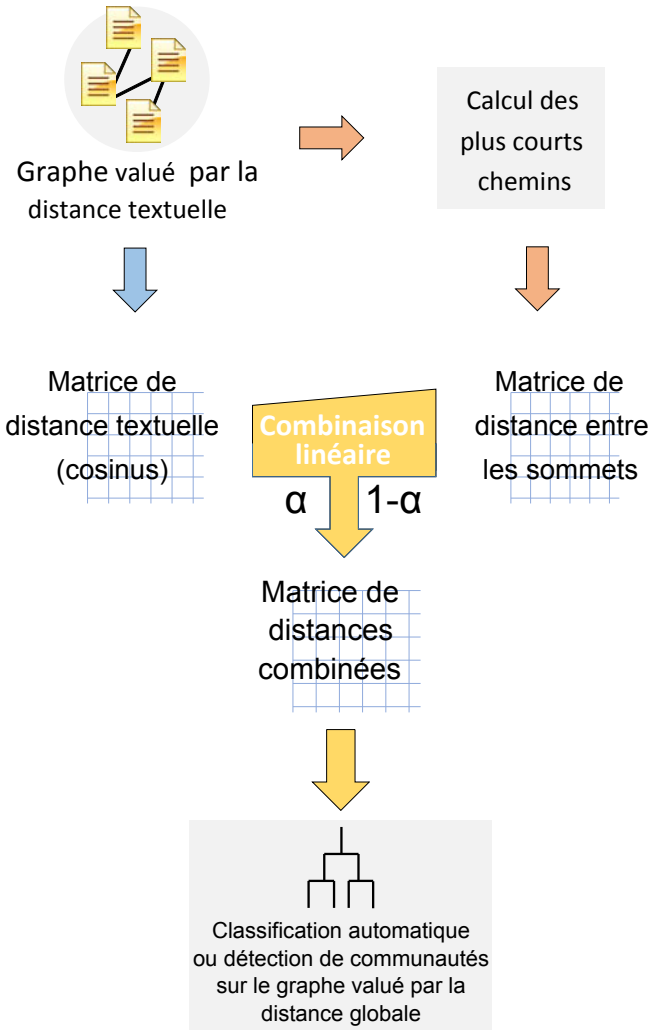



FIGURE 3.11 – Déroulement de la méthode $TS_3$

Le tableau 3.20 présente les résultats pour la partition en trois classes. Dans les matrices de coincidences, les nombres en gras correspondent aux effectifs de sommets qui sont utilisés pour calculer le taux de sommets bien classés. Ici le taux de bien classés est de 87% ($\frac{(11+16+38+21)}{99} \cdot 100$).

| Classe prédite → Session réelle ↓ | Classe 1 | Classe 2 | Classe 3 | Total |
|---|---|---|---|---|
| A - SAC Bioinformatique | **11** | | 13 | 24 |
| B - SAC Robotique | | | **16** | 16 |
| C - IJCAI Robotique | | | **38** | 38 |
| D - IJCAI Contraintes | | **21** | | 21 |
| Total | 11 | 21 | 67 | 99 |

TABLE 3.20 – Résultat de la méthode T en 3 classes

Le tableau 3.21 présente ensuite les résultats de la méthode $T$ pour la partition en quatre classes. Le taux d'auteurs bien classés est de 69%. C'est ce taux que l'on gardera pour la méthode textuelle de référence lorsque l'objectif sera de déterminer la session de chaque auteur, pour la comparaison avec les méthodes de combinaison.

| Classe prédite → Session réelle ↓ | 1 | 2 | 3 | 4 | Total |
|---|---|---|---|---|---|
| A - SAC Bioinformatique | **11** | | | 13 | 24 |
| B - SAC Robotique | | | **2** | 14 | 16 |
| C - IJCAI Robotique | | | 4 | **34** | 38 |
| D - IJCAI Contraintes | | **21** | | | 21 |
| Total | 11 | 21 | 6 | 61 | 99 |

TABLE 3.21 – Résultat de la méthode T en 4 classes

#### 3.8.3.2 Identification de la conférence (scénario 2)

Notre hypothèse était que les données relationnelles permettaient d'identifier la conférence à laquelle les auteurs ont participé.

Cette section décrit les résultats obtenus à l'aide de la méthode relationnelle de référence ($S$) à savoir la méthode de Louvain sur le graphe non valué de la relation de coparticipation à des conférences.

L'algorithme donne sans surprise une partition en 2 classes et un taux de bien classés de 100% pour le scénario de reconnaissance des conférences. Celle-ci est consultable dans le tableau 3.22.

| Communauté prédite → Session réelle ↓ | Classe 1 | Classe 2 | Total |
|---|---|---|---|
| A - SAC Bioinformatique | **24** | | 24 |
| B - SAC Robotique | 16 | | 16 |
| C - IJCAI Robotique | | **38** | 38 |
| D - IJCAI Contraintes | | 11 | 21 |
| Total | 40 | 59 | 99 |

TABLE 3.22 – Résultats de la méthode relationnelle de référence

Il n'est pas possible de contraindre la méthode de Louvain à fournir 4 classes sur ce problème. Par conséquent le taux de bien classés pour la reconnaissance des sessions n'est que de $63\%$.

#### 3.8.3.3 Identification de la session (scénario 3)

Notre hypothèse est que pour identifier la session d'un auteur les informations relationnelles et textuelles sont nécessaires. Nous allons le vérifier en comparant les résultats produits par les méthodes exploitant les deux types de données (ToTeM, $TS_1$, $TS_2$, $TS_3$) aux méthodes textuelle et relationnelle de référence $T$ et $S$. De plus, nous pourrons évaluer la performance de ToTeM par rapport aux autres approches.

Le tableau 3.23 résume les résultats obtenus avec les différentes méthodes. De plus, les matrices de coïncidence sont présentées dans la table 3.24. Le taux de bien classés s'élève à 63% pour la méthode ToTeM comme pour Louvain.

On constate que $TS_1$ a fourni une classe supplémentaire. Malgré cela, c'est la méthode qui donne le meilleur résultat en terme de taux de bien classés. La méthode $TS_2$ obtient aussi un bon résultat. La méthode $TS_3$ voit son résultat moins marqué vis-à-vis des deux conférences, elle identifie cependant parfaitement une catégorie.

On notera entre autres que la méthode de Louvain ne permet pas de déterminer le nombre de communautés à obtenir dans le cas du scénario 3 visant à identifier les sessions.

En effet ces méthodes trouvent bien les classes détectables par les relations, mais passent à côté de l'information détectable par la prise en compte du texte. Ceci s'explique par le fait que l'information relationnelle est très forte dans le sens où la relation de coparticipation à une même conférence produit un réseau très dense sur lequel les méthodes de détection de communautés sont très stables. Pour que la prise en compte de l'information des attributs soit profitable, il faudrait que l'information contenue dans les vecteurs textuels soit elle aussi très marquée et qu'elle discrimine fortement les auteurs.

| | Précision vis-à-vis de : | | |
|---|---|---|---|
| Modèle | $P_T$ | $P_S$ | $P_{TS}$ |
| $T$ | 87% | - | 69% |
| $S$ | - | 100% | 63% |
| $TS_1$ | - | - | 76% |
| $TS_2$ | - | - | 73% |
| $TS_3$ | - | - | 47-69% |
| ToTeM | - | - | 63% |

TABLE 3.23 – Synthèse des résultats : modèles $T$, $S$, $TS_1$, $TS_2$, $TS_3$ et ToTeM

### 3.8.4 Conclusion de l'expérimentation sur le jeu des quatre sessions

Tels que ceux-ci ont été présentés dans les sections précédentes, nous obtenons des résultats très différents selon la méthode employée pour la détection de communautés dans le réseau d'attributs, mais également selon le scénario et la partition de référence considérée.

Les attributs textuels permettent assez bien de retrouver les thèmes de recherche et les données structurelles de co-participation permettent de retrouver quasiment parfaitement les conférences.

Les articles publiés dans les différentes thématiques sont décrits par des vecteurs contenant principalement des termes distincts. Grâce à cela, une méthodologie basée sur l'usage du *tf-idf* et de la distance du cosinus fonctionne bien. Des méthodes différentes de génération des vecteurs du réseau peuvent donner des résultats différents. Le choix des mots vides peut à lui seul se révéler important.

Quand il est question de manipuler les données structurelles, nous avons utilisé l'information de co-participation pour construire le graphe. D'autres relations, telles que la co-écriture ou la citation pourraient être utilisées à ce stade.

Certaines communautés ne peuvent être déterminées qu'en associant l'information relationnelle et l'information textuelle. Dans notre cas, ces communautés sont les quatre sessions sélectionnées. Dans le but de retrouver cette partition, nous avons combiné ces deux informations, en proposant trois méthodes de combinaison différentes qui sont comparées à ToTeM. Les résultats montrent que, dans notre cas, la combinaison linéaire n'est pas en mesure d'améliorer les résultats. De plus, elle nécessite un paramètre de pondération relative des deux critères.

Les méthodes $TS_1$ et $TS_2$ donnent de meilleurs résultats que celle utilisant la combinaison linéaire des deux distances textuelle et relationnelle.

Quelque soit l'application et le jeu de données, la combinaison de deux types de

| Communauté prédite → Session réelle ↓ | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| A SAC'09 - Bioinformatique | | 13 | | **11** | |
| B SAC'09 - Robotique | | **11** | | | 5 |
| C IJCAI'09 - Robotique | | | **38** | | |
| D IJCAI'09 - Contraintes | **15** | | 6 | | |
| Total | 15 | 24 | 44 | 11 | 5 |

(a) $TS_1$

| Com. préd. → Session ↓ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A | | **24** | | |
| B | **4** | 11 | 1 | |
| C | | | 1 | **37** |
| D | | | **7** | 14 |
| Total | 4 | 35 | 9 | 51 |

(b) $TS_2$

| Communauté prédite → Session réelle ↓ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A SAC'09 - Bioinformatique | **11** | | | 13 |
| B SAC'09 - Robotique | | | **2** | 14 |
| C IJCAI'09 - Robotique | | | 4 | **34** |
| D IJCAI'09 - Contraintes | | **21** | | |
| Total | 11 | 21 | 6 | 61 |

(c) $TS_3$

| Communauté préd. → Session réelle ↓ | 1 | 2 |
|---|---|---|
| A | | **24** |
| B | | 16 |
| C | **38** | |
| D | 21 | |
| Total | 59 | 40 |

(d) $ToTeM$

TABLE 3.24 – Matrices de coïncidence pour les quatre méthodes de combinaison comparées

données nécessite de s'appuyer sur la donnée la plus robuste dans le but de ne pas amplifier les défauts.

Nous avons également montré que de bons résultats de classification peuvent être obtenus en utilisant des méthodes simples à mettre en œuvre, du moment que le scénario est adapté aux données et que les caractéristiques décrivant ce qu'est une bonne communauté soient bien déterminées. Malheureusement, ce dernier point est souvent le plus difficile.

## 3.9 Évaluation sur un autre réseau de grande taille : PubMed-Diabètes

Cette fois-ci nous évaluons notre proposition sur un second jeu de données, PubMed-Diabètes, issu d'une base de données bibliographiques médicales.

### 3.9.1 Présentation du jeu de données

Le jeu de données Pubmed-Diabètes comprend 19 717 publications scientifiques traitant du diabète (Sen et al., 2008). Celles-ci sont classées en trois catégories (1) "Diabetes Mellitus, Experimental", (2) "Diabetes Mellitus Type 1", (3) "Diabetes Melli-

rat common use examin pathogenesi retinopathi mous studi anim model metabol abnorm contribut develop investig mice 2 month compar obtain method induc 6 inject experiment normal diet 30 hyperglycemia level lipid oxid activ protein kinas c measur result increas retin stress 3 similar observ conclus play import role present p m r muscl control chang dure lower higher mass correl decreas determin concentr stimul period caus mark group evid fast type signific differ ratio suggest degre occur vivo respect dysfunct region high appear sever affect cardiovascular complic primari death patient clinic suscept cardiac tissu specif function defect possibl indic state onli bodi weight loss valu howev 4 condit durat 8 week onset data direct report provid addit evalu sensit heart object mean blood glucos strong hba 1c a1c variabl independ assess relat trial

FIGURE 3.12 – Extrait du vocabulaire de 500 mots retenu dans PubMed

tus Type 2". Les catégories 2 et 3 correspondent respectivement à ce qui se rapporte au diabète de type 1 et 2. La catégorie 1 traite elle des cas de diabète provoqués par des traitements ou des opérations. Le jeu de données comprend 44 338 liens de citation entre les articles. Chaque article est associé à un vecteur de mots pondéré par le *tf-idf* (voir section 1.4.4.2). Les vecteurs comprennent 500 mots uniques.

Les catégories sont issues du thésaurus MeSH (Medical Subject Headings [2]) de la *National Library of Medicine*, institut national américain supportant PubMed. Ce thésaurus propose une classification complète et hiérarchisée des thèmes de tous les articles de la base de données bibliographiques. Chaque document y est associé à plusieurs catégories (généralement 10 à 12). Les articles sont assignés à des catégories par des experts [3].

Les effectifs des trois catégories sont :

– 4 103 articles (20,81%) pour Diabetes Mellitus, Experimental ;
– 7 875 articles (39,94%) pour Diabetes Mellitus, Type 1 ;
– 7 739 articles (39,25%) pour Diabetes Mellitus, Type 2.

Il convient de remarquer que la définition des catégories repose sur le contenu des articles et donc plutôt sur les attributs que sur les données relationnelles.

On considère le graphe $G = (V, E)$ où $V$ est l'ensemble des articles et $E$ est l'ensemble des liens de citation entre les éléments de $V$. Malgré le caractère non symétrique de la relation de citation, nous exploiterons le graphe sans tenir compte du sens de la citation.

Les 500 mots de l'index sont présentés dans la figure 3.12. Les mots les plus courants ont été enlevés. Le vocabulaire est en grande partie relatif au domaine de la biologie. La racine du mot diabète n'est pas présente, car trop fréquente, mais on trouve en revanche *nondiabet*. Un exemple de résumé ainsi que le vecteur associé sont

2. http://www.nlm.nih.gov/mesh/
3. http://nnlm.gov/training/resources/meshtri.pdf

**Involvement of O2 radicals in 'autoimmune' diabetes.**
Spontaneous diabetes in the non-obese diabetic (NOD) mice is a CD4 T cell-dependent process. We have suggested that specific beta cell destruction results from free radical production at the site of islet inflammation ; oxygen radicals are produced by activated inflammatory cells. We reported here that in vivo treatment of spontaneously diabetic NOD mice with the enzyme superoxide dismutase (2000 U for seven injections) and catalase (40,000 U for seven injections) protects islet tissue from disease recurrence following transplantation into spontaneously diabetic mice. Similar results were obtained when animals were treated with either enzyme alone. This effect was dose-dependent and little protection was observed when the dose of enzyme was reduced four-fold. These results indicate that oxygen metabolites, specially superoxide and hydrogen peroxide, are directly involved in the pathogenesis of immunology mediated diabetes.

FIGURE 3.13 – Exemple de résumé

présentés par les figures 3.13 et 3.14.

Contrairement au jeu précédent, où les informations relationnelles et textuelles se voulaient complémentaires, le plus souvent dans la pratique, données relationnelles et vectorielles tendent à se recouvrir et à se compléter (là où l'une des deux informations viendrait à manquer), de part le phénomène d'homophilie. Il sera cette fois question de combiner des informations qui sont corrélées pour mieux s'approcher d'une vérité terrain qui serait plus difficile à trouver si on ne traitait qu'une seule dimension des données.

Dans cette expérimentation, les méthodes qui seront comparées sont ToTeM, Louvain et la méthode de référence $TS_1$ présentée dans la section 3.8.2.5, qui a produit les meilleurs résultats dans l'expérimentation précédente. L'évaluations consistera d'abord en la comparaison de la partition produite avec les trois catégories de la vérité terrain. Ensuite, dans la mesure où la vérité terrain de ce jeu de données ne satisfait pas au critère de connexité présenté dans la section 3.6.3, nous ferons subir une opération à la vérité terrain en trois classes pour isoler chaque groupe de sommets d'une catégorie donnée qui ne soit pas relié aux autres sommets de sa catégorie en ne passant que par d'autres sommets adjacents de la même catégorie. Nous comparerons alors le résultat avec cette partition que nous désignerons comme "connexifiée". Cette comparaison sera faite sur les critères d'ARI, qui mesure l'accord entre la classification des paires d'éléments, la NMI basée sur l'information mutuelle, l'AMI une version corrigée pour la chance de l'information mutuelle, ainsi que la V-mesure et de ses sous-indices, l'homogénéité qui nous permettra de savoir dans quelle mesure les

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| pathogenesi | 0,0505 | anim | 0,0393 | mice | 0,1163 | obtain | 0,0489 |
| inject | 0,0897 | activ | 0,0289 | result | 0,0362 | similar | 0,0319 |
| observ | 0,0292 | suggest | 0,0230 | vivo | 0,0529 | tissu | 0,0437 |
| specif | 0,0419 | indic | 0,0328 | direct | 0,0502 | report | 0,0361 |
| treatment | 0,0247 | effect | 0,0196 | product | 0,0446 | reduc | 0,0276 |
| cell | 0,0549 | process | 0,0508 | diseas | 0,0241 | treat | 0,0398 |
| dose | 0,0463 | follow | 0,0383 | mediat | 0,0535 | involv | 0,0424 |
| nonobes | 0,0540 | 40 | 0,0471 | enzym | 0,1616 | free | 0,0573 |
| islet | 0,0775 | beta | 0,0465 | cd4 | 0,0666 | spontan | 0,1743 |
| protect | 0,1017 | destruct | 0,0569 | t | 0,0469 | produc | 0,0508 |
| nod | 0,1050 | transplant | 0,0638 | | | | |

FIGURE 3.14 – Vecteur associé au résumé de la figure 3.13

classes produites contiennent bien des éléments de même catégorie, et la complétude, pour savoir si les classes contiennent bien tous les éléments des catégories.

### 3.9.2 Résultat sur la vérité terrain brute (en 3 classes)

Le tableau 3.26 ainsi que la figure 3.15 présentent les scores atteints par le partitionnement du jeu de données PubMed face aux 3 classes.

| | Louvain | K-means (k=3) | $TS_1$ | ToTeM |
|---|---|---|---|---|
| ARI | 0,11 | 0,19 | 0,11 | 0,0045 |
| NMI | 0,23 | 0,20 | 0,23 | 0,27 |
| AMI | 0,13 | 0,20 | 0,14 | 0,09 |
| V-Mesure | 0,20 | 0,20 | 0,20 | 0,18 |
| Homogénéité | 0,13 | 0,21 | 0,14 | 0,10 |
| Complétude | 0,39 | 0,20 | 0,38 | 0,69 |

TABLE 3.26 – Résultats par rapport à la vérité non connexe (3 classes)

ToTeM obtient de bons résultats sur les critères de NMI et de complétude. La méthode de Louvain et la méthode $TS_1$ obtiennent de bons résultats. Le fait que la méthode de Louvain procure un bon résultat montre que les données relationnelles sont très importantes vis-à-vis de la vérité terrain.

### 3.9.3 Résultats sur la vérité terrain "connexifiée" (en 2 644 classes)

Les méthodes Louvain et ToTeM privilégiant des classes connexes, nous avons également évalué les algorithmes par rapport à une vérité terrain où les catégories retrouvées sont connexes. Le tableau 3.27 et la figure 3.16 présentent les scores atteint

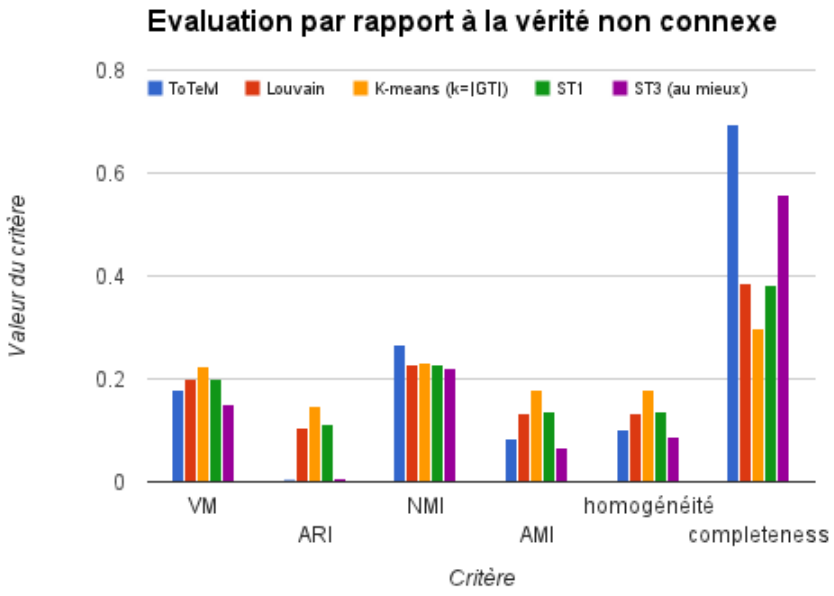



FIGURE 3.15 – Résultats sur les 3 catégories de la vérité terrain brute

par le partitionnement du jeu de données PubMed face aux 2 644 classes de la vérité terrain après connexification.

| | Louvain | K-means (k=2644) | $TS_1$ | ToTeM |
|---|---|---|---|---|
| ARI | 0,14 | 0,16 | 0,15 | 0,006 |
| NMI | 0,35 | 0,22 | 0,35 | 0,46 |
| AMI | 0,20 | 0,09 | 0,21 | 0,13 |
| V-Mesure | 0,35 | 0,20 | 0,35 | 0,40 |
| Homogénéité | 0,32 | 0,34 | 0,32 | 0,27 |
| Complétude | 0,39 | 0,14 | 0,38 | 0,79 |

TABLE 3.27 – Évaluation par rapport à la vérité connexe de PubMed-Diabètes

On constate que le score de V-mesure (dont le sous-score de complétude a fortement augmenté lors de la connexification) et celui de NMI sont cette fois-ci les plus forts pour ToTeM. Les K-means pâtissent assez logiquement de l'opération de connexification.

L'ARI est particulièrement faible dans ce contexte. On rappelle que l'ARI peut cependant atteindre des valeurs négatives de part sa conception. L'AMI est elle en faveur de la méthode de Louvain et de la méthode $TS_1$.

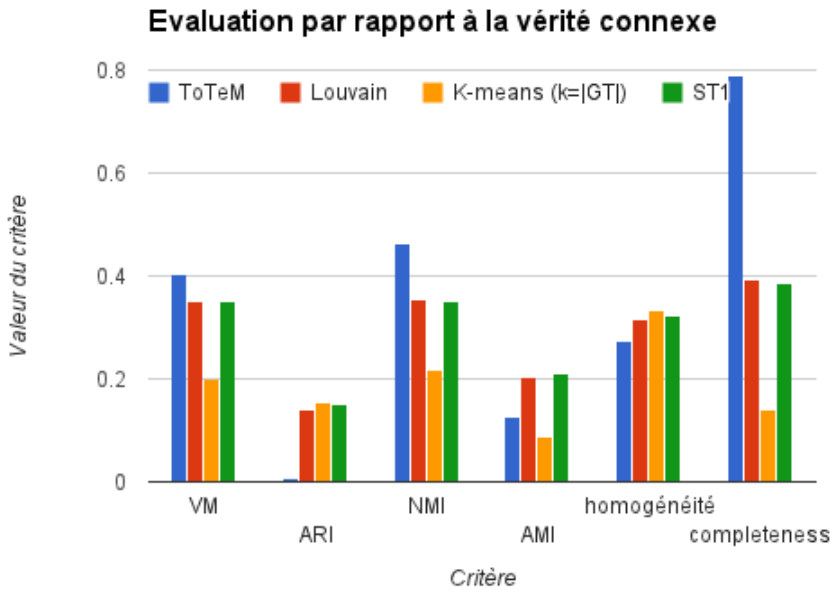



FIGURE 3.16 – Résultats sur les 2 644 classes de la vérité terrain après connexification

La méthode $TS_1$ obtient de façon générale de bons résultats proches de ceux de Louvain.

## 3.10 Conclusion

Dans ce chapitre, nous avons proposé une méthode de classification non supervisée de sommets dans un réseau d'information. La partition produite tient compte des relations entre les éléments et de la proximité de leurs vecteurs d'attributs. Cette méthode repose sur les critères reconnus que sont l'inertie interclasses et la modularité. L'inertie interclasses permet ainsi de caractériser la bonne séparation des éléments en fonction des valeurs prises par des attributs à valeurs réelles. La modularité, elle, permet de privilégier les partitions qui présentent des communautés à la fois denses en arêtes et bien séparées. Nous associons à ces deux mesures l'usage de l'heuristique de la méthode de Louvain. Cette dernière permet de limiter l'exploration de l'espace des partitions possibles. Aussi, elle prend en compte le problème de l'*échelle multiple* des communautés, et ceci aussi bien du point de vue du graphe que des attributs.

Nous proposons de recourir à un critère global tirant parti de la modularité de Newman et Girvan et de l'inertie interclasses. D'autres critères relationnels, comme la conductance et le nombre de triades, qui sont préférés par Yang *et al.* pourraient être

utilisés dans le critère global à la place de la modularité (Yang et Leskovec, 2012).

Nous prenons en compte le besoin de l'heuristique de fournir un score identique à une partition avant et après l'étape de fusion.

Nous rappelons que notre démarche se place dans un cadre non supervisé. D'éventuels éléments dont la classe d'appartenance est connue (on parle souvent d'*exemples étiquetés* en classification) ne sont pas utilisés lors de la classification, mais uniquement à des fins d'évaluation. Pour cette raison, la démarche que nous proposons ne permet pas de corriger l'algorithme en cours de route, sur intervention de l'utilisateur, comme dans le cadre de l'apprentissage actif. En contrepartie, elle permet de ne pas avoir besoin d'interrompre l'utilisateur durant le processus de détection de communautés qui peut prendre beaucoup de temps. Soulignons aussi que notre objectif est de proposer des outils simples, aptes à guider l'utilisateur sans nécessiter de paramétrage.

Pour cette raison nous avons testé notre méthode sur un jeu de données réduit mais auquel sont associées plusieurs problématiques et vérités terrain. Nous avons pu montrer que pour détecter certaines partitions les informations relationnelles et vectorielles sont nécessaires. Nous voyons cependant deux inconvénients à notre méthode. La première est que, même si la modularité et l'inertie interclasses sont normalisées, les valeurs prises par les partitions ne sont pas distribuées de la même façon. D'abord, la modularité d'une partition peut être négative, ce qui n'est pas le cas de l'inertie interclasses. De plus, si une partition choisie au hasard aura une modularité proche de zéro, son inertie interclasses sera, elle, plus élevée. Si des partitions d'inerties interclasses voisines de 0,8 ou 0,9 existent à peu près pour tous les jeux de données, la modularité aura le plus souvent une limite haute qui sera inférieure. En la matière, 0,7 est considéré comme une modularité maximale très forte pour un réseau (Newman et Girvan, 2004). Ainsi, les deux grandeurs ne sont peut-être pas directement comparables. Une réponse à ces problèmes est proposée dans le chapitre suivant où nous introduisons une méthode qui associe les données vectorielles et relationnelles d'une façon plus homogène.

CHAPITRE 4

# 2Mod-Louvain, une méthode de détection de communautés basée sur les modularités relationnelle et vectorielle

**Sommaire**



## 4.1 Introduction

Dans le chapitre précédent nous avons proposé de traiter le problème de la détection de communautés dans un réseau d'information à l'aide d'un critère global défini à partir de la modularité et de l'inertie interclasses. Nous avons pu voir que la façon de conjuguer la modularité et l'inertie interclasses dans un critère commun pouvait ne pas apparaître totalement satisfaisante. Nous avons donc cherché un autre critère applicable pour les attributs.

La modularité de Newman et Girvan est un critère qui, bien que n'étant pas parfait, a produit de bons résultats dans la pratique pour la détection de communautés dans un graphe (Fortunato et Barthélemy, 2007; Lancichinetti et Fortunato, 2011). D'abord, elle est calculable sur des graphes valués ou non valués, et ne nécessite pas de normalisation préalable. Elle repose sur des concepts intelligibles, où on cherche à

former des classes entre sommets mieux reliés entre eux que dans une formation aléatoire. De plus, la prise en compte du degré des sommets dans le calcul de la formation aléatoire de référence a apporté une réelle avancée dans le domaine de la détection de communautés. Contrairement à d'autres critères comme la couverture, la modularité permet de comparer des partitions où les nombres de classes sont différents (Almeida et al., 2011). Pour toutes ces raisons, c'est un critère de choix pour la détection de communautés.

Au regard des chapitres précédents, un critère d'évaluation ayant les propriétés de la modularité de Newman et Girvan n'existe pas pour la classification non supervisée d'éléments décrits par des vecteurs. Nous avons proposé dans la section 3.6 d'utiliser l'inertie interclasses. Le critère proposé comprenait une normalisation par l'inertie totale et le nombre de classes de la partition afin de favoriser les partitions contenant peu d'éléments. Mais l'introduction de cette normalisation apporte encore peu de garanties formelles d'efficacité. Notre objectif est donc d'apporter une réponse plus satisfaisante au problème de la mesure de la qualité de la partition vis-à-vis des attributs. Ceci doit être fait en cohérence vis-à-vis de la mesure de la qualité par rapport aux données relationnelles.

C'est l'objectif de la section 4.2 où nous proposons un critère de mesure de la qualité d'une partition d'éléments représentés par des vecteurs. Cette mesure, inspirée de la modularité, pourra être utilisée pour comparer deux partitions. La section 4.3 est consacrée à l'adaptation de ce nouveau critère à l'heuristique de la méthode de Louvain. Nous proposons une nouvelle méthode de détection de communautés dans un réseau d'information appelée 2Mod-Louvain. Elle est basée sur l'optimisation en parallèle de la modularité de Newman et Girvan et de la modularité que nous introduisons. Nous évaluerons ensuite cette méthode sur des réseaux générés dans la section 4.4 et sur des réseaux bibliographiques réels dans la section 4.5.

## 4.2 Critère de modularité basée sur l'inertie

On considère les sommets de l'ensemble $V$ comme des éléments d'un espace vectoriel à $|T|$ dimensions. Dans le cadre de notre recherche, ces éléments peuvent correspondre à des documents représentés sous forme de sacs de mots. $T$ désigne alors l'index associé à la collection de documents et l'ensemble $V$ des documents est plongé dans $\mathbb{R}^{|T|}$. Chaque élément $v \in V$ est un vecteur d'attributs :

$$v = (v_1, \ldots, v_{|T|}) \tag{4.1}$$

On suppose de plus qu'une masse égale à 1 est associée à chaque sommet $v$ de $V$.

La somme de ces masses est égale à $N$, le nombre de sommets de $V$.

Ainsi, alors que la modularité considère la force du lien et vise à regrouper les éléments les plus fortement liés, notre critère exploite le carré de la distance et vise à regrouper ceux qui sont les moins dissemblables. Nous proposons de représenter cet aspect par l'équation 4.2 après normalisation :

$$\frac{\|v - v'\|^2}{2N \cdot I(V)} \tag{4.2}$$

où $I(V)$ est l'inertie de $V$ par rapport à son centre de gravité désignée simplement comme inertie interne de $V$ ou moment centré d'ordre 2 et défini de la façon suivante :

$$I(V) = \sum_{v \in V} m_v \|v - g\|^2 \tag{4.3}$$

où $g$ est le centre de gravité du nuage de points.

Dans le critère global $Q_{inertie}(\mathscr{P})$, la valeur observée, c'est-à-dire le carré de la norme mesurée entre les éléments, est soustraite de la valeur moyenne attendue alors que dans le cas de la modularité c'est la valeur attendue qui est retranchée de la force du lien observé. Cet aspect du carré de la distance attendue est représenté par l'équation 4.4 après normalisation :

$$\frac{I(V, v) \cdot I(V, v')}{(2N \cdot I(V))^2} \tag{4.4}$$

où $I(V, v)$ est l'inertie de $V$ par rapport à $v$.

Le critère de mesure de la qualité d'une partition d'éléments $\mathscr{P}$ représentés sous la forme vectorielle $Q_{inertie}(\mathscr{P})$, que nous introduisons, est défini par :

$$Q_{inertie}(\mathscr{P}) = \sum_{(v,v') \in V \cdot V} \left[ \left( \frac{I(V, v) \cdot I(V, v')}{(2N \cdot I(V))^2} - \frac{\|v - v'\|^2}{2N \cdot I(V)} \right) \cdot \delta\left(c_v, c_{v'}\right) \right] \tag{4.5}$$

où $c(v)$ est la communauté du sommet $v$ et $\delta$ est la fonction de Kronecker.

L'inertie $I(V, v)$ de $V$ par rapport à un élément $v$ est la somme des carrés des distances entre les éléments de $V$ et $v$.

$$I(V, v) = \sum_{v' \in V} m_{v'} \left\|v' - v\right\|^2 \tag{4.6}$$

### 4.2.1 Distance attendue

Le critère $Q_{inertie}$ compare, pour chaque paire de sommets $(v, v')$ issus d'une même communauté le carré de leur distance avec une valeur attendue $d^2_{Exp}(v, v')$ déduite de leurs inerties respectives.

On définit ce carré de la distance attendue par :

$$d^2_{exp}(v, v') = I(V, v) \cdot I(V, v') \tag{4.7}$$

Il s'agit donc de comparer le carré de la distance entre $v$ et $v'$, à une fonction du carré des distances de chacun de ses éléments aux autres éléments de $V$.

Si la valeur attendue est plus grande que la valeur réelle, alors les deux sommets sont de bons candidats à la classification dans une classe commune.

Nous approfondissons les raisons du choix de ce critère dans la section suivante où nous détaillons ses valeurs limites. De plus, nous justifions ici la normalisation des numérateurs ainsi que ses propriétés, de façon à appuyer le choix du carré de la distance attendue proposé précédemment.

### 4.2.2 Bornes du critère de qualité

Le critère $Q_{inertie}(\mathscr{P})$ que nous proposons varie entre -1 et 1. En effet, chaque terme de la soustraction étant compris entre 0 et 1, le critère contraint par la fonction de Kronecker varie donc entre -1 et 1. La partie gauche de la soustraction (équation 4.4), comprenant les produits d'inerties pour toutes les paires de sommets, vaudra au plus 1. De même, la partie droite du critère $Q_{inertie}(\mathscr{P})$ (équation 4.2) ne pourra pas dépasser 1. Les deux parties étant strictement positives, le critère, contraint par les prises de valeurs de la fonction de Kronecker, varie entre -1 et 1.

Démontrons tout d'abord que le terme droit de la soustraction est inférieur ou égal à 1.

Pour ce faire montrons tout d'abord que $I(V, v) = N \cdot d^2(v, g) + I(V)$.

***Démonstration 1***.

$$I(V, v) = \sum_{v' \in V} m_{v'} \left\| v - v' \right\|^2 \tag{4.8}$$

$$= \sum_{v' \in V} m_{v'} d^2(v, v') \tag{4.9}$$

$$= \sum_{v' \in V} m_{v'} \left[ \sum_{j=1}^{|T|} (v_j - v'_j)^2 \right] \tag{4.10}$$

$$= \sum_{v' \in V} m_{v'} \left[ \sum_{j=1}^{|T|} (v_j - g_j + g_j - v'_j)^2 \right] \tag{4.11}$$

$$= \sum_{v' \in V} m_{v'} \sum_{j=1}^{|T|} \left[ (v_j - g_j)^2 + (g_j - v'_j)^2 + 2(v_j - g_j)(g_j - v'_j) \right] \tag{4.12}$$

Considérons chaque terme de la somme de l'équation 4.12. On a :

$$\sum_{v' \in V} m_{v'} \sum_{j=1}^{p} (v_j - g_j)^2 = \sum_{v' \in V} m_{v'} \cdot d^2(v, g) \tag{4.13}$$

$$= N \cdot d^2(v, g) \tag{4.14}$$

$$\sum_{v' \in V} m_{v'} \sum_{j=1}^{p} (g_j - v'_j)^2 = \sum_{v' \in V} m_{v'} \cdot d^2(v', g) \tag{4.15}$$

$$= I(V) \tag{4.16}$$

Enfin :

$$\sum_{v' \in V} m_{v'} \sum_{j=1}^{p} 2(v_j - g_j)(g_j - v'_j) \tag{4.17}$$

$$= \sum_{j=1}^{p} 2(v_j - g_j) \sum_{v' \in V} m_{v'} (g_j - v'_j) \tag{4.18}$$

$$= \sum_{j=1}^{p} 2(v_j - g_j) \left[ \sum_{v' \in V} m_{v'} g_j - \sum_{v' \in V} m_{v'} v'_j \right] \tag{4.19}$$

$$= \sum_{j=1}^{p} 2(v_j - g_j) \left[ N \cdot g_j - N \cdot g_j \right] \tag{4.20}$$

$$= 0 \tag{4.21}$$

En reportant les équations 4.14, 4.16 et 4.21 dans l'équation 4.12, on obtient :

$$I(V, v) = N \cdot d^2(v, g) + I(V) \tag{4.22}$$

□

Montrons ensuite que $\sum_{v\in V}\sum_{v'\in V}\|v-v'\|^2 = 2N\cdot I(V)$ :

***Démonstration 2***. D'après l'équation 4.3, si les masses sont égales à 1, on a :

$$\sum_{v\in V}\sum_{v'\in V}\left\|v-v'\right\|^2 = \sum_{v\in V} I(V,v) \tag{4.23}$$

En reportant 4.22 dans 4.23 il résulte que :

$$\begin{aligned}
\sum_{v\in V}\sum_{v'\in V}\left\|v-v'\right\|^2 =& \sum_{v\in V}\Big[N\cdot d^2(v,g)+I(V)\Big] && (4.24)\\
=& \sum_{v\in V} N\cdot d^2(v,g) + \sum_{v\in V} I(V) && (4.25)\\
=& N\sum_{v\in V} d^2(v,g) + \sum_{v\in V} I(V) && (4.26)\\
=& N\cdot I(V) + N\cdot I(V) && (4.27)\\
=& 2N\cdot I(V) && (4.28)
\end{aligned}$$

□

On en déduit aisément que le terme droit dans le critère $Q_{inertie}(\mathscr{P})$ défini par l'équation 4.5 est inférieur ou égal à 1 :

$$\sum_{v\in V}\sum_{v'\in V}\frac{\|v-v'\|^2}{2N\cdot I(V)}\cdot\delta\left(c_v,c_{v'}\right)\leq 1 \tag{4.29}$$

Montrons à présent que le terme gauche de la soustraction est également inférieur ou égal à 1 :

$$\sum_{v\in V}\sum_{v'\in V}\frac{I(V,v)\cdot I(V,v')}{(2N\cdot I(V))^2}\cdot\delta\left(c_v,c_{v'}\right)\leq 1 \tag{4.30}$$

***Démonstration 3***. On a :

$$\sum_{v\in V}\sum_{v'\in V} I(V,v)\cdot I(V,v') = \sum_{v\in V} I(V,v)\cdot\sum_{v'\in V} I(V,v') \tag{4.31}$$

D'après 4.23 et 4.28, on en déduit que :

$$\sum_{v\in V}\sum_{v'\in V} I(V,v)\cdot I(V,v') = \sum_{v\in V} I(V,v)\cdot 2N\cdot I(V) \tag{4.32}$$

$$= 2N\cdot I(V)\times \sum_{v\in V} I(V,v) \tag{4.33}$$

$$= (2N\cdot I(V))^2 \tag{4.34}$$

□

Par conséquent, on a :

$$\sum_{v\in V}\sum_{v'\in V} \frac{I(V,v)\cdot I(V,v')}{(2N\cdot I(V))^2}\cdot \delta\left(c_v, c_{v'}\right) \leq 1 \tag{4.35}$$

Ces démonstrations nous permettent de déterminer les bornes pour notre critère. Chaque terme de la soustraction étant compris entre 0 et 1, le critère contraint par la fonction de Kronecker varie donc entre -1 et 1.

### 4.2.3 Propriétés du critère de qualité

Ce critère présente plusieurs propriétés intéressantes. Le critère conserve la même valeur, quelle que soit la transformation affine que l'on applique aux attributs, autrement dit l'ajout d'une constante et/ou la multiplication par un scalaire des vecteurs associés aux éléments à classer n'a pas d'incidence sur la valeur du critère.

Cette propriété permet de dire qu'aucune normalisation à priori n'est nécessaire sur les valeurs d'attributs. Enfin l'ordre des attributs n'a aucune incidence sur le résultat.

***Démonstration 4***. Soient deux coefficients $a$ et $b$ de $\mathbb{R}$ et deux ensembles d'éléments $V$ et $V'$ tels que le vecteur associé à un élément $v'$ de $V'$ soit linéairement fonction du vecteur $v$ associé à l'élément correspondant de $V$ et que leurs masses soient identiques :

$$\forall v'\in V',\ v'_j = a\cdot v_j + b,\ j = 1,\ldots,|T| \tag{4.36}$$

On définit $g'$ comme le centre de gravité des éléments de $V'$. Montrons d'abord que $I(V') = a^2\cdot I(V)$ :

$$I(V') = \sum_{v' \in V'} m_{v'} \left\| v' - g' \right\|^2 \tag{4.37}$$

$$= \sum_{v \in V} m_v \left\| a \cdot v + b - (a \cdot g + b) \right\|^2 \tag{4.38}$$

$$= \sum_{v \in V} m_v \left\| a \cdot v - a \cdot g \right\|^2 \tag{4.39}$$

$$= \sum_{v \in V} m_v \sum_{j=1}^{|T|} (a \cdot v_j - a \cdot g_j)^2 \tag{4.40}$$

$$= \sum_{v \in V} m_v \sum_{j=1}^{|T|} a^2 (v_j - g_j)^2 \tag{4.41}$$

$$= \sum_{v \in V} m_v a^2 \left\| v - g \right\|^2 \tag{4.42}$$

$$= a^2 \sum_{v \in V} m_v \left\| \| v - g \right\|^2 \tag{4.43}$$

$$= a^2 \cdot I(V) \tag{4.44}$$

Montrons également que $I(V', v') = a^2 \cdot I(V, v)$ :

$$I(V', v') = N \cdot d^2(v', g') + I(V') \tag{4.45}$$

$$= N \cdot d^2(a \cdot v + b, a \cdot g + b) + I(V') \tag{4.46}$$

En reportant 4.44 dans 4.46 on obtient :

$$I(V', v') = N \cdot a^2 \left\| v - g \right\|^2 + a^2 I(V) \tag{4.47}$$

puis en reportant 4.22 dans 4.47 il s'ensuit que :

$$I(V', v') = a^2 I(V, v) \tag{4.48}$$

Soient deux partitions $\mathscr{P}$ et $\mathscr{P}'$ définies respectivement sur V et V' et équivalentes dans le sens ou tout élément $v_x$ de $V$ est affecté à la classe de $\mathscr{P}$ correspondant à celle de l'élément $v'_x$ de $V'$ tel que $v'_x = a \cdot v_x + b$.

Montrons que $Q_{inertie}(\mathscr{P}') = Q_{inertie}(\mathscr{P})$ :

$$Q_{inertie}(\mathscr{P}') \tag{4.49}$$

$$= \sum_{(v'_x,v'_y)\in V'\times V'} \left[ \left( \frac{I(V',v'_x)\cdot I(V',v'_y)}{(2N\cdot I(V'))^2} - \frac{\left\|v'_x - v'_y\right\|^2}{2N\cdot I(V')} \right) \cdot \delta\left(c_{v'_x}, c_{v'_y}\right) \right] \tag{4.50}$$

$$= \sum_{(v_x,v_y)\in V\times V} \left[ \left( \frac{a^2\cdot I(V,v_x)\cdot a^2 I(V,v_y)}{(2N\cdot a^2\cdot I(V))^2} - \frac{\left\|(a\cdot v_x+b)-(a\cdot v_y+b)\right\|^2}{2N\cdot a^2\cdot I(V)} \right) \cdot \delta\left(c_{v_x}, c_{v_y}\right) \right] \tag{4.51}$$

$$= \sum_{(v_x,v_y)\in V\times V} \left[ \left( \frac{I(V,v_x)\cdot I(V,v_y)}{(2N\cdot I(V))^2} - \frac{a^2\cdot\left\|v_x - v_y\right\|^2}{a^2\cdot 2N\cdot I(V)} \right) \cdot \delta\left(c_{v_x}, c_{v_y}\right) \right] \tag{4.52}$$

$$= Q_{inertie}(\mathscr{P}) \tag{4.53}$$

□

En revanche, ce critère présente aussi certaines limites. Il est indéfini si les vecteurs numériques sont identiques, car l'inertie totale est alors nulle. Ceci n'est pas réellement un inconvénient, car dans ce cas, les attributs n'apportant aucune information, la détection des communautés sera basée uniquement sur les données relationnelles.

De plus, comme la modularité de Newman et Girvan, on peut s'attendre à ce que notre critère présente une limite de résolution. Il convient donc de s'interroger sur la façon d'éviter cette limite de résolution dans un cadre non supervisé où il n'y a pas d'échantillon d'apprentissage permettant de cadrer l'intervalle raisonnable de classes à produire. Une adaptation des travaux d'Arenas *et al.* et Reichardt *et al.* visant à pallier cet effet pourrait être envisagée pour notre critère (Arenas et al., 2008; Reichardt et Bornholdt, 2006). Il s'agirait alors d'introduire un paramètre permettant d'ajuster le comportement du critère.

### 4.2.4 Application sur un exemple

On considère un ensemble $V$ des éléments représentés dans $\mathbb{R}^2$. Cet ensemble est présenté dans la figure 4.1. Tous les sommets sont affectés d'une masse égale à 1. Le tableau 4.2 présente la matrice des carrés des distances euclidiennes associées, normalisée par $N\cdot I(V) = N\cdot\sum_{v\in V} m_v\cdot d^2(v,g)$, ici 42.

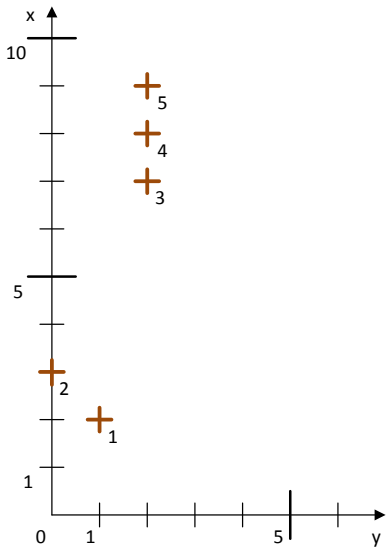



FIGURE 4.1 – Représentation des points de l'exemple

| Point | Coordonnées |
|---|---|
| 1 | $(1;2)$ |
| 2 | $(0;3)$ |
| 3 | $(2;7)$ |
| 4 | $(2;8)$ |
| 5 | $(2;9)$ |

TABLE 4.1 – Coordonnées des éléments de $V$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 0,00476 | 0,06190 | 0,08809 | 0,11904 |
| 2 | 0,00476 | 0 | 0,04761 | 0,06904 | 0,09523 |
| 3 | 0,06190 | 0,04761 | 0 | 0,00238 | 0,00952 |
| 4 | 0,08809 | 0,06904 | 0,00238 | 0 | 0,00238 |
| 5 | 0,11904 | 0,09523 | 0,00952 | 0,00238 | 0 |

TABLE 4.2 – Matrice des carrés des distances, normalisées par l'inertie totale associée à $V$

On calcule l'inertie par rapport à chacun des individus de $V$, selon la formule donnée par l'équation 4.6, puis on la normalise en la divisant par $2N \cdot I(V)$. Les résultats sont donnés dans le tableau 4.3.

| $v$ | $I(V,v)/2N \cdot I(V)$ |
|---|---|
| 1 | 0,27380 |
| 2 | 0,21666 |
| 3 | 0,12142 |
| 4 | 0,16190 |
| 5 | 0,22619 |

TABLE 4.3 – Inertie associée à chaque point de $V$

On calcule la matrice symétrique qui à chaque couple de points associe la distance attendue normalisée (voir tableau 4.4). Ceci s'effectue en multipliant l'inertie normalisée associée aux deux points considérés, prise dans le tableau 4.3.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0,07497 | 0,05933 | 0,03325 | 0,04433 | 0,06193 |
| 2 | 0,05933 | 0,04694 | 0,02631 | 0,03508 | 0,04901 |
| 3 | 0,03325 | 0,02631 | 0,01474 | 0,01966 | 0,02747 |
| 4 | 0,04433 | 0,03508 | 0,01966 | 0,02621 | 0,03662 |
| 5 | 0,06193 | 0,04901 | 0,02747 | 0,03662 | 0,05116 |

TABLE 4.4 – Distance attendue $d^2_{exp}$ entre chaque couple de points

Il est maintenant possible de calculer, pour chaque couple d'individus, le gain de modularité des attributs que l'on obtient quand on les place dans une classe commune. Ce gain est présenté dans la table 4.5. Il prend pour valeur la différence entre les éléments de la matrice de distance et la distance attendue pour chaque couple de points.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | **0,07497** | **0,05456** | -0,02866 | -0,04376 | -0,05711 |
| 2 | **0,05456** | **0,04694** | -0,02131 | -0,03397 | -0,04623 |
| 3 | -0,02866 | -0,02131 | **0,01474** | **0,01728** | **0,01794** |
| 4 | -0,04376 | -0,03397 | **0,01728** | **0,02621** | **0,03424** |
| 5 | -0,05711 | -0,04623 | **0,01794** | **0,03424** | **0,05116** |

TABLE 4.5 – Matrice de gain de modularité des attributs quand on place deux individus dans une même classe

Dans cet exemple, il est facile de voir que l'on a tout intérêt à placer les points 1 et 2 ensemble dans une première classe et les points 3, 4 et 5 dans une seconde classe. La modularité des attributs de la partition $\{\{1,2\},\{3,4,5\}\}$ sera alors de 0,462086, soit la somme des valeurs en gras.

## 4.3 Méthode 2Mod-Louvain

Nous proposons une méthode de détection de communautés dans un réseau d'information tirant parti du critère de modularité basée sur l'inertie $Q_{inertie}$ introduit dans ce chapitre. Nous utilisons ce critère conjointement à la modularité de Newman et Girvan $Q_{NG}(\mathscr{P})$ dans une méthode basée sur le principe d'exploration de la méthode de Louvain comme présenté dans la section 2.3.2.5. Cette méthode, appelée 2Mod-Louvain, consiste à optimiser le critère global $QQ^{+}(\mathscr{P})$ défini par :

$$QQ^{+}(\mathscr{P}) = Q_{NG}(\mathscr{P}) + Q_{inertie}(\mathscr{P}) \tag{4.54}$$

Il convient de noter qu'il n'est pas utile de normaliser ces deux critères $Q_{NG}(\mathscr{P})$ et $Q_{inertie}(\mathscr{P})$ car leurs bornes sont identiques comme démontré dans la section 4.2.2.

La méthode 2Mod-Louvain est détaillée dans l'algorithme 6.

Cet algorithme comporte deux étapes. La première est une phase itérative qui vise à déplacer un sommet de sa classe vers celle d'un de ses voisins dans le graphe si ce changement induit un gain de modularité. La seconde est une phase de fusion qui consiste à construire un nouveau graphe dont les sommets correspondent aux communautés obtenues à l'issue de la phase précédente. Cette seconde phase fait intervenir deux procédures *Fusion_Matrice_Adjacence* et *Fusion_Matrice_Inertie*. La procédure *Fusion_Matrice_Adjacence* est identique à celle mise en œuvre dans la méthode de Louvain et elle a déjà été présentée en détail dans la section 3.3.3.1. La procédure *Fusion_Matrice_Inertie* est décrite dans la section suivante.

**Algorithme 6** : 2Mod-Louvain

**Entrées** : Un réseau d'information $G_0$
**Sorties** : Une partition $\mathscr{P}_{res}$

$\mathscr{P} \leftarrow$ partition discrète des sommets de $V_0$;
$\mathcal{A} \leftarrow$ matrice d'adjacence de $G_0$;
$\mathcal{D} \leftarrow$ matrice des carrés des distances euclidiennes entre les sommets de $V_0$ calculées sur leurs attributs ;
$G \leftarrow G_0$;
**répéter**
    fin $\leftarrow$ faux;
    $QQ^+$_antérieur $\leftarrow QQ^+(\mathscr{P})$ ;
    **répéter**
        **pour tous les** *sommet $u$ de $G$* **faire**
            $B \leftarrow$ communauté voisine maximisant le gain de $QQ^+$;
            **si** *le placement de $u$ dans $B$ induit un gain strictement positif* **alors**
                Placer $u$ dans la communauté $B$;
                Mettre à jour la partition $\mathscr{P}$ suite au transfert de $u$ dans $B$;
    **jusqu'à** *ce qu'aucun sommet ne puisse plus être déplacé* ;
    **si** $QQ^+(\mathscr{P}) > QQ^+$*_antérieur* **alors**
        $G, \mathcal{A} \leftarrow$ Fusion_Matrice_Adjacence$(\mathcal{A}, \mathscr{P})$ ;
        $\mathcal{D} \leftarrow$ Fusion_Matrice_Inertie$(\mathcal{D}, \mathscr{P})$ ;
    **sinon**
        fin $\leftarrow$ vrai ;
**jusqu'à** *fin* ;
$\mathscr{P}_{res} \leftarrow \mathscr{P}$ **partition des sommets de** $V_0$ ;

### 4.3.1 Synthèse des informations de distance dans la deuxième phase

Si le graphe $G$ considéré au début de la phase itérative comporte $|V|$ sommets alors la matrice $\mathcal{D}$ est une matrice carrée symétrique de taille $|V| \times |V|$ dont chaque terme $\mathcal{D}[a, b]$ correspond au carré des distances entre les vecteurs descriptifs des sommets $v_a$ et $v_b$ de $V$. A l'issue de la phase itérative, on obtient une partition $\mathscr{P}'$ de $V$ en $k$ communautés, dont chaque classe va correspondre à un sommet de $V'$ dans le nouveau graphe $G'$. La matrice $\mathcal{D}'$ associée au graphe $G'$ sera définie par :

$$\mathcal{D}'[x, y] = \sum_{(v_a, v_b) \in V \times V} \mathcal{D}[v_a, v_b] \cdot \delta(\tau(v_a), x) \cdot \delta(\tau(v_b), y) \tag{4.55}$$

où la fonction $\tau$ indique pour chaque sommet $v$ de $V$ par quel sommet $v'$, correspondant à sa classe d'affectation, il est représenté dans $V'$.

### 4.3.2 Optimisation de l'algorithme durant la phase itérative par calcul incrémental du gain de modularité

2Mod-Louvain peut être optimisée en calculant uniquement le gain de la modularité basée sur le changement d'inertie induit par le transfert d'un sommet $u$ de sa classe d'origine vers une autre classe. On présente dans cette section les formules permettant d'adapter le calcul incrémental de la modularité, qui a été décrit dans la section 3.4, au critère $QQ^+$. Ces formules permettront d'utiliser l'heuristique de la méthode de Louvain. On vise en particulier certains de ses avantages en terme de temps de calcul localisé du gain de modularité et d'exploration de l'espace des solutions.

En effet, un des avantages de la méthode de Louvain est de limiter les calculs à ceux nécessaires pour connaître la classe dans laquelle il est le plus avantageux d'affecter le sommet étudié. Ces calculs ont été largement détaillés par les auteurs de la méthode (Aynaud et al., 2010).

De même, dans la méthode 2Mod-Louvain, le calcul du gain de modularité basée sur l'inertie peut être limité au calcul du gain induit par le déplacement d'un sommet de sa classe vers celle d'un de ses voisins. Nous détaillons ci-après les optimisations pouvant être opérées par ce calcul local de la modularité basée sur l'inertie.

Considérons deux partitions, $\mathscr{P}$ la partition d'origine et $\mathscr{P}'$ la partition induite par un transfert d'un sommet $u$ de sa classe d'origine $A$ vers sa classe d'affectation $B$.

$$\mathscr{P} = (A, B, C_1, .., C_r) \tag{4.56}$$

$$\mathscr{P}' = (A \setminus \{u\}, B \cup \{u\}, C_1, .., C_r) \tag{4.57}$$

Par la suite, $A \setminus \{u\}$ désigne la classe $A$ privée du sommet $u$ et $B \cup \{u\}$ la classe $B$ augmentée du sommet $u$. Dans un souci de simplification des notations dans la suite nous notons le terme $D[v, v']$ de la matrice abusivement $\mathcal{D}_{vv'}$. La modularité basée sur l'inertie de la partition $\mathscr{P}$ vaut :

$$Q_{inertie}(\mathscr{P}) = \sum_{C \in \mathscr{P}} \frac{1}{2N \cdot I(V)} \sum_{v,v' \in C} \left[ \frac{I(V,v) \cdot I(V,v')}{2N \cdot I(V)} - \mathcal{D}_{vv'} \right] \tag{4.58}$$

$$
\begin{aligned}
=&\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{C\neq A,B}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right] \qquad (4.59)\\
=&\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A\setminus\{u\}}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\left[\frac{I(V,u)^2}{2N\cdot I(V)}-\mathcal{D}_{uu}\right]\\
&+\frac{1}{N\cdot I(V)}\sum_{v\in A\setminus\{u\}}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{C\neq A,B}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right] \qquad (4.60)
\end{aligned}
$$

La modularité de la partition $\mathscr{P}'$ vaut quant à elle :

$$
\begin{aligned}
Q_{inertie}(\mathscr{P}')=&\sum_{C\in\mathscr{P}}\frac{1}{2N\cdot I(V)}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right] \qquad (4.61)\\
=&\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A\setminus u}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B\cup u}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{C\neq A\setminus\{u\},B\cup\{u\}}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right] \qquad (4.62)
\end{aligned}
$$

$$
\begin{aligned}
=&\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A\setminus u}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\left[\frac{I(V,u)^2}{2N\cdot I(V)}-\mathcal{D}(u,u)\right]\\
&+\frac{1}{N\cdot I(V)}\sum_{v\in B}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{C\neq A\setminus\{u\},B\cup\{u\}}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right] \qquad (4.63)
\end{aligned}
$$

Le gain de modularité lors du passage de $\mathscr{P}$ à $\mathscr{P}'$ a donc pour valeur :

$$
\begin{aligned}
\Delta Q_{inertie} =&Q_{inertie}(\mathscr{P}')-Q_{inertie}(\mathscr{P}) \qquad (4.64)\\
=&\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A\setminus\{u\}}\left[\frac{I(V,v)\cdot I(V,v')}{2N}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{N\cdot I(V)}\sum_{v\in B}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\left[\frac{I(V,u)^2}{2N\cdot I(V)}-\mathcal{D}_{uu}\right]\\
&+\frac{1}{2N\cdot I(V)}\sum_{C\neq A\setminus\{u\},B\cup\{u\}}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&-\left[\frac{1}{2N\cdot I(V)}\sum_{v,v'\in A\setminus\{u\}}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\right.\\
&+\frac{1}{2N\cdot I(V)}\sum_{v,v'\in B}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\\
&+\frac{1}{2N\cdot I(V)}\left[\frac{I(V,u)^2}{2N\cdot I(V)}-\mathcal{D}_{uu}\right]\\
&\left.+\frac{1}{N\cdot I(V)}\sum_{v\in B}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right]\right.
\end{aligned}
$$

$$+\frac{1}{2N\cdot I(V)}\sum_{C\neq A\setminus\{u\},B\cup\{u\}}\sum_{v,v'\in C}\left[\frac{I(V,v)\cdot I(V,v')}{2N\cdot I(V)}-\mathcal{D}_{vv'}\right]\Bigg] \quad (4.65)$$

$$=\frac{1}{N\cdot I(V)}\sum_{v\in B}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right]$$

$$-\frac{1}{N\cdot I(V)}\sum_{v\in A\setminus\{u\}}\left[\frac{I(V,u)\cdot I(V,v)}{2N\cdot I(V)}-\mathcal{D}_{uv}\right] \quad (4.66)$$

De plus, on peut remarquer que la variation de modularité induit par la suppression de $u$ de sa classe d'origine sera la même quelque soit sa classe d'affectation. Par conséquent le calcul de variation de modularité peut être effectué en considérant uniquement la différence induite par l'insertion de $u$ dans sa nouvelle communauté d'affectation, décrite par le premier terme de l'équation 4.66.

Ces calculs nous permettent de montrer que notre critère bénéficie lui aussi de la possibilité d'être calculé de façon incrémentale. Le gain de modularité basée sur l'inertie repose uniquement sur des informations locales relatives au sommet déplacé et à sa distance avec les autres sommets.

## 4.4 Évaluation de la méthode *2Mod-Louvain* sur des réseaux artificiels

On propose, comme on l'a fait pour ToTeM dans la section 3.7, d'évaluer la méthode 2Mod-Louvain qui optimise le critère global $QQ^+$ basé à la fois sur la modularité de Newman et Girvan et la modularité par rapport à l'inertie. Dans un premier temps, nous étudions la robustesse de la méthode sur des réseaux artificiels vis-à-vis d'une dégradation de la structure de communautés définie par rapport aux relations, ou des classes définies par rapport aux attributs, ou encore d'une augmentation de la taille du réseau d'information ou d'une variation de la densité des liens.

L'évaluation sera faite selon une vérité externe en fonction des critères de NMI, d'ARI, d'AMI, de nombre de classes et, quand c'est possible, de taux de bien classés qui ont été définis dans la section 2.2.3.2. On notera que les évolutions du réseau ont été opérées ici indépendamment de celles évoquées dans le chapitre précédent, ce qui explique des résultats différents pour les méthodes de Louvain et des K-means.

### 4.4.1 Réseau de référence (réseau R)

On utilise, de même que dans la section 3.7.1 dédiée à l'évaluation de la méthode ToTeM, un réseau de référence R qui comporte 3 classes composées chacune de

| | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classe 1 | 55 | | |
| Classe 2 | 2 | 53 | |
| Classe 3 | 1 | 7 | 50 |

TABLE 4.6 – Répartition des extrémités des liens du graphe R

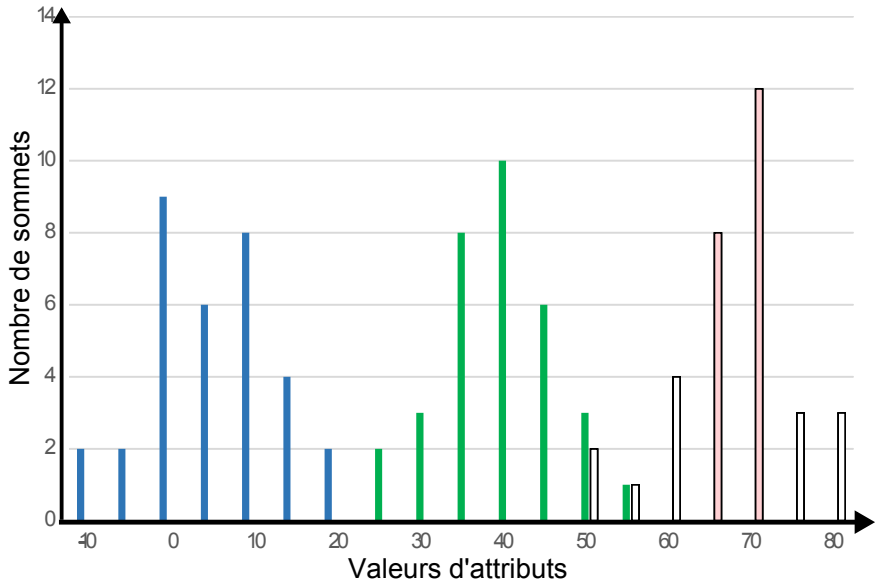


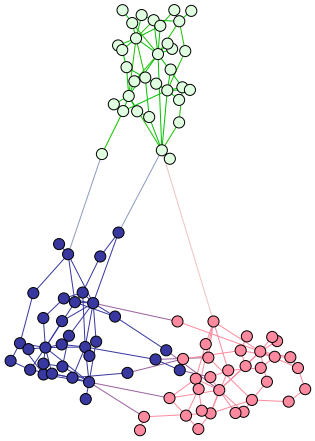

FIGURE 4.2 – Distribution des valeurs de l'attribut des sommets de R par classe

33 sommets. Chaque sommet est décrit par une valeur réelle. Nous considérons les mêmes paramètres de génération de ce réseau. Les attributs suivent une loi normale d'écart-type 7, centrée autour d'une valeur propre à sa classe d'origine. Ainsi la première classe a un centre de 10, la deuxième un centre de 40 et la troisième un centre de 70. La classe d'origine du sommet sert de vérité terrain pour l'évaluation. Enfin, durant la génération du réseau de référence, nous avons fait en sorte que le calcul précédent de génération des arêtes crée au maximum deux arêtes à chaque fois qu'un nouveau sommet est introduit.

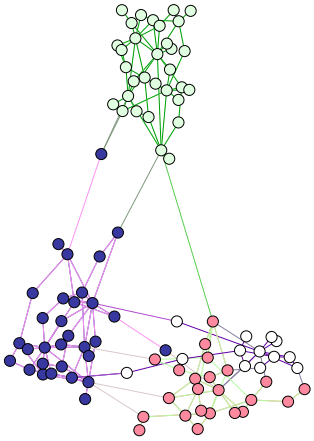
Le réseau R, qui servira de référence, est représenté dans la figure 4.3a.

Il comporte 99 sommets et 168 arêtes. La table 4.6 montre la répartition des arêtes entre les classes dans le graphe R.

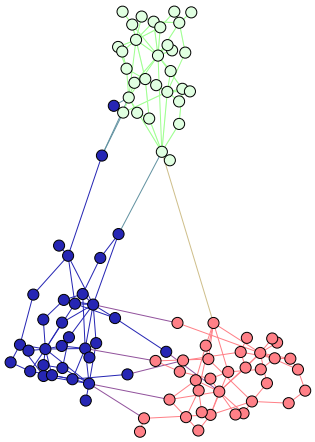
La distribution des valeurs de l'attribut attaché aux sommets de chaque classe est présentée dans la figure 4.2. La figure 4.3a illustre le graphe, issu du modèle, qui nous servira de référence.



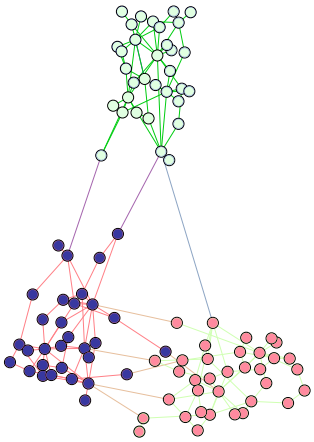
(a) Classes de la vérité terrain



(b) Application de 2Mod-Louvain



(c) Application de la méthode de Louvain



(d) Application des K-means

FIGURE 4.3 – Partitions du réseau de reéférence R

### Application de la méthode de Louvain

La visualisation du résultat de l'application de la méthode de Louvain sur le réseau de référence est présentée dans la figure 4.3c. La matrice de coïncidence associée est présentée dans le tableau 4.7.

| Classes prédites → / Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Classe 4 |
|---|---|---|---|---|
| Classe 1 | **32** | 1 | | |
| Classe 2 | | **30** | 2 | 1 |
| Classe 3 | | | **21** | 12 |

TABLE 4.7 – Matrice de coïncidence associée à l'application de la méthode de Louvain qui produit 4 classes sur le réseau de référence R

Le taux de sommets bien classé s'élève à 84%. Le score de NMI est de 0,78.

La méthode de Louvain considère uniquement les données relationnelles. On constate que les classes réelles sont bien identifiées, mais la troisième est scindée en deux. 21 sommets sont affectés à l'une des classes prédites et 12 à une autre.

### Application des K-means

Les résultats des K-means sur le réseau de référence sont présentés dans le tableau 4.8 et illustrés par la figure 4.3d.

| Classes prédites → / Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classe 1 | | **31** | 2 |
| Classe 2 | 2 | | **31** |
| Classe 3 | **33** | | |

TABLE 4.8 – Matrice de coïncidence du réseau de référence R.1.1 issue de l'application des K-means

Le taux de bien classés est de 96%. La NMI est de 0,86. On constate que les K-means obtiennent donc sur cette tâche un bon résultat mais rappelons que cet algorithme nécessite un paramétrage correspondant au nombre de classes à produire.

### Application de 2Mod-Louvain

Les résultats de 2Mod-Louvain sur le réseau de référence sont présentés dans le tableau 4.9 et illustrés par la figure 4.3b.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classe 1 | 33 | | |
| Classe 2 | | 31 | 2 |
| Classe 3 | | | 33 |

TABLE 4.9 – Matrice de coïncidence du réseau de référence R issue de l'application de 2Mod-Louvain

Le taux de bien classés s'élève à 98%. La NMI est de 0,93. On constate que sur ce jeu posant *a priori* peu de difficultés, la combinaison des informations est déjà bénéfique puisqu'elle permet en particulier de corriger la scission d'une classe par la méthode de Louvain.

### 4.4.2 Dégradation de l'information relationnelle (réseaux R.1.1 et R.1.2)

Un nombre important d'arêtes intraclasses aide à la fois la méthode de Louvain et 2Mod-Louvain à trouver les communautés de la vérité terrain. On veut savoir si la méthode parvient à maintenir ses résultats dans la situation où l'information relationnelle est dégradée. Pour cela, on réduit le nombre d'arêtes intraclasses et on introduit à la place des arêtes interclasses.

On réutilise l'algorithme de dégradation de l'information relationnelle présenté dans le chapitre précédent. On introduit un paramètre qui détermine la proportion d'arêtes intraclasses à remplacer par une arête interclasses.

Pour le graphe dégradé à 25%, la matrice de coïncidence est présentée dans le tableau 4.10. Le taux de bien classés est de 78%. La NMI s'élève à 0,60.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Classe 4 | Classe 5 |
|---|---|---|---|---|---|
| Classe 1 | **33** | | | | |
| Classe 2 | 3 | **18** | 3 | 9 | |
| Classe 3 | 1 | 1 | **26** | 3 | 2 |

TABLE 4.10 – Matrice de coïncidence du graphe R.1.1 dégradé à 25%

Pour le réseau R.1.2 dégradé à 50%, la matrice de coïncidence est présentée dans le tableau 4.11. Le taux de bien classés s'élève à 63%. La NMI s'élève à 0,35.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Classe 4 | Classe 5 | Classe 6 |
|---|---|---|---|---|---|---|
| Classe 1 | **29** | | | 1 | 1 | 2 |
| Classe 2 | 6 | 6 | 6 | 2 | **10** | 3 |
| Classe 3 | 1 | 2 | **23** | 2 | 4 | 1 |

TABLE 4.11 – Matrice de coïncidence du graphe R.1.2

On constate que 2Mod-Louvain souffre de la dégradation de l'information relationnelle et produit alors des classes plus nombreuses et moins pertinentes. Elle est cependant moins pénalisée que la méthode de Louvain, qui atteint elle 31% de bien classés.

### 4.4.3 Dégradation des attributs (réseaux R.2.1 et R.2.2)

On teste ensuite la robustesse de la méthode envers des réseaux d'information où l'attribut est moins caractéristique de chacune des classes de la vérité terrain. Pour cela, on propose d'augmenter l'écart-type de l'attribut en le fixant à 10 alors qu'il valait 7 dans le réseau de référence. Le but est d'obtenir des distributions des différentes classes qui se chevauchent de plus en plus du point de vue des attributs. Ainsi on considère que les attributs descriptifs des sommets des 3 classes suivent respectivement des lois normales de paramètres (10, 10), (40, 10), (70, 10) pour le graphe R.2.1 puis (10, 12), (40, 12), (70, 12) pour le graphe R.2.2.

La matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.12. Le taux de bien classés est de 96%. La NMI s'élève à 0,89.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classes 1 | **33** | | |
| Classes 2 | | **29** | 4 |
| Classes 3 | | | **33** |

TABLE 4.12 – Matrice de coïncidence du graphe R.2.1 avec des écarts-types de 10

Pour le réseau d'information où les attributs ont été remplacés par des attributs dégradés d'écart-type 12, la matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.13. Le taux de bien classés est de 98%. La NMI s'élève à 0,93.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classes 1 | **33** | | |
| Classes 2 | | **31** | 2 |
| Classes 3 | | | **33** |

TABLE 4.13 – Matrice de coïncidence du graphe R.2.2 avec des écarts-types de 12

On constate que la classification est peu modifiée par l'étalement des distributions des valeurs d'attributs. Avec des taux de réussite de 88% pour le réseau R.2.1 et de 90% pour le réseau R.2.2, les K-means ont subi une dégradation de leurs résultats. La prise en compte de l'information relationnelle a permis de limiter la dégradation subie par le résultat de notre méthode.

### 4.4.4 Augmentation de la taille du réseau (réseaux R.3.1 et R.3.2)

On cherche ensuite à déterminer l'influence du nombre de sommets sur les résultats de la classification. On propose d'ajouter des sommets dans le réseau de référence et de mesurer les résultats sur de nouveaux réseaux comportant 999 et 9 999 sommets.

Pour le réseau à 999 sommets répartis en 3 classes de 333 sommets, la matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.14. Le taux de bien classés est de $84\%$. La NMI s'élève à 0,80.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Classe 4 |
|---|---|---|---|---|
| Classes 1 | **330** | 1 | | 2 |
| Classes 2 | 9 | **183** | 11 | 130 |
| Classes 3 | | 7 | **326** | |

TABLE 4.14 – Matrice de coïncidence du réseau R.3.1 à 999 sommets

Pour le graphe à 9 999 sommets répartis en 3 classes de 3 333 sommets, la matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.15. Le taux de bien classés est de 85,46%. La NMI s'élève à 0,77.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Classe 4 |
|---|---|---|---|---|
| Classes 1 | **3 278** | 5 | | 50 |
| Classes 2 | 251 | 1 030 | 75 | **1 977** |
| Classes 3 | 2 | 28 | **3 291** | 12 |

TABLE 4.15 – Matrice de coïncidence du graphe R.3.2 à 9 999 sommets

Dans la mesure où il est difficile de produire des résultats en termes de nombres de biens classés quand le nombre de classes produites ou réelles devient important, nous comparerons les partitions produites par 2Mod-Louvain à ceux produits par la méthode de Louvain et les K-means par l'indice de NMI. La méthode de Louvain produit, pour les réseaux de 999 et 9 999 sommets, des partitions de NMI s'élevant à 0,60. Au passage, les nombres de classes grandissent pour s'élever à 10 et 13. Les K-means produisent des résultats de 0,88 et 0,89. Cela montre que le résultat des K-means ne diminue pas lorsque la taille du réseau augmente. Ceci est normal dans la mesure où la proportion des éléments qui se chevauchent demeure identique dans les 3 distributions malgré l'évolution de la taille du réseau.

On constate que l'augmentation de la taille du graphe a un impact limité sur les résultats de 2Mod-Louvain qui restent très satisfaisants. Le nombre de classes produites est contenu par rapport au résultat donné par la méthode de Louvain. Cette dernière apparaît elle comme ayant plus de mal à trouver la structure communautaire à mesure que le réseau grandit.

L'augmentation de la taille du graphe nous permet également d'étudier l'évolution des temps de traitement des différentes méthodes. Sur le réseau de référence, la méthode de Louvain et les K-means ont des temps d'exécution très rapides, inférieurs à la seconde. 2Mod-Louvain dure lui une dizaine de secondes. Sur le réseau de 999 sommets, 2Mod-Louvain prend 12 minutes. Sur le réseau de 9 999 sommets, les temps d'exécution des K-means restent quasi instantanés, la méthode de Louvain perd moins de 20 secondes et 2Mod-Louvain environ 3 heures. Ces temps d'exécution montrent les limites de 2Mod-Louvain pour ce qui est de la classification dans de grands réseaux d'information. Pour diminuer ces temps de calcul, il peut être envisagé de mémoriser les valeurs de critères pour les partitions, qui sont susceptibles d'être recalculées pendant le déroulement de l'algorithme. Dans tous les cas, une approche ne nécessitant pas de considérer la distance d'un sommet avec tous les autres lors d'un déplacement serait un réel apport pour manipuler des réseaux de plus grandes tailles ou faisant l'objet de plus d'attributs.

### 4.4.5 Augmentation du nombre d'arêtes (réseaux R.4.1 et R.4.2)

On cherche ici à déterminer si la densité d'arêtes dans le réseau a une influence sur le résultat. Le réseau R utilise une valeur de 2 arêtes par sommet introduit (les résultats sont présentés à la section 4.4.1). On propose de tester les résultats avec des valeurs de 5 et 10 arêtes.

Pour le réseau R.4.1 à 5 arêtes par sommet, la matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.16. Le taux de bien classés s'élève à 94%. La NMI s'élève à 0,82.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classe 1 | **33** | | |
| Classe 2 | 4 | **27** | 2 |
| Classe 3 | | | **33** |

TABLE 4.16 – Matrice de coïncidence du graphe R.4.1

Pour le graphe à 10 arêtes par sommet, la matrice de coïncidence issue de l'application de 2Mod-Louvain est présentée dans le tableau 4.17. Le taux de bien classés est de 98%. La NMI est de 0,92.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 |
|---|---|---|---|
| Classe 1 | **33** | | |
| Classe 2 | 1 | **31** | 1 |
| Classe 3 | | | **33** |

TABLE 4.17 – Matrice de coïncidence du graphe R.4.2

On constate que l'augmentation de la densité ne modifie pas les résultats de façon significative, qui demeurent élevés. Avec des taux de bien classés de 96 et 97% respectivement sur R.4.1 et R.4.2, la méthode de Louvain trouve ses scores améliorés par l'augmentation de la densité du réseau.

### 4.4.6 Synthèse des résultats des méthodes 2Mod-Louvain, Louvain et des K-means et conclusion

On récapitule les résultats en matière de taux de bien classés (tableau 4.18a) et de NMI (tableau 4.18b).

La méthode proposée réussit à apporter de la robustesse à la méthode de Louvain face à la dégradation de l'information relationnelle. Dans le cas où la taille du ré-

seau augmente, la méthode proposée permet de parer à la multiplication des classes qui survient alors avec la méthode de Louvain (4 classes contre 10). Les K-means conservent de bons résultats dans le cas où la taille du réseau évolue, car l'information des attributs demeure de bonne qualité. De plus contrairement aux deux autres méthodes, le nombre de classes étant donné, la multiplication des classes n'est pas un risque pour les K-means.

| | 2Mod-Louvain | | Louvain | | K-means |
|---|---|---|---|---|---|
| | TBC (%) | Nb de classes | TBC (%) | Nb de classes | TBC (%) |
| | Graphe de référence | | | | |
| **R** | **98** | 3 | 84 | 4 | 96 |
| | Dégradation de l'information relationnelle | | | | |
| **R.1.1** | **78** | 5 | 40 | 8 | (96)* |
| **R.1.2** | **63** | 6 | 31 | 8 | (97)* |
| | Étalement des distributions | | | | |
| **R.2.1** | **96** | 3 | (84)* | 4 | 88 |
| **R.2.2** | **98** | 3 | (84)* | 4 | 90 |
| | Taille du réseau d'information | | | | |
| **R.3.1** | 84 | 4 | 50 | 10 | **97** |
| **R.3.2** | 85,46 | 4 | † | 13 | † |
| | Densité | | | | |
| **R.4.1** | 94 | 3 | **96** | 3 | (96)* |
| **R.4.2** | **98** | 3 | 97 | 3 | (97)* |

* La dégradation de l'information relationnelle et le changement de densité n'influençant pas les résultats des K-means, et la dégradation de l'information des attributs n'influençant pas les résultats de la méthode de Louvain, ces résultats ne sont pas pris en compte dans la comparaison.

† Le calcul du taux de bien classés est impraticable compte tenu du nombre élevé de classes produites.

(a) Évaluation selon le taux de bien classés

| NMI | 2Mod-Louvain | Louvain | K-means |
|---|---|---|---|
| | Graphe de référence | | |
| **R** | **0,93** | 0,78 | 0,86 |
| | Dégradation de l'information relationnelle | | |
| **R.1.1** | **0,60** | 0,31 | (0,86) |
| **R.1.2** | **0,35** | 0,13 | (0,91) |
| | Étalement des distributions | | |
| **R.2.1** | **0,89** | 0,78 | 0,69 |
| **R.2.2** | **0,93** | 0,78 | 0,69 |
| | Taille du réseau d'information | | |
| **R.3.1** | 0,80 | 0,60 | **0,88** |
| **R.3.2** | 0,77 | 0,60 | **0,89** |
| | Densité | | |
| **R.4.1** | 0,82 | **0,85** | (0,86) |
| **R.4.2** | **0,91** | 0,88 | (0,89) |

(b) Évaluation selon la NMI

TABLE 4.18 – Bilan de l'expérimentation sur des réseaux artificiels

## 4.5 Évaluation sur des réseaux réels

Nous complétons les résultats expérimentaux précédents par une étude sur des graphes réels. On propose d'évaluer la méthode 2Mod-Louvain sur le réseau des 4 sessions. Ensuite, nous l'évaluerons pour la classification d'articles issus du réseau de données bibliographiques PubMed-Diabetes.

### 4.5.1 Réseau des 4 sessions

On utilise ici le jeu de données dont la construction a été décrite dans la section 1.4.4. On rappelle que ce réseau d'information comporte 99 auteurs et 2 623 relations de coparticipation à des conférences.

On applique 2Mod-Louvain sur le réseau d'information des 4 sessions. Comme le montre le tableau 4.19, 2Mod-Louvain trouve à trois sommets près le découpage selon les deux conférences, ce qui était déjà le cas pour la méthode ToTeM. Selon l'évaluation envers les quatres sessions, le taux de bien classés s'élève à 63%.

| Classes prédites → Classes réelles ↓ | Classe 1 | Classe 2 | Classe 3 | Total |
|---|---|---|---|---|
| A - SAC Bioinformatique | | **24** | | 24 |
| B - SAC Robotique | | 16 | | 16 |
| C - IJCAI Robotique | **37** | 1 | | 38 |
| D - IJCAI Contraintes | 19 | 1 | **1** | 21 |
| Total | 56 | 42 | 1 | 99 |

TABLE 4.19 – Résultat de l'application de 2Mod-Louvain sur le réseau des 4 sessions

Les résultats issus de l'application de la méthode de Louvain sont présentés dans le tableau 4.20, où le taux de bien classés s'élève à 63%. On peut voir que les deux conférences sont identifiées, mais pas les sessions. Ce résultat est à comparer avec les résultats de la méthode de Louvain et des K-means présentés dans la section 3.8.3.3 qui sont respectivement de 63% et 41%. Le bilan de cette expérimentation est que la composante textuelle incarnée par les vecteurs *tf-idf* n'a pas une influence suffisante pour permettre à 2Mod-Louvain de différencier les résultats de ceux de la méthode de Louvain appliquée seule. La méthode atteint cependant le meilleur des résultats procurés par chacune des deux modalités isolées.

| Session prédite → Session réelle ↓ | Classe 1 | Classe 2 | Total |
|---|---|---|---|
| A - SAC Bioinformatique | **24** | | 24 |
| B - SAC Robotique | 16 | | 16 |
| C - IJCAI Robotique | 1 | **37** | 38 |
| D - IJCAI Contraintes | 1 | 20 | 21 |
| Total | 42 | 57 | 99 |

TABLE 4.20 – Résultat de l'application de Louvain sur le réseau des 4 sessions

En conclusion, ce réseau a une information relationnelle trop proéminente pour que les valeurs des attributs soient exploitées avantageusement.

### 4.5.2 Jeu de données PubMed-Diabètes

Nous considérons le jeu de données PubMed-Diabètes qui a été présenté dans la section 3.9.1 (Sen et al., 2008). Nous rappelons qu'il comprend 19 717 publications scientifiques traitant du diabète, qui sont réparties en trois catégories. L'index associé comporte 500 termes.

D'abord, pour la méthode des K-means, le taux de bien classés est de 60%. C'est un taux élevé, mais qui doit être analysé au regard des 3 classes qui ont été imposées.

Quant à la méthode de Louvain, celle-ci produit un nombre de classes élevé (36) et un taux de bien classés de 25%.

Les résultats complets sont présentés dans le tableau 4.21.

| | 2Mod-Louvain | Louvain | K-means |
|---|---|---|---|
| Vérité terrain non connexifiée (3 catégories réelles) | | | |
| NMI | **0,24** | 0,23 | 0,18 |
| ARI | 0,09 | 0,11 | **0,15** |
| AMI | 0,13 | 0,13 | **0,17** |
| V-Mesure | **0,21** | 0,20 | 0,18 |
| Homogénéité | **0,14** | 0,1348 | 0,18 |
| Complétude | **0,43** | 0,39 | 0,17 |
| Taux de bien classés | 0,22 | 0,25 | **0,60** |
| Nombre de classes | 64 | 36 | (3) |

TABLE 4.21 – Résultat de l'évaluation de 2Mod-Louvain et des méthodes de référence sur PubMed-Diabètes

Le bilan de cette expérimentation est que notre proposition donne les meilleurs résultats pour plusieurs critères et procure toujours un résultat bien placé face aux méthodes de référence. Il est difficile d'interpréter les résultats au regard du nombre

de bien classés ; les nombres de classes sont variables et la partition en 3 classes est ainsi très favorisée.

2Mod-Louvain produit plus de classes (64), mais celles-ci sont mesurées comme s'appariant mieux à la vérité terrain du point de vue des critères perfectionnés de NMI et de V-mesure.

En effet, une des difficultés rencontrées par notre méthode et celle de Louvain est qu'elle favorise les communautés connexes. Or dans ce jeu de données les classes de la vérité terrain ne sont pas connexes. L'évaluation face à la vérité terrain "connexifiée" consiste à faire subir un traitement préalable à la vérité terrain. Ainsi s'il n'existe pas de chemin entre deux sommets d'une même classe ne passant que par des sommets de cette même classe, alors il y a séparation de la classe en deux nouvelles classes connexes. Vis-à-vis de la vérité terrain où les classes ont subi ce traitement pour devenir connexes (voir section 3.2), le taux de bien classé n'est plus calculable en temps raisonnable en raison de la hausse du nombre de classes réelles, mais les autres indicateurs confirment l'amélioration des résultats produits par les méthodes basées sur la modularité et au contraire une dégradation de ceux fournis par les K-means.

| | 2Mod-Louvain | Louvain | K-means |
|---|---|---|---|
| Vérité terrain connexifiée (2 644 catégories réelles) | | | |
| NMI | **0,37** | 0,36 | 0,18 |
| ARI | 0,12 | **0,15** | 0,11 |
| AMI | 0,21 | **0,21** | 0,06 |
| V-Mesure | **0,37** | 0,35 | 0,16 |
| Homogénéité | 0,32 | **0,32** | 0,28 |
| Complétude | **0,43** | 0,40 | 0,11 |
| Nombre de classes | 64 | 36 | (3) |

TABLE 4.22 – Résultat de l'évaluation de 2Mod-Louvain et des méthodes de référence sur PubMed-Diabètes, après connexification des classes

Une visualisation du réseau coloré selon les communautés formées est présentée par la figure 4.4. On voit que malgré le nombre élevé de communautés découvertes, un petit nombre s'étendent assez largement sur le réseau. En effet, 7 des 64 classes contiennent 50% des sommets du réseau.

## 4.6 Conclusion

Dans ce chapitre, nous avons défini la modularité basée sur l'inertie, un indice de mesure de la qualité d'une partition d'éléments décrits dans un espace vectoriel. Cet indice se veut un pendant de la modularité de Newman et Girvan, adapté au

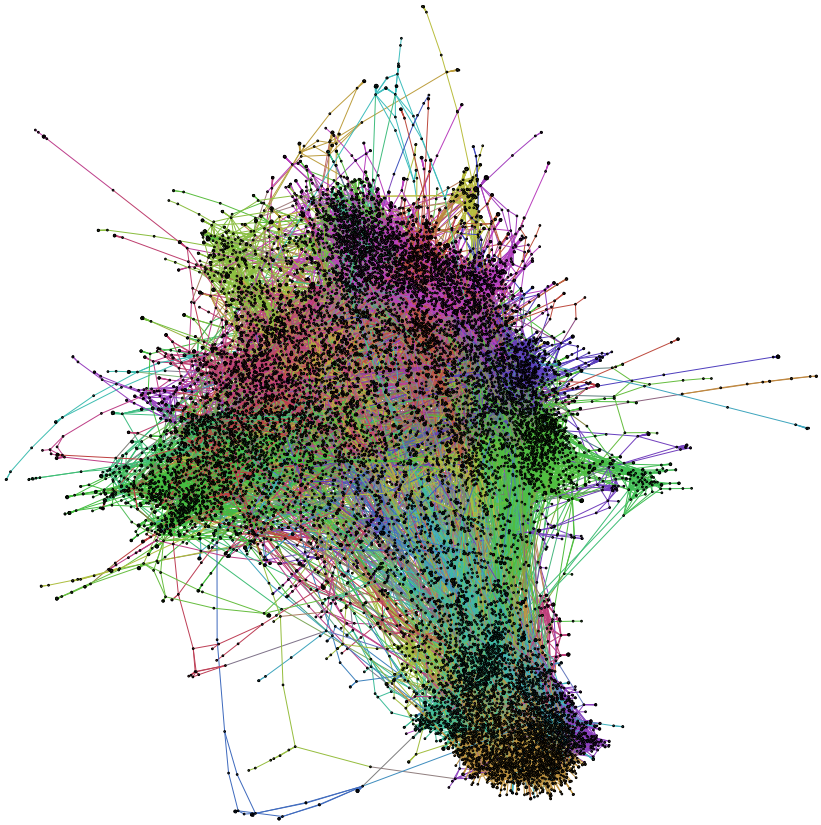


FIGURE 4.4 – 2Mod-Louvain appliqué à PubMed

contexte de la classification non supervisée. Il permet de produire une classification ne nécessitant pas comme paramètre le nombre de classes à produire. Pour cela, il compare le carré de la distance entre deux points à sa valeur attendue compte tenu des valeurs d'inertie de chacun des deux points. Nous avons montré que notre critère comportait une normalisation intrinsèque et qu'il apportait certaines des propriétés de la modularité de Newman et Girvan au domaine de la classification de vecteurs.

Nous avons proposé 2Mod-Louvain, une méthode utilisant notre critère de modularité basée sur l'inertie conjointement à la modularité de Newman et Girvan pour détecter des communautés dans des réseaux d'information artificiels et réels. Nous avons testé la robustesse de notre proposition face à diverses modifications que peut subir un réseau d'information. 2Mod-Louvain a ainsi montré de bons résultats sur des réseaux ayant subit une dégradation des relations, des attributs, une augmentation de la densité des relations et de la taille du réseau. Alors que la méthode de Louvain produit un nombre de classes plus important lorsque le réseau d'information grandit, notre combinaison des deux informations relationnelles et des attributs s'est avérée plus robuste. Sur les réseaux réels, bien que l'ARI, plus sensible au nombre de classes, soit plus sévère, l'évaluation a montré que notre proposition fournissait de bons résultats selon les critères de NMI ou de V-mesure, sur des réseaux où la dégradation des liens ou des attributs ne permettait plus d'obtenir des taux de bien classés satisfaisants avec la méthode de Louvain ou avec les K-means.

Ce qui semble être la plus grande faiblesse du critère de modularité basée sur l'inertie est sa complexité. Si la modularité de Newman et Girvan tire parti des matrices d'adjacence creuses, les matrices de distance ne le sont jamais. Une étude des améliorations algorithmiques possibles est nécessaire afin de pouvoir proposer une méthode pouvant traiter des réseaux de tailles supérieures à 10 000 sommets. De plus, nous nous attendons à ce que notre critère soit affecté, comme la modularité de Newman et Girvan, par une limite de résolution. Il serait judicieux de poursuivre ce travail par l'étude des conséquences que cette limite apporte sur la classification de vecteurs numériques.

Ainsi, les perspectives que nous jugeons intéressantes concernent-elles l'étude du critère proposé dans un cadre faisant uniquement intervenir des vecteurs numériques, notamment pour étudier la sensibilité à la limite de résolution. De plus, un cadre plus formel pour considérer simultanément les relations et chacun des attributs pourrait permettre de combiner ces informations de façon plus pertinente. Ainsi la conception d'un réseau d'information aléatoire et d'une modularité associée unifiant les deux paradigmes incarnés par les relations et les attributs semble également être une voie digne d'intérêt.

CHAPITRE 5

# Conclusion et perspectives

Dans cette thèse, nous avons étudié le problème de la détection de communautés dans des réseaux d'information, c'est à dire des graphes dont les sommets sont munis d'un vecteur d'attributs numériques. L'objectif était de permettre la détection de communautés dans ces réseaux, en tirant parti le mieux possible des informations relationnelles et vectorielles.

Nous nous sommes placés dans le contexte difficile de l'apprentissage non supervisé. Nous ne disposons ainsi que du réseau d'information et devons retourner, sans rien connaître ni du processus de génération des données ni de l'attente de l'utilisateur, la meilleure partition possible.

Nous avons abordé le problème en produisant des états de l'art distincts sur la classification automatique puis la détection de communautés. De cette façon nous avons pu étudier les approches pratiques pour les deux types de classification. Nous avons ensuite abordé les techniques existantes en matière de combinaison des informations relationnelles et d'attributs. Cette démarche nous a permis de mieux comprendre les limites des approches existantes de détection de communautés dans des réseaux d'information.

Une des premières barrières qui nous est apparue est le problème de la normalisation des critères employés sur les liens et les attributs. C'est la raison pour laquelle notre première méthode ToTeM a fait l'objet de plusieurs suggestions de critères globaux. Une deuxième difficulté tient à la pondération des deux informations, dont l'une peut être plus pertinente que l'autre, ou plus nuancée par exemple. C'est un problème qui n'est pas souvent soulevé, mais qui apparaît très vite lorsque l'on se penche sur la classification non supervisée de données hétérogènes.

La première méthode que nous avons introduite, ToTeM, montre que l'optimisation de l'inertie interclasses combinée à la modularité de Newman et Girvan, s'intègre bien dans l'heuristique multi-échelle de la méthode de Louvain. Comme pour la modularité de Newman et Girvan, on peut calculer de manière incrémentale un gain (ou une perte) d'inertie induite par le changement d'affectation d'un sommet. Un autre point au moins aussi intéressant est que nous avons étendu le principe de changement d'échelle pour l'appliquer aux attributs. En effet, en tenant compte des masses associées initialement à chaque sommet, l'inertie interclasses garde sa cohérence lors

de la phase de fusion des sommets. Cette propriété étant également présente dans le critère de modularité de Newman et Girvan, elle nous assure que l'utilisation d'un critère global s'appuyant uniquement sur ces deux mesures ne varie pas lors d'un changement d'échelle, tel que dans la phase de fusion de Louvain. Cela fait de l'inertie interclasses un critère ayant de très bonnes propriétés pour la classification, même adapté à une heuristique, celle de la méthode de Louvain, initialement introduite pour la détection de communautés dans les graphes.

Cependant, malgré ses bonnes propriétés mathématiques, le critère d'inertie interclasses a comme défaut le fait que le changement d'affectation d'un élément entraîne des calculs faisant intervenir tous les sommets de son ancienne et de sa nouvelle classe. De ce fait les traitements sont plus lourds que pour la modularité de Newman et Girvan, pour laquelle seuls les degrés avec les sommets voisins sont considérés.

Nous avons exposé plusieurs critères globaux d'optimisation qui peuvent être utilisés dans le cadre de notre méthode. Tous sont dépendants de l'inertie interclasses et de la modularité de Newman et Girvan, et certains font usage du nombre de classes produites. Nous avons ensuite comparé ToTeM à des méthodes de référence sur des graphes réels et artificiels. Nous avons montré sur un jeu de données spécifiquement conçu que la prise en compte des deux informations permettait de trouver des partitions qui ne l'auraient pas été avec des méthodes n'utilisant qu'un seul type de données. L'évaluation sur des graphes artificiels produits par des générateurs nous a de plus permis de connaître les points forts et les faiblesses de ToTeM au regard des caractéristiques des réseaux.

De manière à répondre à ces faiblesses et à certains problèmes de normalisation du critère d'inertie interclasses, nous avons proposé une deuxième méthode de détection de communautés dans des réseaux d'information dénommée 2Mod-Louvain. Elle tire parti d'un nouveau critère que nous avons introduit, que nous appelons la modularité basée sur l'inertie. C'est un critère pour la classification automatique non supervisée, qui agit indépendamment du nombre de classes de la partition évaluée. Nous avons montré que ce critère gardait une valeur constante face à une transformation linéaire des données. Prenant sa valeur entre -1 et 1, il est d'autant plus grand que la partition testée est bonne. Bien que ce critère soit inspiré de la modularité de Newman et Girvan, il ne repose pas sur les concepts de degrés et de distribution des arêtes, inexistants en classification automatique. Il utilise des notions d'inertie par rapport à un point et d'inertie totale. Le modèle nul, qui désigne dans la modularité classique la valuation attendue entre deux sommets en fonction de leurs degrés, est remplacé par le carré de la distance attendue, qui se calcule d'une façon très similaire.

L'exploitation en parallèle des deux critères de modularité nous permet de jeter un regard nouveau sur le problème de la détection de communautés combinant les deux

informations.

Pour donner une égale importance aux deux types d'information, nous avons conçu notre algorithme 2Mod-Louvain autour d'un critère global défini comme la somme des deux critères de modularité. Nous avons donc proposé une réponse à la problématique de détection de communautés dans un réseau d'information par une adaptation de la méthode de Louvain opérant à la fois sur les données relationnelles et d'attributs.

Nous avons conduit des expérimentations sur des jeux de données réels et artificiels qui ont confirmé l'intérêt de l'approche.

Néanmoins, nous pensons que des progrès restent à faire du point de vue de la vitesse de la méthode. En effet, celle-ci nécessite de disposer de la matrice des distances entre les objets, qui se calcule en $\mathcal{O}(N^2)$. En outre, si la méthode de Louvain profite de matrices d'adjacence souvent creuses en ce qui concerne les graphes, nous sommes en revanche pénalisés quand il est question d'opérer sur les attributs et leur matrice de distance. Nous pensons qu'il est possible de concevoir une heuristique qui soit plus adaptée que celle de la méthode de Louvain à l'optimisation de la modularité basée sur l'inertie.

Sur un plan plus général, nous désirons trouver de nouvelles applications pour la modularité basée sur l'inertie, dans les domaines de la classification. Enfin, nous espérons que la proposition de ce critère permettra de mieux rapprocher les domaines de la classification automatique et de la détection de communautés.

ANNEXE A

# Comparaison des outils d'analyse de réseaux sociaux

Cette annexe reprend un rapport technique que nous avons réalisé en début de thèse sur des outils adaptés à notre problématique. Elle présente une étude comparative de logiciels et de bibliothèques de développement fournissant des méthodes d'analyse de réseaux qui a fait l'objet d'une communication dans la conférence WIVE - ProVE 2010 (Combe et al., 2010).

## A.1 Introduction

The explosion of Web 2.0 (blogs, wikis, content sharing sites, social networks, etc.) opens up new perspectives for sharing and managing information. In this context, among several emerging research fields related to "Web Intelligence", one of the most exciting are the applications specialized in the handling of the social dimension of the Web. In particular, building and managing communities require the development of a new generation of tools integrating social network mining and modeling features.

Several decades ago, the first studies on Social Network Analysis (SNA) were carried out by researchers in Social Sciences who wanted to understand the behavior of human networks (Wasserman et Faust, 1994a; Scott, 2000). Several indicators were proposed to characterize the actors as well as the network itself. One of them, for example, is the centrality, which can be viewed as a characterization of some kind of power within social networks, and thus can be used in viral marketing to discover the early adopters or the people whose activity is likely to spread information to many other people very quickly.

Nowadays, the wide use of Internet around the world allows a lot of people to connect. Facebook currently claims over 500 million active users [1] and according to Datamonitor they will be around one billion in 2012. As pointed out by the Gartner study, this very important development of the networks gives rise to a growing need

1. Facebook Factsheet page `http://www.facebook.com/press/info.php?statistics`

for social network mining and social network analysis methods (Gartner, 2008). SNA tools should provide some deeper functionalities in order to analyze networks and study their evolution, considering various applications as, for example, community marketing, social shopping, recommendation mechanisms, personalization filtering or alumni management.

For this reason, many new technologies (wikis, social bookmarks and social tagging, etc.) and services (GData [2], Google Friend Connect [3], OpenSocial [4], Open Graph protocol [5]. . .) have been developed and some of them include SNA tools.

These tools are very useful in analyzing a social network theoretically but also in representing it graphically. They compute several indicators which characterize the network's structure, the relationships between the actors as well as the position of a particular actor. They also allow the comparison of several networks.

The purpose of this stydy is to present some key cutting edge tools and to describe some of their functionalities. A similar study has already been done by *Huisman et al.*, but with a more statistical vision (Huisman et Van Duijn, 2003) while the one of (Xu et al., 2010) adopts a computational point of view with a focus on scalability in time and memory. Our survey targets professionals and researchers, who are not specialized in social mining and who need a rapid introduction in this field or to choose an SNA tool for a specific purpose.

Our comparative survey on the state-of-the-art tools for SNA is focused on three main points :

- Graph visualization ;
- Computation of various indicators providing a local or a global description (i.e. at the actor level or at the network level) ;
- Clustering and community detection.

In order to present the characteristics of the different tools, the main concepts used to represent social networks are defined in the next section. The different measures expected in a SNA tool are presented in section A.3. The benchmarking approach and the results of this comparative study are described in section A.4.

## A.2 Notations

The theoretical framework for social network analysis was introduced in the 1960s. Following the basic idea of Moreno who suggested representing agents by points

2. http://code.google.com/apis/gdata/
3. http://www.google.com/friendconnect/
4. http://code.google.com/apis/opensocial/
5. http://opengraphprotocol.org/

connected by lines, Cartwright and Harary proposed analyzing this sociogram using the graph theory (Moreno, 1934). For this reason, they are considered as the founders of the modern graph theory for social network analysis (Cartwright et Harary, 1977).

Two types of graphs can be defined to represent a social network : one-mode and two-mode graphs.

### A.2.1 One-mode graph

When the relationships between actors are considered, the social network can be represented by a graph $G = (V, E)$ where $V$ is the set of nodes associated with the actors , and $E \in V \times V$ is the set of edges which correspond to their relationships. This is the case, for instance in a classic dataset related to a karate club where the nodes correspond to the members of the club and where the edges are used to describe their friendships (Zachary, 1977). When the relationships are directed, edges are replaced by arcs. Nodes as well as edges can have attributes. In that case, the graph is labeled.

### A.2.2 Two-mode graph

When the relationships concern two types of elements, for example the members and the competitions in the karate club, a two-mode graph is most suited to represent the social network. A two-mode graph, also called a bipartite graph, is a graph with two types of nodes. The edges are allowed only between nodes with different types.

A two-mode graph can be transformed into a one-mode graph using a projection on one of the node types. Then, various aggregation functions can be used to create the edges between these nodes.

The most common way to store two-mode data is a rectangular data matrix with the two node types respectively in rows and columns. For example, a 2 dimensional matrix with the actors in rows and the events in columns can be used to represent the two-mode graph of the karate club. In this case, there is a link between an actor and an event if this actor participates to this event. One-mode graphs can be also stored as a 2 dimensional matrix, representing the nodes in rows and columns and having non null values for elements if the corresponding nodes are linked in the graph. This matrix is also called adjacency matrix. This representation is very common in SNA (Guillaume et Latapy, 2004). The concept of the graph can also be generalized by the concepts of hypergraph and multigraph. In a hypergraph, an edge can connect two set of nodes. A multigraph is a graph in which two nodes can be linked by several edges.

In the next sections, we note $|V|$ and $|E|$ the number of nodes and edges in G and $deg(v)$, the degree of the node $v$, defined as the number of adjacent edges to $v$.

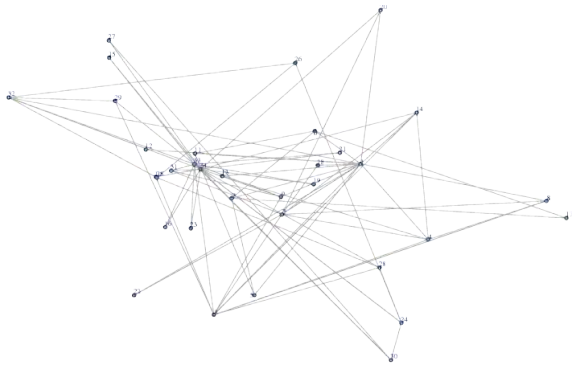


FIGURE A.1 – Visualization of Zachary's Karate club using the igraph library and spring layout

## A.3 Expected functionalities of network analysis tools

This work focuses on the different functionalities provided by network analysis tools. These functionalities are firstly the visualization of the network, secondly the computation of indicators based on nodes and on edges, and finally the community detection.

### A.3.1 Visualization

Visualization is one of the most wanted functionalities in general graph handling programs, and this remains true for network analysis software.

Many algorithms consist in pushing the isolated nodes toward empty spaces and in grouping adjacent nodes, following iterative rules. These algorithms, called force-based layouts, are directly inspired by physical phenomena. For example, edges can be seen as springs and nodes can be handled as electrically charged particles. The location of each element is recalculated step by step so these methods require several iterations in order to provide a good result on large graphs. Force-based layouts are simple to develop but are subject to poor local minimum results as Fig. A.1 shows.

Among these algorithms, we can mention, Fruchterman Reingold, which is a common force-based algorithm for graph visualization (Fruchterman et Reingold, 1991). It uses a grid system in order to limit the force-induced calculations ; the influence over a node is calculated only from nodes in the same cell. An example is provided in Fig. A.2.

An alternative taking advantage of approximation by Hooke's law is the algorithm of Kamada-Kawai (Kamada et Kawai, 1989) (see for example Fig. A.3), which has a

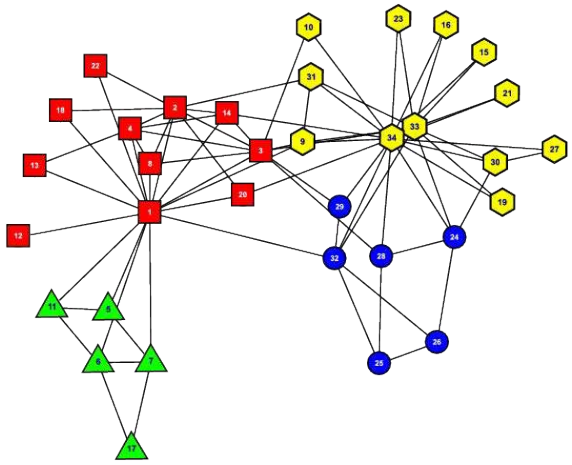


FIGURE A.2 – Community detection with igraph and the spinglass algorithm

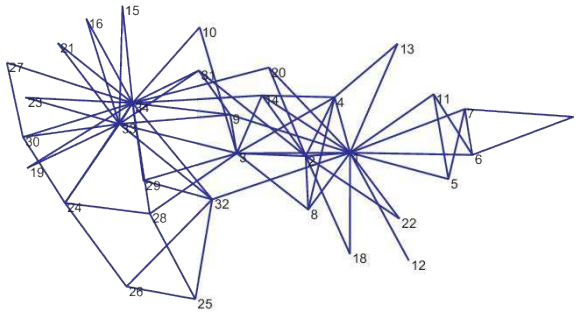


FIGURE A.3 – Visualization of Zachary's Karate club using the Pajek application and Kamada-Kawai layout

faster convergence than Fruchterman Reingold, but which often does not give such good results as the latter. For this reason, the use of Kamada-Kawai can be envisaged to calculate a first placement of the nodes. These two methods are among those called "spring algorithms".

Some other layouts are different in the way they provide a view of a node's neighborhood (i.e. radial layout, hyperbolic layout). The 3D graph visualization is the logical extension of planar representations. Most of the methods proposed are adaptable to 3D.

The radial layouts consist of representing the network around a particular node. Each node comes at a place where its distance to the central node is a function of the

shortest path between them.

Local zoom based, called fish-eye functionality, can prove to be fruitful in order to explore large graphs visually by focusing on a particular node or fragment (Gansner et al., 2005).

### A.3.2 Network description with indicators

Many quantitative indicators have been defined on networks (Wasserman et Faust, 1994a; Scott, 2000).

The descriptors at the network level are used to compare the proportion of nodes versus edges, or to evaluate the properties of the graph, like the small world or random distribution.

On the other hand, the descriptors at the node level are useful for detecting the nodes strategically placed in the network, highlighting those that play an important role in communication such as bridges or hubs.

#### A.3.2.1 Node and edge scoring

The location of a given actor in the network can be described using measures based on node scores. The centrality measures are among the most common scores. Within graph theory and network analysis, there are various measures of the centrality of a node to determine the relative importance of this node within the graph. For example, to measure how important a person is within a social network, Freeman has distinguished three main centralities (Freeman, 1979; Opsahl et al., 2010) :

a) Degree centrality : This is the first and simplest centrality measure. It emphasizes nodes with high degrees (Nieminen, 1974).

   The degree centrality of a node $v$ belonging to $V$ is defined by :

$$C_D(v) = \frac{deg(v)}{|V| - 1} \tag{A.1}$$

   where $deg(v)$, $v \in V$ is the number of neighbors of $v$.

b) Closeness centrality is the inverse of the average distance to all other nodes. For graphs with several components, closeness centrality equals $\infty$ in its simplest version. This indicator can be useful for many applications in the real world. For instance, if edges represent streets, the crossroad (node) with the highest closeness centrality would be the best place for emergency services.

c) Betweenness centrality is another centrality measure of a node within a graph. Nodes that lie on many shortest paths between the other nodes have a higher

betweenness (Freeman, 1979). An improved implementation of this indicator has been proposed by Ulrik Brandes with a running time of $O(|V|.|E|)$ (Brandes, 2001).

There is also another type of centrality measure : the eigenvector centrality which measures the importance of a node in a network (Klein, 2010; Taylor, 1969). It is based on the principle that the connections between a node and nodes having a high score must contribute more to the score of this node than its connections with nodes having a low score. These different measures can also be calculated on oriented graphs. For them, other measures can be defined like PageRank or HITS :

a) PageRank : The score computed by Page Rank is higher for nodes that are highly connected and connected with nodes that are highly connected themselves (Brin et Page, 1998). PageRank is a variant of the Eigenvector centrality measure.

b) HITS algorithm : Hyperlink-Induced Topic Search (also known as hubs and authorities) calculates two scores : hub and authority score (Kleinberg, 1999). The more a node has outgoing arcs to “authority” nodes, the higher its hub score is. The more a node has incoming links from “hub” nodes, the higher its authority score is.

#### A.3.2.2 Network scoring

Network density is the proportion of edges in the network relative to the total number of edges that could exist in the network. This measure shows if the underlying graph is sparse or dense (Scott, 2000).

These indicators have since been translated in versions applicable to directed graphs, useful in information dissemination theory. This asymmetry leads to the concept of prestige.

a) Dyad Census : A dyad is a term borrowed from sociology used to describe a group of two people, i.e. the smallest possible social group. By extension, it is used in social network analysis for designating two linked nodes. Four states are observable between two nodes ($a$ and $b$) for directed graphs : no arc, two mutual arcs, $a$ to $b$, $b$ to $a$. Each dyad is classified into one of these states and the proportion of each of these cases is computed. These values are useful to verify different hypotheses like random distribution or small-world (Holland et Leinhardt, 1970).

b) Triad Census : Davis and Leinhardt have also proposed the triad count, with 16 distinct cases (directed graphs) (Davis et Leinhardt, 1967). The triadic analysis performs the count of the triads in each configuration. The information provided is again useful for comparing a network with the random model.

#### A.3.2.3 Graph and node similarity

In SNA tools, one can expect to find functions giving the similarity of nodes in a graph and also functions to measure the similarity between graphs themselves. Some examples of similarity measures available in the programs are the Jaccard, Dice or Tanimoto similarities. However, the measure of similarity between graphs as well the as search of subgraph isomorphism remain open problems notably due to their algorithmic complexity.

### A.3.3 Clustering and community detection

The aim of clustering is to detect groups of nodes with dense connections within the groups and sparser connections between the groups. In social networks, these groups can be interpreted as communities composed of people sharing, for example, some common interests. We present a few community detection methods in the following section. A wider survey on graph clustering can be found in (Fortunato, 2009).

#### A.3.3.1 Main approaches of community detection

Among the different methods proposed to detect communities, two main approaches can be distinguished : on the one hand there is the hierarchical approach in which the nodes are aggregated in a hierarchy of clusters from the discrete partition to the whole network (Johnson, 1967). This approach evaluates the proximity between two nodes with a similarity measure and builds the groups using an agglomerative strategy like the single linkage or the complete linkage. On the other hand, there is the partitional clustering which consists in dividing the network directly into a predefined number of groups. The minimum cut method is an example of this approach in which the groups are defined so that the number of edges between them is minimized.
The programs considered in this benchmarking include three clustering methods. The first one is the Newman and Girvan method (Newman et Girvan, 2004). This is a hierarchical method, based on the betweenness of the edges, which consists in removing the edge with highest betweenness, and repeating this process until no edge remains.

The second method, called Walktrap, is based on an algorithm that uses a random walk in the graph in order to detect the components in which the walker tends to stay (Pons et Latapy, 2005). A distance between two nodes is calculated as the probability for a walker to go from a node to the other. A hierarchical clustering is then performed in order to obtain the clusters.

The last algorithm is called Spinglass (Reichardt et Bornholdt, 2006). Fig.A.2 shows an example of community detection done with the Spinglass algorithm of

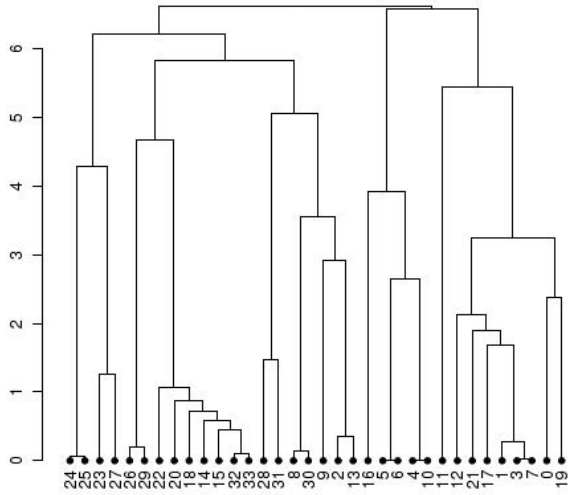



FIGURE A.4 – Dendrogram of the Walktrap algorithm results on the Zachary dataset (igraph website example)

igraph. In this figure, different node shapes indicate different communities.

With hierarchical methods, a dendrogram (cf. Fig. A.4) is the best representation for choosing the number of clusters to retain. Another way to determine the number of groups that should be retained consists in maximizing a particular criterion such as modularity.

#### A.3.3.2 Clustering validation

Modularity is a quality function useful to evaluate clustering. It was proposed by Newman and Girvan (Newman et Girvan, 2004). Modularity is defined by :

$$Q = \frac{1}{2.|E|} \sum_{(u,v) \in V \times V} (A_{uv} - P_{uv}) \delta(C_u, C_v) \tag{A.2}$$

where the couple $(u, v)$ runs over all pairs of nodes, $A$ is the adjacency matrix where $A_{uv}$ contains 1 if $u$ and $v$ are linked by an edge and 0 otherwise, $P_{uv}$ is the expected number of edges between $u$ and $v$, $C_v$ is the group to which node $v$ belongs and $\delta$ is the Kronecker delta, which is 1 if its two arguments are equal, and 0 otherwise. The clustering corresponding to a unique partition containing the whole graph has a modularity of zero.

## A.4 Benchmarking

Many tools have been created for network analysis. A list of dozens of tools is available on Wikipedia [6], with very different approaches. Many are purely academic programs. Some are oriented toward visualization ; others are APIs allowing graph modeling with sometimes the possibility of animation on nodes like JUNG. Some tools are optimized for large data manipulation. Others propose low level implementations of specific algorithms.

In this survey, we have studied more than ten tools (see section A.4.8) but we will detail four of them, two standalone programs and two libraries : Pajek, Gephi, igraph and NetworkX. The official documentation was inspected and experiments were carried out on datasets with the different for libraries and tools.

The following sections describe these tools, the criteria considered in the comparative study and the main datasets used in the experiments.

### A.4.1 Evaluated tools

We consider four tools, more precisely : Pajek, Gephi, igraph and NetworkX. The criteria for choosing these are based on :

- a balance between well established tools and newer ones based on recent development standards (in terms of ergonomics, modularity and data portability) ;
- a SNA point of view. The tools must provide basic indicators for network analysis ;
- the network size must reach tens of thousands of nodes.

Pajek is a *legacy* program, with its own graph-oriented approach. Gephi represents a modern answer for graph study with GUI (graphical user interface), with an open source philosophy and a plug-in orientation. NetworkX and igraph are two essential libraries for efficient handling of large graphs. The first one can be integrated easily into any framework. NetworkX is suitable for social network analysis in Python. igraph can be recommended for a *MATLAB-like* (console-based) approach. It can be used in a general purpose environment called *R*, which is dedicated to statistics. It is organized into many packages amongst which, some are dedicated to social network analysis. The functionalities covered are :

- *tnet* (Opsahl, 2009) : weighted, two-mode, and longitudinal network (network study over time) analysis,
- *statnet* (Goodreau et al., 2008) : statistical analysis of social networks,

6. `http://en.wikipedia.org/wiki/Social_network_analysis_software`

– *sna* (Butts, 2008) : node and graph-level indices, structural distance and covariance methods, structural equivalence detection, theoretical models fitting, random graph generation, and 2D/3D network visualization,
– *RSiena* (Snijders, 2006) : statistical model fitting for longitudinal graph-level study.

These packages are available on the Comprehensive R Archive Network [7]. *igraph* has also C and Python interfaces.

Among these tools, Gephi is the easiest to understand. It is clear that the two libraries are not as user friendly as the two other tools because they require programming. However, these libraries present other advantages for competent users, for example, the possibility to adapt or to insert their methods in their own software.

### A.4.2 Datasets

The dataset considered in this survey for illustration purposes and indicator evaluation is a dataset widely used in the SNA literature. This dataset, proposed by Zachary, presents the affiliation graph between 34 members of the karate club of a US university in 1970.

Zachary's Karate Club [8] has 34 nodes and 78 edges. Each node is numbered. An edge is present between two nodes when the two corresponding individuals "consistently interacted in contexts outside those of karate class, workouts and club meetings" (Zachary, 1977).

When larger graphs are needed, evaluating data load capacity for example, we have generated random graphs with $n$ nodes and $n \times 10$ edges with iGraph's graph generation feature. These graphs are exported in Pajek format, with sometimes syntactic adaptations for import in the other tools. The generated graphs are undirected ones.

### A.4.3 Evaluated criteria

In our comparative study, we have selected the following evaluation criteria : the license of the tool (c1), the capacity *i.e.* the number of nodes that can be loaded and treated in a reasonable time : (c2), the available indicators (c3), the handled file formats (c4), the supported graph types (c5), the available visualization layouts (c6), the included clustering algorithms (c7), and the custom attribute handling features (c8).

7. `http://cran.r-project.org/`
8. The dataset is available at `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm`, at the date of July 2011, 7th.

```
*Vertices      34
      1 "1"
      ...
      34 "34"
*Arcs
      1      2       1
      ...
      34     33       1
```

FIGURE A.5 – Zachary dataset extract in Pajek .net format

### A.4.4 File formats

There are three main ways to express the structure of a network in a serial manner :

- adjacency matrix (square for directed graphs, triangular for undirected ones) ;
- adjacency lists (for directed graphs), where the source node is followed by the list of the nodes that are the targets of every arc starting from the node ;
- node pairs.

Several file formats were created in order to provide graph representations. Here are the main ones :

a) Pajek graph file format (.net extension), while not very well documented, it is very popular in social network analysis tools. It represents first the nodes (one per line) and then the edges in a text file. An example is given in Fig. A.5.

   Weighted networks are allowed. The weights for the arcs can be given, in option, in the third column.

b) GML (Graph Modelling Language) is also a structured text file, where nodes and edges begin with the keywords "node" and "edge". Their content is given between square brackets. It allows annotations such as coordinates for nodes. Fig. A.6 presents an example.

   GML supports :
   - directed and undirected graphs,
   - node and edge labels,
   - graphical placement of nodes (coordinates),
   - other annotations.

c) GraphML is an XML-based graph description language as illustrated in Fig. A.7.

   As mentioned in its documentation [9], it supports :

9. http://graphml.graphdrawing.org

```
Creator "M. Newman on Fri Jul...2006"
graph [
  node [ id 1 ]
  ...
  node [ id 34 ]
  edge [
    source 2
    target 1
  ]
  ...
  edge [
    source 34
    target 33
  ]
]
```

FIGURE A.6 – Zachary dataset extract in GML format

- directed, undirected, and mixed graphs,
- hypergraphs,
- hierarchical graphs,
- graphical representations, and
- application-specific attribute data.

As with all XML-based representations, it is quite verbose.

d) DL (Data Language) format comes from the UCINet program (Borgatti et al., 2002). The common extension for this format is *.dat*. An example is given in Fig. A.8.

DL format supports :

- edge representation with a full matrix, a half-matrix, an arc list or an edge list,
- index labels,
- rectangular matrices for two-mode networks.

e) DOT is another popular graph description language, handled mainly by GraphViz (Ellson et al., 2001).

f) GEXF [10] is an XML-based format, from the GEXF Working Group. It supports

- dynamic graphs,
- application-specific attribute data, through the use of users XML namespaces,
- hierarchical structure (nodes can contain nodes)

10. `http://gexf.net`

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.
               graphdrawing.org/xmlns"
  ...
  <graph id="G"
          edgedefault="undirected">
    <node id="1"/>
    <node id="2"/>
    <edge id="e1" source="1"
                    target="2"/>
    ...
  </graph>
</graphml>
```

FIGURE A.7 – Zachary dataset extract in GraphML format

– visualization and positioning information such as 3D coordinates, colors, shapes.

As this is not a standard among network analysis software, this format will not be considered.

### A.4.5 Benchmarking results

The benchmarking results are summarized in the Table A.1.

They are detailed in this section, following the evaluation criteria introduced previously (see section A.4.3). The first criterion is licensing. It appears that NetworkX has the most permissive license, allowing integration in proprietary software. igraph and Gephi have chosen GNU GPL which does not allow the integration in proprietary software. Concerning Pajek, the source code is undisclosed and the use of the software for commercial use is not free.

Concerning the capacity in terms of the size of the handled graphs, the tool that appears the least efficient is Gephi. On our test environment, only 150,000 nodes are tractable. Moreover, memory lacking issues can appear when performing memory-intensive operations on large graphs. The visualization pane is an important part of Gephi. While the other tools can process indicators independently of drawing the graph, this is not the case for Gephi. Such architecture could penalize the application. Pajek does not suffer from this point and can load 500,000 nodes in 52 minutes. igraph is very fast for data loading (22 seconds for 2.9 millions of nodes for an attribute-free dataset). Gephi and NetworkX appear to be limited in their capacity by the RAM consumption. NetworkX is quite slow when loading 100,000 nodes, but the

```
DL
N=34 NM=2
FORMAT = FULLMATRIX DIAGONAL PRESENT
LEVEL LABELS:
ZACHE
ZACHC
DATA:
 0 1 1 1 1 1 1 1 ... 0 0 0 0 0 0 1 0 0
 1 0 1 1 0 0 0 1 ... 0 0 0 0 0 1 0 0 0
 1 1 0 1 0 0 0 1 ... 0 0 1 1 0 0 0 1 0
 ...
```

FIGURE A.8 – Zachary dataset in DAT format

loading time stays reasonable beyond this number. Some features, such as multigraph management, can be the cause of the diminished performance.

The four tools are suitable for computing common indicators, such as graph statistics, degree centrality, closeness centrality and betweenness centrality (igraph and NetworkX implementations of betweenness centrality are based on the algorithm from Brandes (Brandes, 2001)). Dyad and triad census are available in igraph and Pajek (for triad census). For HITS and PageRank indexes, Pajek cannot be relied upon as it is not up to date. If one needs to create his own indicators, the two libraries and Gephi are extensible.

In the matter of data format, Gephi handles all the formats mentioned previously. GEXF is not available elsewhere mainly because this format started in the Gephi project. DL comes with UCINet ; the latter being a project linked to Pajek, it is one of the preferred formats for this tool. GML and GraphML are not supported in Pajek. For this reason, one would prefer the .net format, which is universal in our panel.

Concerning the bipartite graph study and their manipulation, most tools propose few primitives, such as projection (conversion of a bipartite graph into a one-mode graph), but we would not recommend Gephi on this point. Pajek can handle links of different kinds. Temporality starts being taken into account in different projects (e.g. in Gephi, cf. A.4.6). For now, the data can be filtered according to a year associated to the nodes for example, if the data format is adapted.

Concerning visualization layouts, NetworkX lacks basic algorithms. If advanced visualization is needed, you should switch your data to another platform. The three other tools include the popular force-based algorithms : Fruchterman Reingold and Kamada Kawai. The best tool for visualization is Gephi.

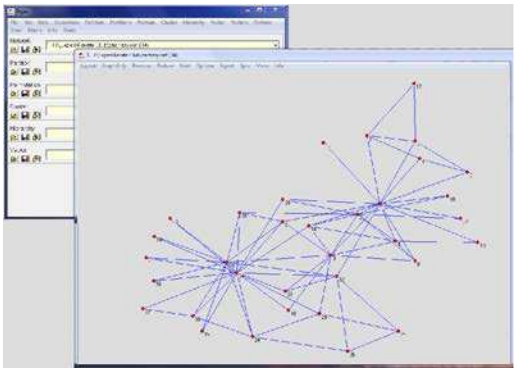


FIGURE A.9 – Pajek snapshot

Community detection is experimental in Gephi with a beta version of Markov cluster algorithm (MCL) while few algorithms are available in igraph. Pajek offers hierarchical clustering capabilities. It can provide a dendrogram representation of hierarchical clustering, as an EPS (Encapsulated PostScript) image. igraph offers the dendrogram plotting capabilities of R. Gephi, Pajek and igraph give a dendrogram representation for the communities obtained.

Depending on the tool, the possibility of handling extra attributes affected to nodes or edges is more or less easy. While libraries require skills based on the programming language used, the standalone programs can fix limits on the type of the data or on the file formats to use.

### A.4.6 Overview per tool

#### A.4.6.1 Pajek

The development of Pajek started in 1996. This program is both a reference and quite different from newer tools (see Fig. A.9). It appears as mature even if it suffers from deficiencies in recent metrics (e.g. PageRank, cf. A.4.5) and file formats for network analysis (e.g. GML and GraphML, cf. A.4.5). The fact that Pajek is a closed-source software is a problem in the academic domain. It is a fast tool and comfortable for visualization purposes. It is not as extensible as the three other studied tools. Nevertheless, Pajek is useful in hierarchical data manipulation and it provides powerful and accessible data manipulation functions. The 3D visualization and its export to VRML are also available. Macro programming is possible, which enables automation of tasks over networks. Moreover, many graph operators are included. Finally, the network data can be represented as edge list, arc list or adjacency matrix.

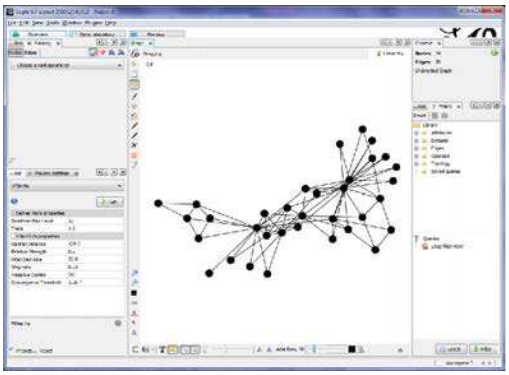


FIGURE A.10 – Gephi snapshot

#### A.4.6.2 Gephi

Gephi (see Fig. A.10) is quite a new tool (its development started in 2008) and it is updated frequently. Many functionalities are already supported, but several algorithms are missing. It has a user friendly graphical user interface. The rendering is highly customizable and quite fast. It is possible to move nodes while rendering which makes Gephi even more interactive. The ability to export metric results as spreadsheets would help. At the time of writing, community detection is still experimental. In the case of very large networks (exceeding 150,000 nodes), one should run Pajek or a library. The most interesting point in Gephi seems to be the extensibility through plug-ins for creating metrics or import/export capabilities for a new file formats. The release of the Gephi Toolkit [11] offers the ability to use the graph management and metric functions in a Java program. The operations on timed graph elements are possible, such as filtering nodes depending on an attached date. In addition, interactive force-based layouts and temporality awareness turns Gephi ready for graph streaming capacities, i.e. study of dynamic graphs through a temporal dimension.

#### A.4.6.3 NetworkX

NetworkX is well documented and makes graph manipulation easy in Python but community detection algorithms are lacking. The first public version of the library was published in 2005. It is possible to use any hashable Python object as node and edge, which makes the integration of the library easy and elegant. NetworkX includes also some useful functions for bipartite graph manipulation. NetworkX is an interesting tool if the manipulated nodes and / or edges are complex objects. It is not suited for

11. http://gephi.org/toolkit/

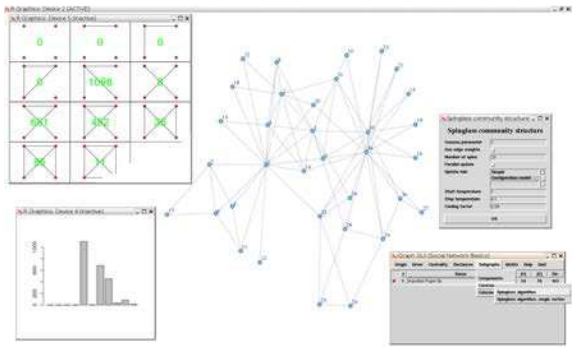


FIGURE A.11 – tkigraph user interface for igraph

the visualization. It offers the ability to handle over 1,000,000 node graphs.

#### A.4.6.4 igraph

igraph (see Fig. A.11) offers many algorithms among which some are clustering oriented. The first release of igraph has been published in 2006. It is available for Python, R and C environments. With R, it is easy to integrate igraph routines in a statistical process. A graphical user interface offers easy visualization. igraph is performance-oriented and most of its functionalities are implemented in C. 3D visualization layouts are available. It offers some node-related neighborhood similarities such as Jaccard, Dice and the inverse log-weighted similarities (Adamic et Adar, 2003). This is the only tool in our study that provides really useful community detection capabilities. igraph is able to load a large number of nodes and edges but handling custom attributes on the elements of a graph requires the use of the underlying programing language.

### A.4.7 Software matching special interests

Table A.2 summarizes the Table A.1 following our eight key criteria introduced previously (see section A.4.3). We give an evaluation, on a scale from *Weak or unavailable (- -)* to *Mature functionality (++)*.

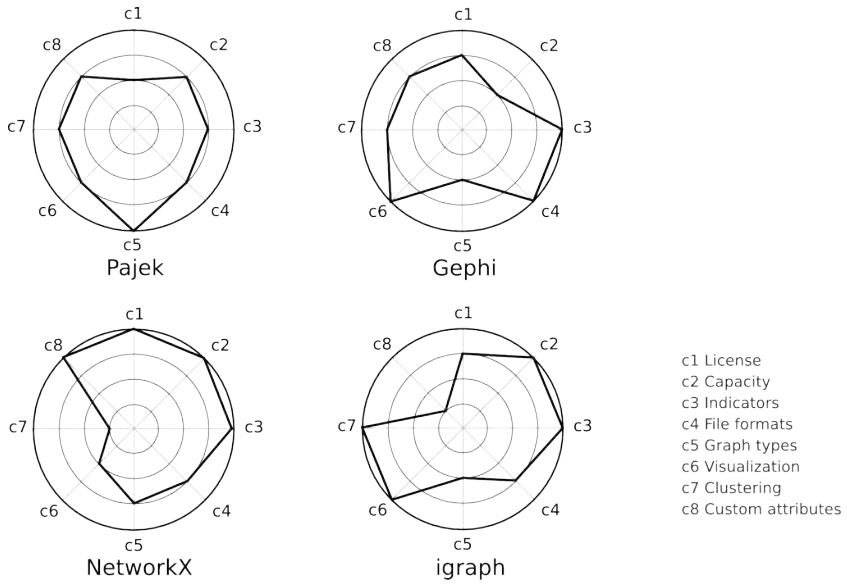



FIGURE A.12 – Radar view of some key criteria for choosing social network analysis tools

| Criteria | | Pajek (Batagelj et Mrvar, 1998) | Gephi (Bastian et al., 2009) | NetworkX (Hagberg et al., 2008) | igraph (Csárdi et Nepusz, 2006) |
|---|---|---|---|---|---|
| c1 | License | – | + | ++ | + |
| c2 | Capacity | + | – | ++ | ++ |
| c3 | Indicators | + | ++ | ++ | ++ |
| c4 | File formats | + | ++ | + | + |
| c5 | Graph types | ++ | – | + | – |
| c6 | Visualization | + | ++ | – | ++ |
| c7 | Clustering | + | + | – – | ++ |
| c8 | Custom attributes | + | + | ++ | – – |

TABLE A.2 – Criteria evaluated from unavailable or weak (– –) to mature (++)

The graphical representation is visible in Fig. A.12.

One of the main questions that a person interested in SNA might ask is : "How to choose the right tool ?". For standard graph visualization, it is likely that any user could find software to suit his needs. If the data is not in one of the standardized formats given in the list above, the best way to begin is to generate a suitable representation from a memory-loaded graph or to convert it into GML for example. One can also refer to this Wikipedia page [12] which presents a list of input/output formats allowed

12. http://en.wikipedia.org/wiki/Social_network_analysis_software

by a large panel of programs. For specific needs or for particular types of nodes and edges, libraries are often mandatory. In order to choose a program for visualizing or manipulating graphs, it is recommended to try a few of them to check if they fit the problem. For libraries, the choice depends on the user's favorite languages. The computation time can also be an important criterion for the choice ; if this is the case, one should prefer software based on Python or on C. People interested by an interactive console environment (as the MATLAB experience) should definitely select igraph on R.

### A.4.8 Other interesting tools for social network analysis

There are many other SNA tools available, we studied and tested some of them such as :

- GraphViz (Ellson et al., 2001) is dedicated to graph visualization. While it is not focused on social networks but usable for some other kinds of graphs, in bioinformatics or engineering, this tool is a reference in graph drawing.
- Tulip (Auber, 2004) can handle over 1 million nodes and 4 million edges. It has visualization and clustering functionalities and capabilities of extension by plug-ins.
- UCINet (Borgatti et al., 2002) is not free. It uses Pajek and Netdraw for visualization. It is specialized in statistical analysis. It calculates indicators (such as triad census or Freeman betweenness) and performs hierarchical clustering.
- JUNG (O'Madadhain et al., 2005), for Java Universal Network/Graph Framework, is a library mainly developed for creating interactive graphs in Java user interfaces. JUNG has been extended with some SNA metrics.
- GUESS (Adar, 2006) is dedicated to visualization purposes and is powered by many different layouts. It is published under the GPL license.
- NodeXL (Smith et al., 2009) is a template for Microsoft Excel which provides advanced visualization and analysis capabilities for graphs right in the spreadsheet program.

We don't present in details these tools, because :

- They focus on a narrow functionalities (i.e. GUESS on visualization).
- Some tools have similar features and target audience and while interesting do not bring an experience to the user different enough from the ones described in this paper (i.e. Tulip with Gephi).
- Some tools may be too far from the computer science oriented vision of the authors.
- And finally the reason for putting aside a tool might be that it is not freely

available.

## A.5 Conclusion

The fact that Social Network Analysis is situated between several domains (sociology, computer science, mathematics and physics) has led to many different methodological approaches. That is why so many programs have been created in order to manipulate and study social networks[11]. While a standalone tool is very useful for graph visualization (up to a maximum of few thousands of nodes), data format conversion or indicators computation, libraries are more adapted for tasks involving tens of thousands of nodes and for operations such as the union and the difference between sets of nodes or for the clustering. A fair separation of the algorithms, the user interface (including the visualization capabilities) and data manipulation appears as an important task in order to gather and promote best functionalities among the tools. For example, lately, Gephi started such an approach with the recent release of the *Gephi toolkit*, a library created from the Gephi logic and algorithms.

We can also say that today, the freely available tools are able to provide a very rich set of functionalities, but if specific analysis is required, specific code developments may be needed.

Concerning the need to store larger networks, new architectures were recently proposed notably the NoSQL databases, like Neo4j [13] that can be used to store data structured in graphs rather than in tables.

Finally at this point in time, the main challenges concerning graph exploration are oriented toward high-level visualization notably for hierarchical graphs. Amongst the possible enhancements of social network analysis tools, we can mention firstly social mining which simultaneously exploits node attributes and graph structure and secondly temporal analysis which should allow us to study the evolution of networks over time (see for instance [14] (Snijders, 2006)).

13. http://neo4j.org/
14. http://www.stats.ox.ac.uk/~snijders/siena/

| **Tool** | Pajek (Batagelj et Mrvar, 1998) | Gephi (Bastian et al., 2009) | NetworkX (Hagberg et al., 2008) | igraph (Csárdi et Nepusz, 2006) |
|---|---|---|---|---|
| Version | 1.26 | 0.8 alpha | 0.6 | 0.5.4 |
| Website (http ://) | pajek.imfm.si/doku.php | gephi.org | networkx.lanl.gov/ | igraph.sourceforge.net |
| Type | Standalone program | Standalone program | Library | Library |
| Platform | Windows | Java | Python | R / Python / C libraries |
| License | Free for non-commercial use | GNU GPL | BSD License | GNU GPL |
| Computing time | Fast (C) | Medium (Java) | Fast (C, Python) | Fast (C) |
| Tractable number of nodes | 500,000 nodes | 150,000 nodes | 1,000,000 nodes | > 1.9 million relations (without attributes) |
| Time to load $10^5$ nodes and $10^6$ edges | 24 seconds | 40 seconds | 137 seconds | 11 seconds |
| **Ergonomics** | | | | |
| Ease of handling | Good | Good | Good (but Python handling needed) | Average (R handling recommended) |
| **File formats** | | | | |
| GML | X | ✓ | ✓ | ✓ |
| Pajek (.net) | ✓ | Import only | ✓ | ✓ |
| GraphML | Export only | ✓ | ✓ | ✓ |
| DL | ✓ | ✓ | X | X |
| **Graph types** | | | | |
| Two-mode graphs | ✓ | X | ✓ | ✓ |
| Multi-relational graphs | ✓ | X | X | X |
| Temporality | ✓ | ✓ | ✓ | X |
| **Visualization layouts** | | | | |
| Fruchterman Reingold | ✓ | ✓ | X | ✓ |
| Kamada Kawai | ✓ | ✓ | X | ✓ |
| Other spring layouts | X | ✓ | ✓ | ✓ |
| **Indicators** | | | | |
| Degree, betweenness and closeness centrality | ✓ | ✓ | ✓ | ✓ |
| Dyad census | X | X | X | ✓ |
| Triad census | ✓ | X | X | ✓ |
| HITS, PageRank | X | ✓ | ✓ | ✓ |
| **Clustering algorithms** | | | | |
| Edge betweenness | X | X | X | ✓ |
| Walktrap | X | X | X | ✓ |
| Spinglass | X | X | X | ✓ |
| Dendrogram display | ✓ | ✓ | X | ✓ |

TABLE A.1 – Features of the main algorithms in the retained tools

# Publications

COMBE, D., LARGERON, C., EGYED-ZSIGMOND, E., & GÉRY, M. (2013). TeM : une méthode de détection de communautés adaptée aux réseaux d'information. In *Extraction et gestion des connaissances (EGC 2013)* (pp. 305–310).

COMBE, D., LARGERON, C., EGYED-ZSIGMOND, E., & GÉRY, M. (2012a). Détection de communautés dans des réseaux scientifiques à partir de données relationnelles et textuelles. In *4ième conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques (MARAMI)*.

COMBE, D., LARGERON, C., EGYED-ZSIGMOND, E., & GÉRY, M. (2012b). Combining relations and text in scientific network clustering. In *First International Workshop on Semantic Social Network Analyis and Design at IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (pp. 1280–1285).

COMBE, D., LARGERON, C., EGYED-ZSIGMOND, E., & GÉRY, M. (2012c). Getting clusters from structure data and attribute data. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (pp. 710–712).

COMBE, D., LARGERON, C., EGYED-ZSIGMOND, E., & GÉRY, M. (2010). A comparative study of social network analysis tools. In *Web Intelligence Virtual Enterprise 2010* (Vol. 2).

# Bibliographie


Adamic, L. et E. Adar (2003). Friends and neighbors on the web. *Social Networks 25*(3), pp. 211–230.

Adamic, L. A. et N. Glance (2005). The political blogosphere and the 2004 U.S. election. Dans *Proceedings of the 3rd international workshop on Link discovery - LinkKDD*, New York, New York, USA, pp. 36–43. ACM Press.

Adar, E. (2006). Guess : a language and interface for graph exploration. Dans *SIGCHI conference on Human Factors in computing systems*, pp. 781–800. ACM.

Albert, R. et A. L. Barabási (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics 74*(1), pp. 47–97.

Almeida, H., D. Guedes, W. Meira Jr, et M. J. Zaki (2011). Is there a best quality metric for graph clusters ? Dans *Machine Learning and Knowledge Discovery in Databases*, pp. 44–59. Springer.

Anagnostopoulos, A., R. Kumar, et M. Mahdian (2008). Influence and correlation in social networks. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 10*(1), pp. 7.

Arenas, A., A. Fernández, et S. Gómez (2008). Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics 10*(5), pp. 053039.

Artignan, G. et M. Hascoët (2011). Analyse Visuelle d'Algorithmes de Clustering. Une méthodologie et un cas d'étude. Dans *Journée Fouille de Grands Graphes*.

Auber, D. (2004). *Tulip : A huge graph visualisation framework*. P. Mutzel and M. Junger.

Aynaud, T., V. Blondel, J.-L. Guillaume, et R. Lambiotte (2010). Optimisation locale multi-niveaux de la modularité. Dans Charles-Edmond Bichot et P. Siarry (Eds.), *Partitionnement de graphe : optimisation et applications*, Chapter 13, pp. 389–422. Hermes-Lavoisier.

Aynaud, T. et J.-L. Guillaume (2011). Multi-Step Community Detection and Hierarchical Time Segmentation in Evolving Networks. Dans *Proceedings of the 5th SNA-KDD Workshop*.

Baker, F. et L. Hubert (1975). Measuring the power of hierarchical cluster analysis. *Journal of the American Statistical Association 70*(349), pp. 31–38.

Banerjee, A., C. Krumpelman, J. Ghosh, S. Basu, et R. J. Mooney (2005). Model-based overlapping clustering. Dans *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, New York, NY, USA, pp. 532–537. ACM.

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

Bastian, M., S. Heymann, et M. Jacomy (2009). Gephi : An Open Source Software for Exploring and Manipulating Networks. Dans *International AAAI Conference on Weblogs and Social Media*, pp. 361–362.

Batagelj, V. et A. Mrvar (1998). Pajek-program for large network analysis. *Connections 21*(2), pp. 47–57.

Baumes, J., M. Goldberg, et M. Magdon-Ismail (2005). Efficient identification of overlapping communities. Dans *IEEE international conference on Intelligence and Security Informatics*, pp. 27–36.

Blondel, V. D., J.-L. Guillaume, R. Lambiotte, et E. Lefebvre (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics : Theory and Experiment 10 (P10008*.

Borgatti, S. (2005). Centrality and network flow. *Social Networks 27*(1), pp. 55–71.

Borgatti, S., M. Everett, et L. Freeman (2002). Ucinet for Windows : Software for social network analysis.

Brandes, U. (2001). A Faster algorithm for betweenness centrality. *Journal of Mathematical Sociology 25*, pp. 163–177.

Brandes, U. (2008). On Variants of Shortest-Path Betweenness Centrality and their Generic Computation. *Social Networks 30*(2), pp. 136–145.

Brandes, U., D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, et D. Wagner (2007). On Finding Graph Clusterings with Maximum Modularity. Dans *International Workshop on Graph-Theoretic Concepts in Computer Science*, pp. 121–132. Lecture Notes in Computer Science (LNCS).

Brandes, U., M. Gaertler, et D. Wagner (2003). Experiments on Graph Clustering Algorithms. Dans *11th European Symposium on Algorithms*, pp. 568–579. Springer.

Brandes, U., M. Gaertler, et D. Wagner (2007). Engineering graph clustering : Models and experimental evaluation. *ACM Journal of Experimental Algorithmics 12*(1.1), pp. 1–26.

Brin, S. et L. Page (1998). The anatomy of a large-scale hypertextual Web search engine. Dans *Seventh International World-Wide Web Conference (WWW)*.

Butts, C. (2008). Social network analysis with sna. *Journal of Statistical Software 24*(6), pp. 1–51.

Cailliez, F., J. Pages, G. Morlat, et J. Amiard (1976). *Introduction à l'analyse des données*. Société de mathématiques appliquées et de sciences humaines.

Calinski, T. et J. Harabasz (1974). A dendrite method for cluster analysis. *Communications in Statistics Theory and Methods 3*(1), pp. 1–27.

Cartwright, D. et F. Harary (1977). A graph theoretic approach to the investigation of system-environment relationships. *Journal of Mathematical Sociology 5*, pp. 87–111.

Chan, P. K., M. D. F. Schlag, et J. Y. Zien (1994). Spectral K-way ratio-cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 13*(9), pp. 1088–1096.

Clauset, A., M. Newman, et C. Moore (2004). Finding community structure in very large networks. *Physical Review E 70*(6), pp. 1–6.

Collingsworth, B. et R. Menezes (2013). A Self-organized Approach for Detecting Communities in Networks. Dans G. Fortino, C. Badica, M. Malgeri, et R. Unland (Eds.), *Intelligent Distributed Computing VI*, Volume 446 de *Studies in Computational Intelligence*, pp. 29–39. Berlin, Heidelberg : Springer Berlin Heidelberg.

Combe, D., C. Largeron, E. Egyed-Zsigmond, et M. Géry (2010). A comparative study of social network analysis tools. Dans *Web Intelligence Virtual Enterprise 2010*, Volume 2, pp. 1–12.

Combe, D., C. Largeron, E. Egyed-Zsigmond, et M. Géry (2012a). Détection de communautés dans des réseaux scientifiques à partir de données relationnelles et textuelles. Dans *4ième conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques (MARAMI)*.

Combe, D., C. Largeron, E. Egyed-Zsigmond, et M. Géry (2012b). Getting clusters from structure data and attribute data. Dans *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 710–712.

Crandall, D., D. Cosley, D. Huttenlocher, J. Kleinberg, et S. Suri (2008). Feedback effects between similarity and social influence in online communities. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 363*(3), pp. 160.

Cruz-Gomez, J. D. (2012). *Socio-semantic networks algorithm for a point of view based visualization of on-line communities*. Thèse de doctorat, Télécom Bretagne, Université de Rennes 1.

Cruz-Gomez, J. D., C. Bothorel, et F. Poulet (2011). Entropy based community detection in augmented social networks. Dans *CASoN 2011*, pp. 163–168.

Csárdi, G. et T. Nepusz (2006). The igraph software package for complex network research. *InterJournal Complex Systems 1695*.

Dang, T. A. (2012). *Analysis of communities in social networks*. Thèse de doctorat, Université Paris 13.

Dang, T. A. et E. Viennet (2012). Community Detection based on Structural and Attribute Similarities. Dans *International Conference on Digital Society (ICDS)*, pp. 7–14.

Davies, D. et D. Bouldin (1979). A cluster separation measure. *Pattern Analysis and Machine Intelligence* (2), pp. 224–227.

Davis, J. A. et S. Leinhardt (1967). The Structure of Positive Interpersonal Relations in Small Groups. *Sociological Theories in Progress 2*.

Diday, E. (1971). Une nouvelle méthode en classification automatique et reconnaissance des formes : la méthode des nuées dynamiques. *Revue de statistique appliquée 19*(2), pp. 19–33.

Ding, C., X. He, H. Zha, et M. Gu (2001). A min-max cut algorithm for graph partitioning and data clustering. Dans *Proceedings IEEE International Conf on Data Mining*, pp. 107–114.

Djidjev, H. N. (2008). A Scalable Multilevel Algorithm for Graph Clustering and Community Structure Detection. *Algorithms and Models for the WebGraph 4936*, pp. 117–128.

Duda, R. O. et P. E. Hart (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*.

Elghazel, H. (2007). *Classification et Prévision des Données Hétérogènes : Hospitaliers, Application aux Trajectoires et Séjours*. Thèse de doctorat, Lyon 1.

Ellson, J., E. Gansner, L. Koutsofios, S. North, et G. Woodhull (2001). Graphviz - open source graph drawing tools. Dans *Graph Drawing*, pp. 594–597. Springer.

Erdős, P. et A. Rényi (1959). On random graphs. *Publicationes Mathematicae Debrecen 6*, pp. 290–297.

Ester, M., R. Ge, B. Gao, Z. Hu, et B. Ben-Moshe (2006). Joint Cluster Analysis of Attribute Data and Relationship Data : the Connected k-Center Problem. Dans *SIAM International Conference on Data Mining*, pp. 25–46. ACM Press.

Flake, G., R. Tarjan, et K. Tsioutsiouliklis (2003). Graph clustering and minimum cut trees. *Internet Mathematics 1*(4), pp. 385–408.

Forestier, G., C. Wemmert, et P. Gançarski (2010). Comparaison de critères de pureté pour l'intégration de connaissances en clustering semi-supervisé. Dans *Extraction et gestion des connaissances (EGC)*, pp. 127–132.

Forgy, E. W. (1965). Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics 21*, pp. 768–769.

Fortunato, S. (2009). Community detection in graphs. *Physics Reports 486*(3-5), pp. 75–174.

Fortunato, S. et M. Barthélemy (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences of the United States of America 104*(1), pp. 36–41.

Freeman, L. C. (1979). Centrality in social networks conceptual clarification. *Social networks 1*(3), pp. 215–239.

Fruchterman, T. M. J. et E. M. Reingold (1991, November). Graph drawing by force-directed placement. *Software : Practice and Experience 21*(11), pp. 1129–1164.

Gansner, E., Y. Koren, et S. North (2005). Topological fisheye views for visualizing large graphs. *IEEE Transactions on Visualization and Computer Graphics 11*(4), pp. 457–468.

Gartner (2008). Hype Cycle for social software, 2008. *G00158239*.

Ge, R., M. Ester, B. J. Gao, Z. Hu, B. Bhattacharya, et B. Ben-Moshe (2008). Joint cluster analysis of attribute data and relationship data. *ACM Transactions on Knowledge Discovery from Data 2*(2), pp. 1–35.

Ghaemi, R., M. Sulaiman, H. Ibrahim, et N. Mustapha (2009). A survey : clustering ensembles techniques. *Engineering and Technology (2009) 38*, pp. 636–645.

Ghosh, J., A. Strehl, et S. Merugu (2002). A consensus framework for integrating distributed clusterings under limited knowledge sharing. Dans *Proc. NSF Workshop on Next Generation Data Mining*, pp. 99–108. Citeseer.

Gong, N. Z., A. Talwalkar, L. Mackey, L. Huang, E. C. R. Shin, E. Stefanov, Elaine, Shi, et D. Song (2012). Jointly Predicting Links and Inferring Attributes using a Social-Attribute Network (SAN). Dans *ACM Workshop on Social Network Mining and Analysis (SNA-KDD)*, pp. 9.

Goodreau, S., M. Handcock, D. Hunter, et C. Butts (2008). A statnet Tutorial. *Journal of statistical software 24*(9), pp. 1–26.

Greene, D., D. Doyle, et P. Cunningham (2010). Tracking the Evolution of Communities in Dynamic Social Networks. Dans *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 1236–1241.

Guillaume, J.-L. et M. Latapy (2004). Bipartite structure of all complex networks. *Information Processing Letters 90*(5), pp. 215–221.

Hagberg, A., D. Schult, et P. Swart (2008). Exploring network structure, dynamics, and function using NetworkX. Dans *Python in Science Conference*, pp. 11–15.

Holland, P. et S. Leinhardt (1970). A method for detecting structure in sociometric data. *American Journal of Sociology*.

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

Huisman, M. et M. Van Duijn (2003). Software for social network analysis. Dans *Models and methods in social network analysis*, Chapter Daatset, pp. 270–316. Cambridge University Press.

Johnson, S. C. (1967, September). Hierarchical clustering schemes. *Psychometrika 32*(3), pp. 241–254.

Kamada, T. et S. Kawai (1989). An algorithm for drawing general undirected graphs. *Information processing letters 31*(12), pp. 7–15.

Kantardzic, M. (2011). *Data Mining : Concepts, Models, Methods, and Algorithms*. John Wiley & Sons.

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

Keller, I. et E. Viennet (2012). Caractérisation de la structure communautaire d'un grand réseau social. Dans *3ième conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques (MARAMI)*.

Kernighan, B. W. et S. Lin (1970). An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal 49*(2), pp. 291–307.

Kim, M. et J. Leskovec (2010). Multiplicative attribute graph model of real-world networks. Dans *7th Workshop on Algorithms and Models for the Web Graph*.

Kim, M. et J. Leskovec (2011). Modeling social networks with node attributes using the multiplicative attribute graph model. Dans *Uncertainty in Artificial Intelligence*.

Klein, D. (2010). Centrality measure in graphs. *Journal of Mathematical Chemistry 47*, pp. 1209–1223.

Kleinberg, J. M. (1999, September). Authoritative sources in a hyperlinked environment. *Journal of the ACM 46*(5), pp. 604–632.

Koschützki, D., K. A. Lehmann, D. Tenfelde-Podehl, et O. Zlotowski (2005). Advanced Centrality Concepts. Dans *Network analysis*, Chapter 5, pp. 83–111. Springer Berlin Heidelberg.

Krings, G. et V. D. Blondel (2011). An upper bound on community size in scalable community detection. Dans *arXiv preprint arXiv :1103.5569*.

Labatut, V. (2012). Une nouvelle mesure pour l'évaluation des méthodes de détection de communautés. Dans *Conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques (MARAMI)*.

Lancichinetti, A. et S. Fortunato (2009). Community detection algorithms : a comparative analysis. *Physical review E 80*(5), pp. 056117.

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

Lancichinetti, A., S. Fortunato, et J. Kertész (2009). Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics 033015 - 1*(3).

Lazarsfeld, P. F. et R. K. Merton (1954). Friendship as a social process : A substantive and methodological analysis. Dans M. Berger, T. Abel, et C. H. Page (Eds.), *Freedom and Control in Modern Society*, Volume 18, pp. 18–66. Van Nostrand.

Le Martelot, E. et C. Hankin (2013, January). Fast Multi-Scale Detection of Relevant Communities in Large-Scale Networks. *The Computer Journal 56*(9), pp. 1136–1150.

Leskovec, J., K. Lang, A. Dasgupta, et M. Mahoney (2008). Statistical Properties of Community Structure in Large Social and Information Networks. *Proceeding of the 17th international conference on World Wide Web (WWW) 7*(3), pp. 695.

Leskovec, J., K. J. Lang, et M. Mahoney (2010). Empirical comparison of algorithms for network community detection. Dans *Proceedings of the 19th international conference on World wide web - WWW*, pp. 631–640. ACM Press.

Liu, X., T. Murata, et K. Wakita (2013). Extending modularity by capturing the similarity attraction feature in the null model. Dans *Proceedings of the 22nd international conference on World Wide Web*, pp. 191–192.

Loper, E. et S. Bird (2002). NLTK : The natural language toolkit. Dans *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, pp. 63–70. Association for Computational Linguistics.

Lu, Q. et L. Getoor (2003). Link-based Classification. Dans *International Conference on Machine Learning (ICML)*, pp. 496–503.

Luo, C., Y. Li, et S. Chung (2009). Text document clustering based on neighbors. *Data & Knowledge Engineering 68*(11), pp. 1271–1288.

McPherson, M., L. Smith-Lovin, et J. M. Cook (2001). Birds of a feather : Homophily in Social Networks. *Annual review of sociology*, pp. 415–444.

Meilă, M. (2007, May). Comparing clusterings—an information based distance. *Journal of Multivariate Analysis 98*(5), pp. 873–895.

Milgram, S. (1967). The small world problem. *Psychology today 2*(1), pp. 60–67.

Milligan, G. et M. Cooper (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika 50*(2), pp. 159–179.

Montgolfier, F. D., M. Soto, et L. Viennot (2012). Asymptotic Modularity of some Graph Classes. Dans *Proceedings of the 22nd international conference on Algorithms and Computation (ISAAC)*, pp. 435–444.

Moreno, J. (1934). *Who shall survive ?* New York : Beacon Press.

Moser, F., R. Ge, et M. Ester (2007). Joint Cluster Analysis of Attribute and Relationship Data Without A-Priori Specification of the Number of Clusters. Dans *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 510–519.

Neville, J., M. Adler, et D. Jensen (2003). Clustering relational data using attribute and link information. Dans *Proceedings of the text mining and link analysis workshop, 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 9–15.

Newman, M. (2004). Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems 38*(2), pp. 321–330.

Newman, M. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences 103*(23), pp. 8577–8582.

Newman, M. et M. Girvan (2004). Finding and evaluating community structure in networks. *Physical review E 69*(2), pp. 1–16.

Nicosia, V., G. Mangioni, V. Carchiolo, et M. Malgeri (2009). Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics : Theory and Experiment* (03-P03024).

Nieminen, J. (1974). On the centrality in a graph. *Scandinavian Journal of Psychology 15*(1), pp. 332–336.

Noack, A. et R. Rotta (2009). Multi-level Algorithms for Modularity Clustering. Dans *Experimental Algorithms*, pp. 257–268. Springer.

O'Madadhain, J., D. Fisher, P. Smyth, S. White, et Y.-B. Boey (2005). Analysis and visualization of network data using JUNG. *Journal of Statistical Software 10*(2), pp. 1–35.

Opsahl, T. (2009). *Structure and Evolution of Weighted Networks*. Thèse de doctorat, University of London.

Opsahl, T., F. Agneessens, et J. Skvoretz (2010). Node centrality in weighted networks : Generalizing degree and shortest paths. *Social Networks 32*(3), pp. 245–251.

Page, L., S. Brin, R. Motwani, et T. Winograd (1999). The PageRank Citation Ranking : Bringing Order to the Web. *World Wide Web Internet And Web Information Systems*, pp. 1–17.

Pelleg, D. et A. Moore (2000). X-means : Extending K-means with Efficient Estimation of the Number of Clusters. Dans *International Conference on Machine Learning (ICML)*. Citeseer.

Pons, P. et M. Latapy (2005). Computing communities in large networks using random walks. *Computer and Information Sciences-ISCIS 2005 3733*, pp. 284–293.

Porter, M. (2006). An algorithm for suffix stripping. *Program : electronic library and information systems 40*(3), pp. 211–218.

Porter, M., J. Onnela, et P. Mucha (2009). Communities in networks. *Notices of the American Mathematical Society 56*(9), pp. 1082–1097.

Rabbany, R., M. Takaffoli, J. Fagnan, O. Zaiane, et R. Campello (2012). Relative Validity Criteria for Community Mining Algorithms. Dans *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 258–265.

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association 66*(336), pp. 846–850.

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

Rosenberg, A. et J. Hirschberg (2007). V-measure : A conditional entropy-based external cluster evaluation measure. *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 1*(June), pp. 410–420.

Rousseeuw, P. (1987). Silhouettes : a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics 20*, pp. 53–65.

Ruspini, E. H. (1970, July). Numerical methods for fuzzy clustering. *Information Sciences 2*(3), pp. 319–350.

Sales-Pardo, M., R. Guimerà, A. A. Moreira, et L. A. N. Amaral (2007, September). Extracting the hierarchical organization of complex systems. *Proceedings of the National Academy of Sciences of the United States of America 104*(39), pp. 15224–9.

Satuluri, V. et S. Parthasarathy (2009). Scalable graph clustering using stochastic flows : applications to community discovery. Dans *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 737–746.

Schaeffer, S. (2007). Graph clustering. *Computer Science Review 1*(1), pp. 27–64.

Schuetz, P. et A. Caflisch (2007). Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics 77*(4 Pt 2), pp. 046112.

Scott, J. (2000). *Social Network Analysis : A Handbook*. Sage.

Seifi, M. (2012). *Coeurs stables de communautés dans les graphes de terrain*. Thèse de doctorat, Université Pierre et Marie Curie.

Sen, P., G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, et T. Eliassi-Rad (2008). Collective Classification in Network Data. *AI Magazine 29*(3), pp. 93–106.

Serrour, B. et H. Kheddouci (2010). Community Comparison in Communication Networks. Dans *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 393–394.

Shannon, C. E. et W. Weaver (1949). *The Mathematical Theory of Communication*. The Mathematical Theory of Communication. University of Illinois Press.

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

Smith, M., B. Shneiderman, N. Milic-Frayling, E. Mendes Rodrigues, V. Barash, C. Dunne, T. Capone, A. Perer, et E. Gleave (2009). Analyzing (social media) networks with NodeXL. Dans *Proceedings of the fourth international conference on Communities and technologies*, pp. 255–264. ACM.

Snijders, T. (2006). Statistical methods for network dynamics. Dans *Proceedings of the XLIII Scientific Meeting, Italian Statistical Society*, pp. 281–296.

Sokal, R. et C. Michener (1958). A statistical method for evaluating systematic relationships. *University of Kansas Scientific Bulletin 38*, pp. 1409 – 1438.

Solomonoff, A., A. Mielke, M. Schmidt, et H. Gish (1998). Clustering speakers by their voices. Dans *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Volume 2, pp. 757–760. IEEE.

Steinbach, M., G. Karypis, et V. Kumar (2000). A comparison of document clustering techniques. Dans *KDD workshop on text mining*, Volume 400, pp. 525–526.

Steinhaeuser, K. et N. Chawla (2008). Community detection in a large real-world social network. *Social Computing, Behavioral Modeling, and Prediction*, pp. 168–175.

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

Sun, Y. et J. Han (2012). *Mining Heterogeneous Information Networks : Principles and Methodologies*. Morgan & Claypool Publishers.

Taylor, M. (1969). Influence structures. *Sociometry 32*, pp. 490–502.

Van Dongen, S. (2000). *Graph clustering by flow simulation*. Thèse de doctorat, University of Utrecht, Netherlands.

Van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths.

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

Vinh, N., J. Epps, et J. Bailey (2010). Information theoretic measures for clusterings comparison : Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, pp. 2837–2854.

Von Luxburg, U. (2007). A Tutorial on Spectral Clustering. *Statistics and Computing 17*(4), pp. 1–32.

Wakita, K. et T. Tsurumi (2007). Finding Community Structure in Mega-scale Social Networks. Dans *Proceedings of the 16th international conference on World Wide Web*, Volume 105, pp. 1275–1276. ACM Press.

Wan, L., J. Liao, C. Wang, et X. Zhu (2009). JCCM : Joint Cluster Communities on Attribute and Relationship Data in Social Networks. Dans R. Huang, Q. Yang, J. Pei, J. a. Gama, X. Meng, et X. Li (Eds.), *Advanced Data Mining and Applications*, Volume 5678 de *Lecture Notes in Computer Science*, pp. 671–679. Springer Berlin Heidelberg.

Wang, Q. (2012). *Détection de communautés recouvrantes dans des réseaux de terrain dynamiques*. Thèse de doctorat, Lyon 1.

Wang, Q. et E. Fleury (2009). Detecting overlapping communities in graphs. Dans *European Conference on Complex Systems (ECCS)*.

Ward, J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association 58*(301), pp. 236–244.

Wasserman, S. et K. Faust (1994a). *Social Network Analysis*. Cambridge University Press.

Wasserman, S. et K. Faust (1994b). *Social network analysis : Methods and applications*. Cambridge University Press.

Watts, D. J. et S. H. Strogatz (1998). Collective Dynamics of Small-World Networks. *Nature 393*, pp. 440–442.

Xu, K., C. Tang, R. Tang, G. Ali, et J. Zhu (2010). A comparative study of six software packages for complex network research. Dans *International Conference on Communication Software and Networks*, Washington, DC, USA, pp. 350–354.

Xu, Z., Y. Wang, et J. Cheng (2012). A Model-based Approach to Attributed Graph Clustering. Dans *SIGMOD*.

Yang, J. et J. Leskovec (2012). Defining and Evaluating Network Communities based on Ground-truth. Dans *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, pp. 3. ACM.

Yang, Y. et J. O. Pedersen (1997). A comparative study on feature selection in text categorization. Dans *International Conference on Machine Learning (ICML)*, Volume 97, pp. 412–420.

Ye, Z., S. Hu, et J. Yu (2008). Adaptive clustering algorithm for community detection in complex networks. *Physical Review E 78*(4), pp. 1–6.

Ye, Z.-Q., K. Zhang, S.-N. Hu, et J. Yu (2012). A New Definition of Modularity for Community Detection in Complex Networks. *Chinese Physics Letters 29*(9 ;098901).

Yin, Z., M. Gupta, T. Weninger, et J. Han (2010). LINKREC : a unified framework for link recommendation with user attributes and graph structure. Dans *Proceedings of the 19th international conference on World wide web - WWW '10*, New York, USA, pp. 1211. ACM Press.

Zachary, W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research 33*(4), pp. 452–473.

Zhao, B. et L. Getoor (2006). Entity and Relationship Labeling in Affiliation Networks. *Networks*, pp. 8.

Zhou, Y., H. Cheng, et J. X. Yu (2009). Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment 2*(1), pp. 718–729.

Zhou, Y., H. Cheng, et J. X. Yu (2010). Clustering Large Attributed Graphs : An Efficient Incremental Approach. Dans *IEEE International Conference on Data Mining (ICDM)*, pp. 689–698. IEEE.

# Index