Indexation de graphes à partir d'une structure d'hypergraphe

Salim Jouili, Salvatore Tabbone

Indexation de graphes à partir d'une structure d'hypergraphe [1]

Salim Jouili, Salvatore Tabbone

Laboratoire LORIA UMR-7503
University of Nancy 2
BP 239, 54506 Vandoeuvre-lès-Nancy Cedex, France
{salim.jouili, tabbone}@loria.fr

RÉSUMÉ. *Dans ce papier, nous proposons une nouvelle méthode de clustering de graphes basée sur une modélisation d'hypergraphe. En premier lieu, nous appliquons un algorithme de sélection de prototype dédié aux bases de graphes où le nombre de prototype à sélectionner est déduit automatiquement. En second lieu, nous définissons une méthode de chevauchement des classes pour aboutir à la structure d'hypergraphe, où les hyperarcs sont les classes et les nœuds sont les graphes. Ainsi, un graphe peut être attribué à une ou plusieurs classes. L'originalité de notre approche réside dans la structure d'hypergraphe qui nous permet d'indexer une base de graphes à partir des centroïdes des hyperarcs. En plus, cette nouvelle approche permet de rechercher des graphes similaires à une requête et de naviguer dans une base de graphes en parcourant la structure d'hypergraphe.*

ABSTRACT. *In this paper, we propose a new hypergraph-based method for graph clustering. For this aim, we introduce firstly a prototype selection algorithm dealing with graphs where the number of clusters is automatically detected. Then, we define a clusters overlapping method to build our hypergraph-based model. Therefore, in this representation one graph can be assigned to more than one cluster. The originality of our contribution lies in the hypergraph structure which allows the indexing of the graph database using the centroids of the hyperedges. In addition, this model is interesting to travel the data set and efficient to cluster and retrieve graphs.*

MOTS-CLÉS : *graphe, clustering de graphes, indexation de graphes, recherche de graphes, hypergraphe*

KEYWORDS: *Graph, graph clustering, graph indexing, graph retrieval, hypergraph.*

1. Ce travail est partiellement financé par le projet Navidomass de l'Agence Nationale de la Recherche référencé sous le numéro ANR-06-MCDA-012 et par la région Lorraine.

1. Introduction

Récemment, nous pouvons constater à nouveau un engouement pour la représentation des objets par des graphes car les graphes offrent des alternatives polyvalentes aux vecteurs caractéristiques dans les domaines de reconnaissance des formes (Jouili *et al.,* 2008), l'apprentissage automatique et la recherche d'images (Shokoufandeh *et al.,* 2001). Les graphes sont beaucoup plus puissants en termes de représentation et plus flexibles que les vecteurs caractéristiques qui ne fournissent aucune possibilité directe de décrire les relations structurelles pour les objets en considération. En outre, alors que la taille d'un graphe peut être ajustée à la taille de l'objet, un vecteur est limité à une taille prédéfinie, qui doit être préservée pour tous les objets rencontrés dans une application particulière. Cependant, deux inconvénients majeurs des représentations sous forme de graphes sont leurs manque de méthodes appropriées pour les organiser en cluster dans l'optique d'y accéder efficacement (indexation) et la complexité élevée de l'appariemment des graphes. Par conséquent, ces inconvénient limitent l'utilisation des graphes dans des domaines comme la recherche d'image par le contenu où il s'agit d'explorer de grande bases d'images. Notant que la phase d'appariemment des graphes augmente radicalement la complexité (voir (Conte *et al.,* 2004)) de tout système de recherche. Alors pour réduire le nombre d'opérations d'appariemment, une organisation de la base des graphes sous forme de clusters est généralement requise. De cette façon, pour regrouper les images similaires dans une base, il suffit de regrouper leurs représentations sous formes de graphes. Dans ce contexte, il est naturel d'appliquer une technique de clustering aux bases de graphes. Dans la littérature, le clustering de grandes bases de graphes est resté largement inexploré et est parmi les problèmes les plus ouverts dans la reconnaissance des formes utilisant des approches structurelles.

Durant les dernières années quelques investigations sur le clustering et l'organisation des bases de graphes ont été revitalisées dans (Bunke *et al.,* 2003, Hlaoui *et al.,* 2003, Luo *et al.,* 2002, Shokoufandeh *et al.,* 2001) et l'approche que nous préconisons dans ce papier est différente. Nous proposons un modèle basé sur la structure d'hypergraphe pour regrouper un ensemble de graphes. Récemment, l'hypergraphe a été utilisé, dans le domaine de la reconnaissance des formes, pour la représentation des objets (Ren *et al.,* 2008), les mesures de similarité (Bunke *et al.,* 2008), et le clustering des objets (Agarwal *et al.,* 2005). Dans ce papier, nous établissons un modèle d'hypergraphe pour représenter une base de graphes. Nous avons procédé comme suit : premièrement, une technique de clustering basée sur la sélection de prototype est proposée pour regrouper les graphes en k clusters indépendants (k est définit automatiquement à l'aide d'un certain seuil). Deuxièmement, ces clusters sont organisés de telle façon qu'ils se recouvrent partiellement pour définir la structure d'hypergraphe finale. L'idée de chevauchement des clusters est dans la même veine que les travaux de (Bezdek *et al.,* 1999, Torsello *et al.,* 2008) mais la représentation est différente ici. Enfin, à partir d'une série d'expériences, nous avons remarqué que la structure d'hypergraphe fournit aussi un cadre pour rechercher un graphe dans une base de graphes

et la parcourir. Ces experimentations affichent également des taux élevés de clustering et montrent une amélioration de la performance de la recherche.

1.1. *La notion d'hypergraphe*

Les hypergraphes sont des objets mathématiques généralisant la notion de graphes où les arêtes peuvent connecter n'importe quel nombre de sommets. L'hypergraphe a été défini par Berge (Berge, 1970) comme suit, soit H = (ϑ,ξ) un hypergraphe, où :

– ϑ=$\{x_1$,x_2, x_3 ... , $x_n\}$: est un ensemble fini de sommets

– ξ=$\{E_1$, E_2, E_3 ... , $E_m\}$: est une famille de sous-ensembles de ϑ.

– $E_j \neq \emptyset$,$\bigcup_{j=1,\ldots,m} E_j$=$\vartheta$.

ϑ est appelé l'ensemble des sommets, ξ est l'ensemble des hyperacs et $|\vartheta|$ est le cardinal de H. Dans la figure 1(a), un hyperarc E_i est représenté par une ligne entourant ses sommets si $|E_i| \geq 2$ (par exemple E_1 dans la figure 1(a)), par une boucle sur l'élément si $|E_i|$=1(E_4 dans figure 1(a)), et par une ligne reliant les deux éléments si $|E_i|$=2 (E_5 dans figure 1(a)). Si $|E_i|$=2 pour tout les *i*, l'hypergraphe devient un graphe non orienté ordinaire.

Dans un hypergraphe, deux sommets x_i et x_j sont dits adjacents s'il existe une hyperarc E_k, qui contient les deux sommets ($x_i \in E_k$, $x_j \in E_k$). Deux hyperarcs E_i et E_j sont dits adjacents si leur intersection n'est pas vide. Tout hypergraphe a une matrice d'incidence A_i^j de taille *m*x*n* avec m colonnes représentant les hyperarcs et les n lignes représentant les sommets. Les éléments dans A indiquent l'appartenance de sommets aux hyperarcs comme suit :

$$A_i^j = \begin{cases} 1 & \text{if} \quad x_i \in E_j \\ 0 & \text{if} \quad x_i \notin E_j \end{cases}$$

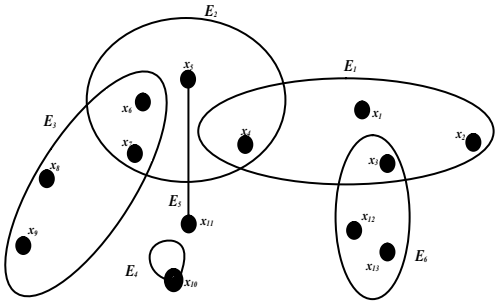
Par exemple, considerons l'hypergraphe H=(ϑ, ξ) dans la figure 1(a), ϑ=$\{x_1$,x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, $x_{13}\}$ et ξ=$\{$ E_1, E_2, E_3, E_4, E_5, $E_6\}$. La cardinalité de H est $|\vartheta|$= 13, et la matrice d'incidence est définie dans dans la figure 1(b).

Le *degré* d'un sommet, noté Δ_ϑ, est le nombre d'hyperarcs auquel il appartient, et le degré d'un hyperarc, noté Δ_ξ (*h*), est le nombre de sommets qu'il contient.

Nous pouvons noter que la différence entre une arête dans un graphe et un hyperarc dans un hypergraphe est que le premier est toujours un sous-ensemble d'un ou deux sommets, et dans le second, le sous-ensemble de sommets peut être de cardinalité arbitraire.

1.2. *Appariemment de graphes*

En fait, le calcul d'une distance entre deux graphes est un problème ouvert. Ce problème est généralement désigné comme la distance d'édition des graphes qui est



(a)

	E 1	E 2	E 3	E 4	E 5	E 6
x 1	1	0	0	0	0	0
x 2	1	0	0	0	0	0
x 3	1	0	0	0	0	1
x 4	1	1	0	0	0	0
x 5	0	1	0	0	1	0
x 6	0	1	1	0	0	0
x 7	0	1	1	0	0	0
x 8	0	0	1	0	0	0
x 9	0	0	1	0	0	0
x 10	0	0	0	1	0	0
x 11	0	0	0	0	1	0
x 12	0	0	0	0	0	1
x 13	0	0	0	0	0	1

(b)

Figure 1. *Exemple d'un hypergraphe et sa matrice d'incidence*

considéré comme un problème NP-complet et nécessite un temps et un espace exponentiels pour trouver la solution optimale (Bunke, 2000). Toutefois, pour surmonter ce problème de nombreuses approches ont été proposées pour approcher la distance d'édition des graphes. Nous renvoyons le lecteur intéressé à l'état de l'art de (Conte *et al.*, 2004). Dans ce papier, nous avons utilisé un travail précédent sur l'appriement des graphes (Jouili *et al.*, 2009a) dont le calcul des mesures de dissimilarité entre graphes repose sur trois étapes :

– l'extraction d'une signature locale en chacun des nœuds du graphe ;

– le calcul d'une matrice des distances entre chaque couple de nœuds dans les deux graphes à comparer ;

– le calcul de la mesure de dissimilarité entre les deux graphes à comparer, basée sur la matrice de dissimilarités entre les nœuds précédemment extraite.

1.2.1. *Extraction des signatures de nœuds*

La signature associée à chacun des nœuds du graphe est un vecteur calculé directement depuis la matrice d'adjacence du graphe. Notons V l'ensemble des nœuds du graphe ; la signature du nœud n_i est : $S(n_i) = \{a_i, d^0(n_i), \{a_{ij}\}_{\forall n_j \in V}\}$ où a_i est le vecteur d'attributs du nœud n_i, $d^0(n_i)$ donne le degré de n_i, et $\{a_{ij}\}_{\forall n_j \in V}$ est l'ensemble des vecteurs d'attributs des arêtes reliant le nœud n_i à l'ensemble des autres nœuds n_j du graphe.

Cette méthode est générique et peut s'appliquer à tous types de graphes (e.g. attribué, pondéré ...).

1.2.2. *Calcul de la matrice de distances entre nœuds*

Puisque les signatures peuvent contenir des éléments de différents types (numériques et/ou symboliques), la plupart des distances usuelles (*eg.* les distances L_1 ou L_2 dédiées à des attributs numériques ou la distance overlap pour des attributs symboliques) ne peuvent être utilisées. Parmi l'ensemble des métriques proposées par Wilson *et al.* dans (Wilson *et al.*, 1997) et permettant de traiter à la fois des données numériques et symboliques, nous avons choisi d'utiliser la distance Heterogeneous Euclidean Overlap Metric (HEOM) comme dans (Jouili *et al.*, 2009a). La raison de ce choix est qu'à la différence des distances basées sur des métriques de différences de valeurs, comme par exemple la Heterogeneous Value Difference Metric, la distance HEOM est indépendante du contexte d'étude et peut être utilisée dans un cadre plus général que celui de la classification. La distance HEOM utilise la distance overlap pour des données symboliques et une distance Euclidienne normalisée pour les attributs numériques. La normalisation permet d'obtenir des distances comprises entre 0 et 1.

1.2.3. *Calcul de la dissimilarité entre graphes*

Dans un premier temps, la dissimilarité entre deux graphes est exprimée par une matrice de distance HEOM entre les nœuds des graphes à comparer pris deux à deux. À partir de cette matrice on cherche à extraire une mesure de dissimilarité scalaire entre les deux graphes. La méthode utilisée dans (Jouili *et al.*, 2009a) est l'algorithme Hungarian (Kuhn, 1955) de complexité $O(n^3)$, où n est la taille du plus grand des deux graphes à comparer. Cet algorithme permet d'obtenir une matrice de permutation P définissant la fonction M de mise en correspondance optimale entre les nœuds des deux graphes :

$$M(n_i^{(g_1)}) = \begin{cases} n_j^{(g_2)} & \text{si } P_{ij} = 1 \\ 0 & \text{sinon} \end{cases}$$

où $n_i^{(g_1)}$ et $n_j^{(g_2)}$ sont respectivement les nœuds n_i et n_j des deux graphes g_1 et g_2 à comparer.

La dissimilarité entre les deux graphes peut alors être calculée comme suit :

$$d(g_1, g_2) = \frac{\hat{M}}{|M|} + |g_1| - |g_2|$$

où $|M|$ est la taille de la fonction de mise en correspondance M (*ie.* la taille du plus petit des deux graphes à comparer) et $\hat{M}$ est le coût de mise en correspondance, défini comme étant $\hat{M} = \sum_{i=1,\dots N_1} HEOM(n_i^{(g_1)}, M(n_i^{(g_1)}))$ où N_1 est le nombre total de nœuds du graphe g_1.

Un exemple concret de calcul de la dissimilarité entre deux graphes est donné dans (Jouili *et al.*, 2009a).

2. Modélisation par hypergraphe

En se basant sur la proprieté d'hypergraphe, l'idée est de représenter une clustering de graphes par un hypergraphe où les sommets correspondent à des graphes et les hyperarcs à des clusters.

Dans ce papier, un graphe peut appartenir à plusieurs hyperarcs (clusters) simultanément. Par conséquent, chaque graphe G_i dans la structure proposée est assigné à $\Delta_\vartheta(G_i)$ groupes et chaque cluster C_j contient $\Delta_\xi(C_j)$ graphes. Cependant, deux problèmes clés se posent dans la structuration d'un ensemble de graphes avec un modèle d'hypergraphe, la détermination du nombre de clusters (hyperarcs) et la détermination de graphes connexes (graphes similaires) qui peuvent être regroupés dans un hyperarc. Dans cette perspective, nous considérons que le nombre d'hyperarcs est égale à la taille d'un ensemble représentatif, définie sur une sélection de graphes les plus représentatifs de l'ensemble. On note chaque graphe sélectionné comme le centroïde d'un hyperarc. La sélection de ces graphes est similaire au problème de la sélection de prototypes (Babu *et al.,* 2001, Riesen *et al.,* 2007, Spath, 1980). K. Riesen et al. (Riesen *et al.,* 2007) énumèrent quelques techniques de sélection de prototypes à partir d'un ensemble d'apprentissage. Ces techniques nécessitent de préciser le nombre de prototypes et il n'y a pas de règles pour déterminer automatiquement ce nombre. Par conséquent, si nous nous situons dans un contexte non supervisé, où aucune information sur le nombre de graphes représentatifs est disponible, ce nombre ne sera déterminé que d'une manière empirique. Dans cette perspective, Spath (Spath, 1980) propose un algorithme utilisant les "leaders" et un seuil où le nombre de prototypes sélectionné est inversement proportionnel à la valeur du seuil sélectionné. Cependant, l'algorithme de Leader (Spath, 1980) est sensible à la sélection du premier prototype qui est choisi au hasard parmi les données en entrée. Pour surmonter ce problème, nous introduisons une technique de sélection de graphes représentatifs (centroïdes des hyperarcs) fondée sur une stratégie de pelure d'oignon. Cette méthode peut être considérée comme une amélioration de l'algorithme du Leader et de l'algorithme de *K*-Centres (Riesen *et al.,* 2007). Après la sélection des centroïdes des hyperarcs, nous définissons la structure d'hypergraphe en attribuant chaque graphe aux hyperarcs correspondants.

2.1. *Sélection de centroïdes des hyperarcs.*

Comme indiqué ci-dessus, la sélection des centroïdes des hyperarcs est similaire au problème de sélection de prototypes. Par conséquent, notre objectif est de sélectionner un sous-ensemble de graphes qui captent les aspects les plus significatifs d'un ensemble de graphes. Nous introduisons une amélioration de l'algorithme du Leader (Spath, 1980). L'algorithme proposé procède comme suit :

1) Sélectionner le graphe médian (à partir d'une technique classique (Jiang *et al.,* 2001)) G_m des graphes non affectés de l'ensemble initial des graphes *S*. Puis choisir le graphe le plus distant G_{p_k}, en fonction d'une mesure de distance choisie, (qui n'a pas été préalablement affecté) de G_m, et affecter ce graphe au cluster C_k comme centroïde.

Dans la première itération, le graphe G_{p_k} est le prototype sélectionné initialement.

2) Comparer les distances de tout les graphes non affectés $g_i \in S \setminus \{G_{p_k}\}$ à celle du dernier prototype sélectionné G_{p_k}. Si les distances d(g_i, G_{p_k}) et d(g_i, $g_{j \in C_k}$) sont inférieures à un seuil prédéfini *T* que nous définissons de manière automatique en fonction de nos données (voir §3), le graphe g_i est affecté au cluster C_k avec le centroïde G_{p_k} et g_i est étiquetté et affecté.

3) Recalculer le graphe médian G_{m_k} de C_k, si $G_{m_k} \neq G_{p_k}$, remplacer G_{p_k} par G_{m_k}. Si aucun remplacement n'est fait ($G_{m_k} = G_{p_k}$), passez à l'étape suivante, sinon tous les $g_j \in C_k$ sont marqués comme non affectés, puis revenez à l'étape 2.

4) Si *S* contient encore des graphes non-affectés revenir à l'étape 1, sinon arrêter.

Une première amélioration consiste à une adaptation de l'algorithme du Leader dans l'espace de graphes en utilisant la notion de graphe médian. Ensuite, une nouvelle méthode de sélection du premier prototype a été mise au point. Dans l'algorithme du Leader, le choix du prototype initial est fait par hasard ce qui influe sur le résultat final du clustering (i.e. à chaque exécution les résultats changent). Nous avons choisi, de fixer comme prototype initial de l'algorithme le graphe le plus éloigné du médian de la base. Ce choix garanti, par conséquent, la stabilité des résultats. Une deuxième amélioration consiste à ajouter une méthode itérative de sélection de prototype. En fait, comme il est indiqué dans le pseudo-code de l'algorithme, dès que le prototype initial est sélectionné nous considérons tous les graphes qui ont une distance au prototype inférieure ou égal à un seuil donné et la distance entre chaque couple de graphes de ce sous-ensemble doit être aussi inférieure ou égal au même seuil. En revanche, dans l'algorithme du Leader, on ne considère que les distances entre le prototype et chaque objets (il n'y a pas de critère sur les distances entre les objets). Une fois le sous-ensemble construit on recalcule le nouveau graphe médian et on itère jusqu'à convergence vers un graphe médian prototype considéré comme un centroïde.

Compte tenu d'un seuil *T*, l'algorithme ci-dessus regroupe l'ensemble des graphes avec une inertie intra-classe (I_i) inférieure ou égale à *T*. Cette propriété est conservée dans l'étape 2. En plus, cet algorithme garantit : 1) la sélection des prototypes qui sont les centroïdes des clusters ; 2) une certaine séparabilité entre classes d'une partition. En outre, en fixant le prototype initial comme le graphe le plus distant du graphe médian dans l'ensemble des graphes, l'algorithme produit des résultats identiques et est déterministe. Nous notons cet algorithme de clustering par D-hypergraphe pour hypergraphe déconnecté.

2.2. *La représentation sous forme d'hypergraphe.*

Soit *S* un ensemble de graphes et *P* l'ensemble de prototypes sélectionnés *P* (*P*⊂*S*). Les techniques de clustering classique cherchent pour chaque graphe $g \in S \setminus P$ son plus proche voisin $p_i \in P$ et ajoutent le graphe dans le cluster C_i correspondant au prototype p_i. En fait, si un graphe *g* présente une distance similaire à deux prototypes p_i et p_j, g est ajouté au cluster avec le prototype le plus proche même si la différence

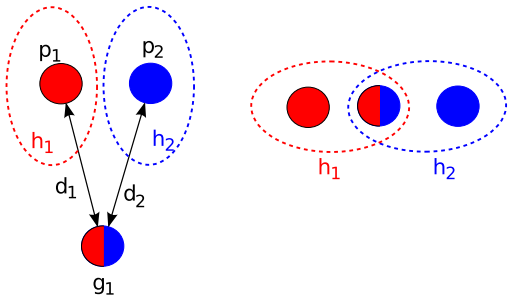


Figure 2. *Illustration du modèle proposé*

entre les deux distances est très mineure. En outre, les clusters sont disjoints et peuvent être exploités pour une tâche de recherche (Robles-Kelly *et al.*, 2005, Sebastian *et al.*, 2002, Sengupta *et al.*, 1995, Shapiro *et al.*, 1982), mais il sera difficile de trouver un algorithme pour naviguer dans l'ensemble de la base de graphes.

Au contraire, nous proposons un modèle basé sur l'hypergraphe qui permet le chevauchement des clusters. Désormais, les clusters seront considérés comme des hyperarcs d'hypergraphe et les graphes comme les sommets. Tout d'abord, pour chaque prototype p_i un hyperarc h_i est défini avec un centroïde p_i. Deuxièmement, chaque hyperarc est défini comme suit : chaque graphe $g \in S \setminus P$ est ajouté aux hyperarcs avec les prototypes proche de g (leurs distances à g est inférieur au seuil T utilisé dans l'algorithme précédent). Nous désignions cette procédure par C-hypergraphe (hypergraphe connecté).

La figure 2 illustre notre motivation, soit $d_i = d(p_i, g_1)$, nous supposons que d_1 et d_2 sont inférieures ou égales à un seuil T, alors le graphe g_1 partage quelques informations avec p_1 et p_2 (les informations sont illustrées en couleur). Avec le modèle d'hypergraphe nous serons en mesure d'affecter g_1 à la fois aux hyperarcs h_1 et h_2. La partie la plus à droite de la figure 2 décrit comment deux hyperarcs (clusters) peuvent se chevaucher avec un graphe en commun. Ici, $\Delta_\vartheta(g_1)=2$ et $\Delta_\xi(h_1)=\Delta_\xi(h_2)=2$.

Une fois que tout les hyperarcs sont définis à partir des graphes, on recalcule, pour chaque hyperarc, le graphe médian généralisé qui sera le nouveau centroïde de l'hyperarc. L'objectif de cette étape est de mettre à jour le centroïde de l'hyperarc et de maintenir autant d'informations que possible des graphes dans le hyperarc correspondant. Nous avons utilisé le graphe médian généralisé pour définir le centroïde d'un cluster car contrairement au supergraphe minimum et commun (Bunke *et al.*, 2003) il est moins coûteux en temps de calcul.

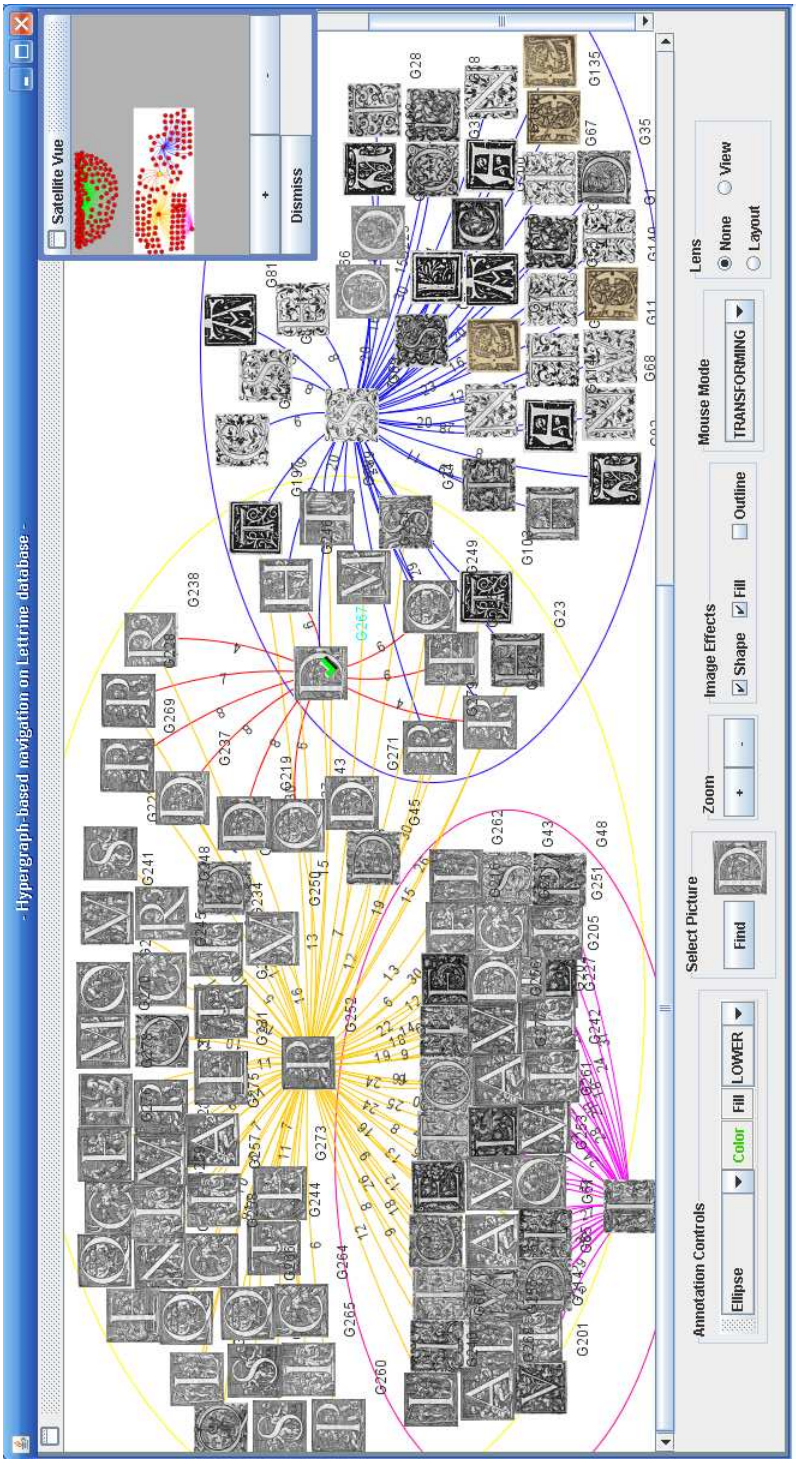


Figure 3. *Illustration de l'interface développée pour la navigation dans une base de graphes*

2.3. *Interrogation et Navigation d'une base de graphe à partir du modèle d'hypergraphe.*

Classiquement, l'interrogation d'un ensemble de graphes consiste à rechercher les graphes les plus semblable à une requête donnée. Cette tâche de recherche trie les distances entre la requête et les graphes de la base dans un ordre croissant. Comme il est déjà indiqué dans la littérature (Robles-Kelly *et al.*, 2005, Sebastian *et al.*, 2002, Sengupta *et al.*, 1995, Shapiro *et al.*, 1982), cette méthode n'exploite pas efficacement la distribution des distances, et les auteurs proposent des techniques basées sur le clustering pour améliorer les résultats de recherche. Dans ce travail, nous introduisons une procédure qui implique le modèle de l'hypergraphe proposé précédemment. L'idée principale est de trouver le centroïde (parmi tous les centroïdes des hyperarcs) le plus proche à une requête donnée. Puis, nous recherchons les graphes les plus similaires au sein de l'hyperarc. Nous décrivons la procédure de recherche dans le modèle de l'hypergraphe comme suit :

1) Pour un graphe de requête g_q, calculer l'ensemble des distances entre g_q et le centroïde de chaque hyperarc.

2) Déterminer le plus proche centroïde p_i à g_q.

3) Récupérer les graphes les plus similaires g_j à g_q, où $g_j \in h_i$ et h_i est le hyperarc avec le centroïde p_i.

Ce modèle basé sur l'hypergraphe peut être exploité pour naviguer à travers l'hypergraphe. Une fois la procédure précédente effectuée, l'utilisateur peut parcourir l'ensemble des graphes, à travers une interface graphique. Dans la figure 3 les clusters (hyperarcs) sont représentés par des ellipses qui se chevauchent. Dans cette figure, l'utilisateur a sélectionné la lettrine avec la lettre D et, en cliquant sur les lettrines voisines, a pu explorer une partie de la base liée à sa requête. Les images au centre de chaque hyperarc sont les centroïdes.

3. Expérimentations

Dans cette partie nous évaluons d'un côté la qualité du clustering à partir de trois bases d'images (§3.1) et de l'autre l'efficacité du modèle proposé pour la recherche de graphes (§3.2).

3.1. *L'évaluation de clustering*

Dans cette première partie des expérimentations, nous effectuons des tests de clustering de graphes. Ici, nous évaluons les deux algorithmes proposés. Le premier est le clustering basé sur la sélection de prototypes, sans connexion des hyperacs dans l'hypergraphe, notée D-hypergraphe. Le second permet le chevauchement des hyperarcs, noté C-hypergraphe. Nous avons établi une comparaison avec l'algorithme de K-means sur trois bases d'images (voir figure 4). La première est la base d'images

COIL (Nene *et al.,* 1996) qui contient différents vues d'objets 3D. Les images de COIL sont converties en graphes par extraction des points d'intérêt en utilisant l'algorithme de Harris (Harris *et al.,* 1988) et la triangulation de Delaunay. La seconde est la base GREC[1] (Riesen *et al.,* 2008) qui se compose de symboles issus de plans architecturaux et électroniques. Ici, les points finaux (les coins, les intersections et les cercles) sont représentés par des nœuds dans le graphe qui sont reliés par des arêtes non orientés et étiquetés comme des lignes ou des arcs. Enfin, nous avons effectué l'évaluation du clustering sur une base de lettres ornementales qui contient des Lettrines (objet graphique) extraites de documents anciens numérisés[2]. Puisqu'une Lettrine contient beaucoup d'informations (i.e la texture, décor de fond, lettres), les graphes sont extraits suite à une segmentation en régions (Felzenszwalb *et al.,* 2004) de la Lettrine. Les nœuds du graphe sont représentés par les régions et les arcs décrivent leurs relations d'adjacence. La mesure de distance entre les graphes utilisé est l'appariemment de graphes basé sur les signatures de nœuds (Jouili *et al.,* 2009a, Jouili *et al.,* 2009b). Les résultats du clustering sont évalués par l'*indice de Rand* (Rand, 1971), l'*indice de Dunn* (Dunn, 1974) et l'*indice de Davies-Bouldin* (Davies *et al.,* 2009).

L'*indice de Rand* mesure à quel point les groupes crées par l'algorithme de clustering correspondent à la vérité terrain. Pour calculer l'*indice de Rand*, nous considérons toutes les paires de graphes (g_i, g_i) avec $g_i \neq g_j$. Soit N_{11} le nombre de paires (g_i, g_i) appartenant à la même classe (dans la vérité terrain) et au même cluster (dans la clustering). Soit N_{00} le nombre de paires (g_i, g_i) n'appartenant ni à la même classe (dans la vérité terrain) et ni au même cluster (dans la clustering), soit N_{10} le nombre de paires (g_i, g_i) appartenant à la même classe (dans la vérité terrain) mais pas au même cluster (dans la clustering) et soit N_{01} est le nombre de paires (g_i, g_i) appartenant au même cluster (dans la clustering) mais pas à la même classe (dans la vérité terrain), L'*indice de Rand* s'écrit :

$$Rand = \frac{N_{11} + N_{00}}{N_{11} + N_{00} + N_{10} + N_{01}}$$

L'*indice de Dunn* est une mesure de la compacité et la séparation des clusters et contrairement à l'*indice de Rand*, l'*indice de Dunn* n'est pas normalisée. Soient d_{min} la plus petite distance entre deux graphes de clusters différents, et soit d_{max} la plus grande distance entre deux graphes de même cluster, l'*indice de Dunn* est donné par :

$$Dunn = \frac{d_{min}}{d_{max}}$$

L'*indice Davies-Bouldin* est une fonction de la compacité intra-cluster et la séparation inter-cluster et donné par :

$$DaviesBouldin = \frac{1}{M}\sum_{i=1}^{m} \underset{j=1,\cdots,M;j\neq i}{max}(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)})$$

1. `http://www.cvc.uab.es/grec2003/`
2. fournis par le CESR - Université de Tours sur le cadre du projet ANR Navidomass `http://l3iexp.univ-lr.fr/navidomass/`

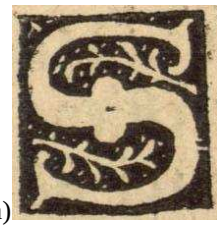
(a)  (b)  (c) 

Figure 4. *Exemples des bases d'images utilisée pour les expérimentations : (a) Lettrine, (b) GREC, (c) COIL.*

où M est le nombre de clusters, σ_i la moyenne des distances entre tout les graphes dans un cluster *i* et le centroïde c_i de *i*. $d(c_i, c_j)$ est la distance entre les deux centroïdes c_i et c_j.

Notons qu'un bon clustering donne des petites valeurs de l'*indice de Davies-Bouldin* et des valeurs importantes des indices de *Rand* et de *Dunn*. Dans cette expérience, le seuil *T*, utilisé par notre méthode, est défini comme la moyenne des distances entre les graphes dans la même base. Le nombre de classes k utilisées par l'algorithme du k-means est défini conformément à la vérité terrain.

La table 1 montre les résultats des trois indices de validation du clustering. A partir de ces résultats, il est clair que notre méthode d'hypergraphe déconnecté produit des clusters plus compactes et bien séparées. Nous remarquons que lorsque l'algorithme du C-hypergraphe est appliqué, l'*indice de Dunn* prend la valeur 0, parce que certains clusters partagent des graphes et la distance minimale entre les clusters devient nulle. En outre, d'un point de vue de similitude avec la vérité terrain, notre modèle donne de meilleurs résultats pour les bases GREC et Lettrines, et nous pouvons également remarquer que les *indices de Rand* produis par l'algorithme du C-hypergraphe pour les trois bases de données sont plus élevés que ceux du D-hypergraphe. Par conséquent, l'hypergraphe connecté s'adapte mieux à la vérité terrain et nous encourage à exploiter la structure de l'hypergraphe pour le problème de la recherche de graphes.

3.2. *Évaluation de la recherche avec le modèle de l'hypergraphe.*

Dans cette partie, nous évaluons la recherche à partir du modèle de l'hypergraphe en exécutant l'algorithme détaillé précédemment (§2.3) sur la base des lettres ornementales. Nous fournissons une comparaison avec une recherche classique dans laquelle la requête est comparée à tous les graphes dans la base de données, en triant les distances du plus similaire au moins similaires. Dans l'approche proposée, les centroïdes des hyperarcs sont les entrées (l'index) de la base de graphes. Plus précisemment, tout d'abord le graphe requête n'est comparé qu'aux centroïdes des hyperacs. Puis, la recherche est effectuée parmi les graphes qui appartiennent à l'hyperarc avec

	K-means	D-Hypergraphe	C-Hypergraphe
COIL	k=100	T=18.66, Nc=276	T=18.66
indice de Rand	0.75	0.74	**0.75**
indice de Dunn	0.03	**0.04**	0.00
indice de DB	0.98	**0.88**	1.25
GREC	k=22	T=6.20, Nc=21	T=6.20
indice de Rand	0.86	0.88	**0.91**
indice de Dunn	0.01	**0.04**	0.00
indice de DB	0.83	**0.76**	0.94
Lettrine	k=4	T=53.20, Nc=4	T=53.20
indice de Rand	0.64	0.68	**0.69**
indice de Dunn	0.10	**0.13**	0.00
indice de DB	0.81	**0.61**	0.92

Tableau 1. *Evaluation du clustering et comparaison avec K-means (Nc : nombre de clusters détectés)*

le plus proche centroïde à la requête. Nous avons utilisé la courbe Précision[3]-Rappel[4] pour visualiser les performances de la recherche. Ces courbes sont formées par le taux de précision par rapport au taux de rappel (voir figure 5). En analysant les deux courbes, on peut remarquer que les résultats sont meilleurs lorsque la recherche est effectuée dans un seul hyperarc. En outre, le modèle de l'hypergraphe est plus rapide que la technique classique car il ne compare pas la requête avec tous les graphes dans la base mais seulement avec des graphes dans un cluster approprié. La figure 6 illustre une comparaison entre la méthode classique de recherche et le modèle proposé. Pour chaque image de la base de Lettrines nous avons compté le nombre d'opérations d'appariemment effectué pour aboutir à l'image correspondant à la requête. Il s'agit ici d'une recherche exacte. Le nombre de comparaisons pour la recherche classique reste constant car nous calculons à chaque fois pour chaque image la distance entre cette image et toutes les autres images de la base. Nous remarquons qu'avec notre approche le nombre d'appriements effectués dépend de l'image requête, mais pour toutes les images testées le nombre de comparaisons est nettement inferieur à la méthode classique de recherche d'images.

4. Conclusion

Dans ce papier, nous avons étudié comment la structure d'hypergraphe peut être utilisée à des fins de représentation de base de graphes. Nous avons proposé une mé-

3. Précision : rapport du nombre de documents pertinents trouvés au nombre total de documents sélectionnés.
4. Rappel : rapport du nombre de documents pertinents trouvés au nombre total de documents pertinents

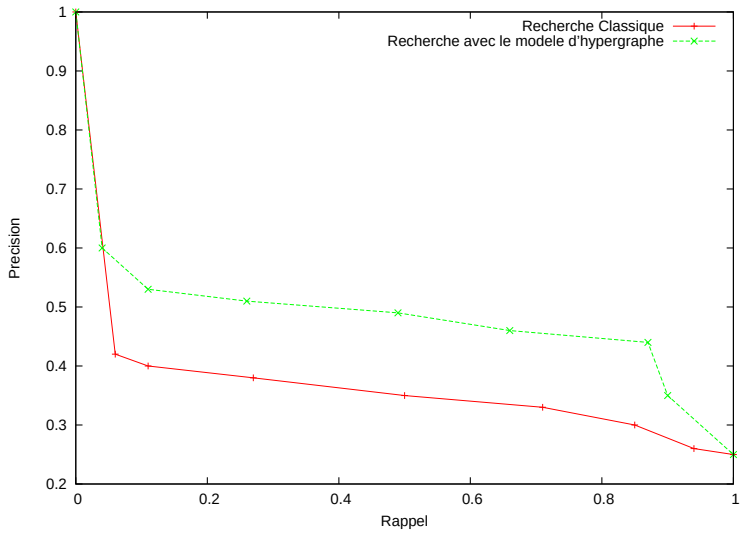


Figure 5. *Les courbes Précision-Rappel : comparaison entre la recherche classique et la recherche basée sur le modèle proposé*

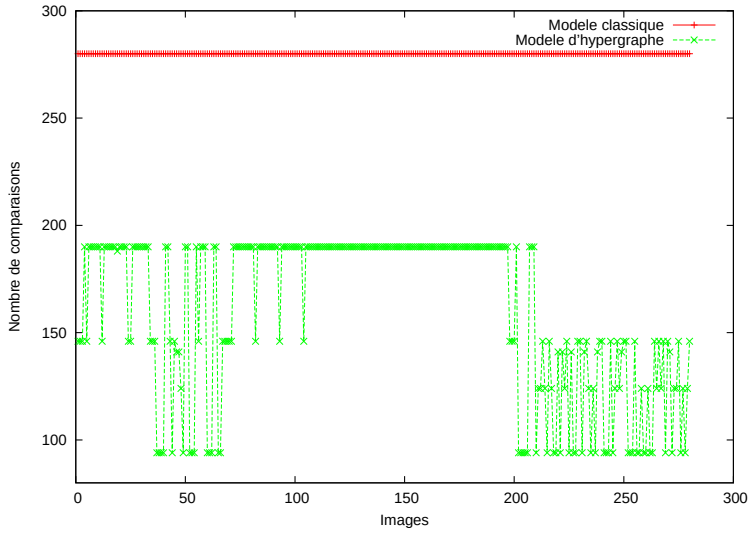


Figure 6. *Nombre de comparaisons effectuées lors de la recherche d'images*

thode basée sur la sélection de prototypes pour le clustering des graphes. La sélection des prototypes proposée permet de définir automatiquement le nombre de graphes. Ce travail permet également la multi-affectation d'un graphe, à savoir un graphe peut être affecté à plusieurs clusters. Nous avons aussi montré que la structrure d'hypergraphe améliore les résultats de la recherche (temps et précision) et peut être utilisée pour naviguer dans une base de graphes. Les algorithmes proposés dans ce papier s'appuient sur un seuil qui est automatiquement définit à partir de la moyenne des distances entre les graphes de la base. En perspective, une étude sur le comportement de notre méthode par rapport à la variation de la valeur du seuil validerait d'avantage le modèle d'hypergraphe proposé.

5. Bibliographie

Agarwal S., Lim J., Zelnik-Manor L., Perona P., Kriegman D. J., Belongie S., « Beyond Pairwise Clustering », *IEEE CVPR*, p. 838-845, 2005.

Babu T. R., Murty M. N., « Comparaison of genetic algorithm based prototype selection schemes », *Pattern Recognition*, vol. 34, p. 523-525, 2001.

Berge C., *Graphes et Hypergraphes*, Paris Dunod, 1970.

Bezdek J. C., Pal M. R., Keller J., Krisnapuram R., *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer Academic Publishers, Norwell, MA, USA, 1999.

Bunke H., « Recent Developments in Graph Matching », *Pattern Recognition, 2000. Proceedings. 15th International Conference on*p. 117-124 vol.2, 2000.

Bunke H., Dickinson P. J., Kraetzl M., Neuhaus M., Stettler M., « Matching of Hypergraphs - Algorithms, Applications, and Experiments », *in* , H. Bunke, , A. Kandel, , M. Last (eds), *Applied Pattern Recognition*, vol. 91 of *Studies in Computational Intelligence*, Springer, p. 131-154, 2008.

Bunke H., Foggia P., Guidobaldi C., Vento M., « Graph Clustering Using the Weighted Minimum Common Supergraph », *IAPR Workshop GbRPR 2003*, *LNCS 2726*p. 235-246, 2003.

Conte D., Foggia P., Sansone C., Vento M., « Thirty years of graph matching in pattern recognition », *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, n° 3, p. 265-298, 2004.

Davies D. L., Bouldin D. W., « A Cluster Separation Measure », *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-1, n° 2, p. 224-227, January, 2009.

Dunn J. C., « Well separated clusters and optimal fuzzy-partitions », *Journal of Cybernetics*, vol. 4, p. 95-104, 1974.

Felzenszwalb P. F., Huttenlocher D. P., « Efficient Graph-Based Image Segmentation », *International Journal of Computer Vision*, vol. 59, n° 2, p. 167-181, 2004.

Harris C., Stephens M., « A Combined Corner and Edge Detection », *Proceedings of The Fourth Alvey Vision Conference*, p. 147-151, 1988.

Hlaoui A., Wang S., « A graph clustering algorithm with applications to content-based image retrieval », *ICMLC, 2003*, vol. 3, p. 1855-1861, 2003.

Jiang X., Munger A., Bunke H., « On median graphs :properties, algorithms, and applications », *IEEE TPAMI*, vol. 23, n° 10, p. 1144-1151, 2001.

Jouili S., Mili I., Tabbone S., « Attributed Graph Matching Using Local Descriptions », *Advanced Concepts for Intelligent Vision Systems, ACIVS 2009, LNCS 5807*, p. 89-99, 2009a.

Jouili S., Tabbone S., « Applications des graphes en traitement d'images », *International Conference on Relations, Orders and Graphs : Interaction with Computer Science, ROGICS'08*, p. 434-442, 2008.

Jouili S., Tabbone S., « Graph matching using node signatures », *IAPR Workshop on GbRPR, LNCS 5534*, p. 154-163, 2009b.

Kuhn H. W., « The Hungarian Method for the assignment problem », *Naval Research Logistics Quarterly*, vol. 2, p. 83-97, 1955.

Luo B., Wilson R. C., Hancock E. R., « Spectral feature vectors for graph clustering », *IAPR Workshop on S+SSPR, LNCS 2396*p. 83-93, 2002.

Nene S., Nayar S., Murase H., « Columbia Object Image Library (COIL-100) », *technical report, Columbia Univ.*, 1996.

Rand W., « Objective criteria for the evaluation of clustering methods », *Journal of the American Statistical Association*, vol. 66, n° 336, p. 846-850, 1971.

Ren P., Wilson R. C., Hancock E. R., « Spectral Embedding of Feature Hypergraphs », *IAPR Workshop S+SSPR, LNCS 5342*, p. 308-317, 2008.

Riesen K., Bunke H., « IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning », *IAPR Workshop on S+SSPR, LNCS 5342*, p. 287-297, 2008.

Riesen K., Neuhaus M., Bunke H., « Graph Embedding in Vector Spaces by Means of Prototype Selection », *IAPR Workshop on GbRPR, LNCS 4538*p. 383-393, 2007.

Robles-Kelly A., Hancock E. R., « Graph Edit Distance from Spectral Seriation », *IEEE TPAMI*, vol. 27, n° 3, p. 365-378, 2005.

Sebastian T. B., Klein P. N., Kimia B. B., « Shock-Based Indexing into Large Shape Databases », *7th European Conference on Computer Vision, LNCS 2352*, p. 731-746, 2002.

Sengupta K., Boyer K., « Organizing large structural modelbases », *IEEE TPAMI*, vol. 17, n° 4, p. 321-332, Apr, 1995.

Shapiro L. G., Haralick R. M., « Organization of Relational Models for Scene Analysis », *IEEE TPAMI*, vol. PAMI-4, n° 6, p. 595-602, Nov., 1982.

Shokoufandeh A., Dickinson S. J., « A Unified Framework for Indexing and Matching Hierarchical Shape Structures », *4th Int. Workshop on Visual Form, LNCS 2059*, p. 67-84, 2001.

Spath H., « Cluster Analysis Algorithms for Data Reduction and Classification of Objects », *Ellis Horwood Limited, West Sussex*, 1980.

Torsello A., Bulò S. R., Pelillo M., « Beyond partitions : Allowing overlapping groups in pairwise clustering », *ICPR*, IEEE, p. 1-4, 2008.

Wilson D. R., Martinez T. R., « Improved Heterogeneous Distance Functions », *Journal of Artificial Intelligence Research*, vol. 6, p. 1-34, 1997.