Intégration Holistique des Graphes basée sur la Programmation Linéaire pour l'Entreposage des Open Data

Alain Berro*, Imen Megdiche*, Olivier Teste*

* IRIT UMR 5505, Université de Toulouse
CNRS, INPT, UPS, UT1, UT2J, 31062 Toulouse Cedex 9
{berro, megdiche, teste}@irit.fr

Résumé. Dans cet article, nous proposons une approche holistique pour l'intégration des graphes d'Open Data. Ces graphes représentent une classification hiérarchique des concepts extraits des Open Data. Nous nous focalisons sur la conservation de hiérarchies strictes lors de l'intégration afin de pouvoir définir un schéma multidimensionnel à partir de ces hiérarchies et entreposer par la suite ces sources de données. Notre approche est basée sur un programme linéaire qui résout automatiquement la tâche de matching des graphes tout en maximisant globalement la somme des similarités entre les concepts. Ce programme est composé de contraintes sur la cardinalité du matching et de contraintes sur la structure des graphes. A notre connaissance, notre approche est la première à fournir une solution optimale globale pour le matching holistique des graphes avec un temps de résolution raisonnable. Nous comparons également la qualité des résultats de notre approche par rapport à d'autres approches de la littérature.

1 Introduction

L'intégration des données est un problème de recherche largement étudié depuis plusieurs années (Rahm et Bernstein, 2001). La difficulté dans ce domaine réside dans la complexité et la diversité des sources de données à traiter et dans l'estimation de la qualité et de la performance des approches automatiques. L'intégration repose sur une tâche appelée matching qui a pour rôle de déterminer les meilleures correspondances entre les éléments des sources de données. Dans la littérature, nous distinguons deux axes de matching selon le nombre de sources à traiter à savoir le pair-wise matching (deux sources de données) et le matching holistique (plusieurs sources de données). Pour le pair-wise matching, le défi consiste à trouver les meilleures correspondances pour deux sources de données (de petite ou moyenne taille). A ce défi s'ajoute le problème de performance lorsque les sources sont de grandes tailles. Quant au matching holistique, les défis consistent à maintenir une bonne qualité des correspondances et à garantir une performance acceptable vis-à-vis de la taille et du nombre de sources traitées.

Les données ouvertes ou Open Data (OD) tabulaires statistiques constituent des sources d'informations intéressantes à intégrer dans les entrepôts de données vu la diversité des scénarios d'analyses qui peuvent en découler. Toutefois, ces données ont trois principales caractéristiques qui complexifient leur intégration : l'hétérogénéité syntaxique et sémantique, la

déstructuration du schéma et la dispersion des sources. D'abord l'hétérogénéité est importante car des vocabulaires différents sont utilisés et une multitude de thématiques est abordée par les OD. Ensuite, les schémas sont peu structurés : ces sources englobent données statistiques et structures dans des tableaux croisés, il est difficile de dissocier automatiquement les structures de données afin de pouvoir constituer et réutiliser le schéma de ces tableaux. Enfin, ces données tabulaires de petite ou moyenne taille sont éparpillées sur un fournisseur ou plusieurs fournisseurs. Il est généralement indispensable de rassembler plusieurs sources pour former un scénario d'analyse.

Nous avons proposé dans nos travaux antérieurs (Berro et al., 2013, 2014) une approche d'intégration d'Open Data tabulaires dans les entrepôts de données. L'idée consiste à faire pivoter des graphes entre les trois phases d'extraction, transformation et chargement (ETL) afin de pallier aux difficultés que posent ces OD tabulaires. Nous avons proposé dans (Berro et al., 2014) un processus permettant de transformer les OD tabulaires en graphes (plus précisément des forêts). Nous avons choisi de transformer les OD en graphes pour représenter de façon unifiée des sources structurellement hétérogènes. Ce choix a été également motivé par la capacité des graphes de supporter d'autres extensions dans notre processus d'extraction. Nous dissocions structures et valeurs par des algorithmes d'extraction et d'annotation de données (Berro et al., 2014). Nous appliquons également des algorithmes d'enrichissement des structures plates afin de proposer des hiérarchies non-complexes (Malinowski et Zimányi, 2006) dans les graphes générés.

Dans cet article nous nous focalisons sur la deuxième phase de notre approche qui consiste à intégrer des graphes d'OD issus de la phase d'extraction en faisant du matching de graphes. Nous avons choisi d'aborder une approche holistique vu le problème d'éparpillement des sources et l'utilisation de dictionnaires externes non spécifiques à un domaine pour répondre au problème d'hétérogénéité sémantique. Nous proposons une nouvelle méthode de résolution du problème de matching holistique des graphes basée sur la technique de programmation linéaire. Cette technique nous permettra d'une part d'apporter une solution optimale globale au problème de matching holistique, d'autre part nous fournira de manière automatique une solution unique plus facilement exploitable par le concepteur pour la construction de l'entrepôt.

Le reste de cet article est organisé comme suit. Dans la section 2, nous présentons certaines approches de la littérature connues ou proches de notre proposition. Dans la section 3, nous définissons le processus de matching de notre programme linéaire. Dans la section 4, nous présentons une série d'expérimentations afin d'évaluer la performance et la qualité de notre approche.

2 État de l'art

Différentes approches et techniques de matching ont été proposées dans la littérature. Les synthèses menées par (Rahm et Bernstein, 2001), (Shvaiko et Euzenat, 2005) et plus récemment par (Bernstein et al., 2011) et (Rahm, 2011) se sont succédées pour classifier ces travaux. En plus du type d'approche pair-wise ou holistique, la plupart des travaux sont hybrides c'est-à-dire qu'ils combinent plusieurs types de matchers individuels. Ces derniers sont de deux types : les matchers basés sur les éléments des sources et les matchers basés sur la structure des sources. Les matchers basés sur les éléments utilisent différents types de mesures de similarités : (i) mesures terminologiques (tel que levenshtein, jaccard, tf.idf, n-gram...) et (ii)

mesures linguistiques qui calculent la similarité des sens des termes qui composent les éléments, nous y trouvons les mesures sémantiques comme sous-type des mesures linguistiques. Les techniques basées sur la structure utilisent les contraintes des éléments et la structure des sources pour calculer les similarités. Par ailleurs, la plupart des approches utilisent un format interne de représentation des données qui s'apparente souvent à des graphes.

Parmi les approches de type pair-wise, nous citons COMA++ de (Aumueller et al., 2005), Similarity Flooding (SF) de (Melnik et al., 2002) et BMatch de (Duchateau et al., 2007). Ces approches sont hybrides, utilisent toutes des matchers terminologiques et un format interne soit en graphe (SF) soit en arbre (COMA++ et BMacth). COMA++ utilise une large palette de matchers linguistiques et structurels et propose soit de combiner les matchers, soit d'appliquer des stratégies prédéfinies de matching de schémas de taille et de format différents. SF propose un algorithme qui propage et calcule jusqu'à un certain point fixe une similarité structurelle entre les graphes de paires de noeuds. Son algorithme considère que si une paire de noeuds est similaire alors les voisins sont également similaires. SF est reconnue comme l'une des meilleures approches de matching des graphes. BMatch propose de combiner deux mesures terminologiques et une mesure structurelle basée sur le contexte des éléments. Il a pour principe d'utiliser un b-tree pour améliorer la performance de comparaison des éléments voisins.

Parmi les approches de type holistique, nous citons PSM de (Su et al., 2006), PORSCHE de (Saleem et al., 2007) et PLASMA de (Benharkat et al., 2007). PSM est une approche non hybride appliquée sur des formulaires web. Les schémas de PSM sont simples puisqu'ils sont composés d'une liste d'attributs. Les auteurs proposent une mesure de similarité terminologique qui calcule des statistiques sur les occurrences des éléments. PORSCHE est une approche hybride qui utilise comme format interne les arbres et une mesure de similarité linguistique propre à un domaine. Elle construit par une technique de fouille d'arbres des regroupements d'éléments similaires à partir desquels un algorithme incrémental calcule les correspondances. PLASMA est une approche hybride qui s'applique sur des arbres. L'objectif de cette approche est d'optimiser la phase de matching en appliquant une phase d'analyse de la structure des arbres à partir de laquelle elle dégage des sous-arbres communs. Le matching structurel et sémantique est ensuite appliqué en pair-wise sur ses sous-arbres.

Discussion Dans le contexte des Open Data, nous avons besoin d'intégrer plusieurs graphes. Généraliser des approches pairwise ou utiliser des approches holistiques existantes peuvent être des pistes mais l'optimalité des solutions automatiques proposées dans un contexte holistique n'est pas garantie. En effet, si nous faisons le matching de façon indépendante de toutes les combinaisons ou un nombre réduit de combinaisons de paire de graphes, les solutions obtenues correspondent à des optimum locaux qui ne tiennent pas compte des contraintes des autres schémas. Si, par ailleurs, nous utilisons des schémas intermédiaires et le principe de réutilisation, il n'y a aucune garantie sur la fermeture du processus qui induit qu'un concepteur obtiendrait des solutions différentes suivant l'ordre avec lequel les OD seraient intégrés.

Pour les approches holistiques, il n'existe pas à notre connaissance d'outils disponibles. Les approches PORSCHE et PLASMA qui semblent les plus prometteuses utilisent des algorithmes incrémentaux, ce qui ne garantit pas l'optimalité globale de la solution obtenue (c'est-à-dire la meilleure solution parmi toutes les solutions possibles).

Afin de garantir l'optimalité globale, nous proposons une approche holistique de matching de graphes basée sur la programmation linéaire (Nash et Sofer, 1996). Par rapport aux tra-

vaux existants, notre approche est hybride puisque nous combinons mesures terminologiques et linguistiques, et contraintes structurelles. En outre, le format interne est un graphe orienté acyclique.

3 Une Approche Holistique d'Intégration des Graphes

Le noyau de notre approche holistique est un programme linéaire en variables 0-1, nommé LP4HM (Programme Linéaire pour le Matching Holistique). Son objectif consiste à trouver un optimum global c'est-à-dire la meilleure somme de similarité entre correspondances. L'optimum global est calculée pour $\sum_{i=1}^{N-1}(N-i)$ combinaisons de deux graphes formées à partir des $N \geq 2$ graphes fournis en entrée par le concepteur. Ces graphes représentent le schéma des Open Data transformés par des processus décrits dans (Berro et al., 2014). Les noeuds de ces graphes sont des concepts et les arcs représentent des relations hiérarchiques entre les concepts. En sortie, notre approche fournit un unique graphe intégré généré à partir des correspondances retournées par le programme linéaire et l'ensemble des graphes en entrée.

Quelques Notations

- $G_i = (V_i, E_i)$, $i \in [1, N]$ est le graphe d'Open Data ;
- $V_i = \{v_{i_k},\ \forall k \in [1, n_i]\}$ est l'ensemble des noeuds de chaque graphe G_i ;
- $E_i = \{e_{i_{k,l}} = (v_{i_k}, v_{i_l}),\ \forall k, l \in [1, n_i]\}$ est l'ensemble des arcs de chaque graphe G_i ;
- i, j sont les numérotations des graphes G_i et G_j ;
- n_i est la cardinalité de l'ensemble des noeuds ($|V_i|$) dans le graphe G_i ;
- i_k est l'indice du noeud d'ordre k (ordre de parcours) dans le graphe G_i ;
- j_l est l'indice du noeud d'ordre l (ordre de parcours) dans le graphe G_j ;
- $i_{pred(k)}$ est l'indice du prédécesseur du noeud v_{i_k} (le prédécesseur est unique dans nos graphes puisque les hiérarchies sont strictes).

Exemple d'illustration Afin d'illustrer notre problématique, nous avons sélectionné quelques sources d'Open Data contenant des statistiques de saisies de drogues au Royaume-Uni. Les graphes G_1, G_2 et G_3 de la figure 1 montrent un extrait de ce cas d'étude concernant les types de drogues. G_1 représente des données extraites du lien [1]. G_2 et G_3 représentent des données extraites du lien [2]. Cet exemple nous servira de support pour expliquer notre programme linéaire.

Dans la section précédente, nous avons signalé qu'un optimum global est meilleur que les optima locaux dans le cadre d'intégration holistique. Nous avons alors effectué, sur notre exemple d'étude, une simple comparaison entre les résultats de LP4HM et les résultats de Similarity Flooding. Nous avons utilisé avec ces deux approches les mêmes mesures de similarités entre les concepts des noeuds des graphes. Ces dernières ont été calculées par les distances que nous proposons dans la section 3.1.

La solution retournée par l'algorithme Similarity Flooding est la somme des optima locaux qui est égale à 3.76 :

1. `http://www.scotland.gov.uk/Topics/Statistics/Browse/Crime-Justice/TrendData`
2. `http://data.gov.uk/dataset/seizures-drugs-england-wales`

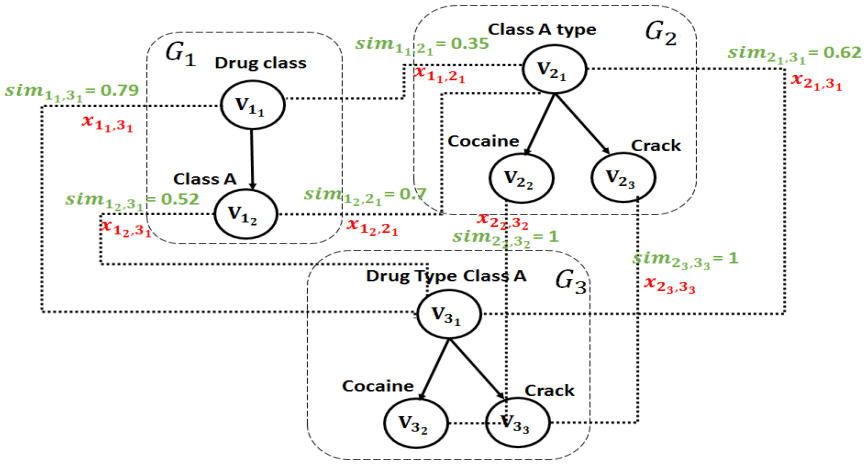


FIG. 1: Un exemple de trois graphes à intégrer holistiquement

- $Match(G_1, G_2) \rightarrow \{(v_{1_1}, v_{2_1}) = 0.35\}$;
- $Match(G_1, G_3) \rightarrow \{(v_{1_1}, v_{3_1}) = 0.79\}$;
- $Match(G_2, G_3) \rightarrow \{(v_{2_1}, v_{3_1}) = 0.62, (v_{2_2}, v_{3_2}) = 1, (v_{2_3}, v_{3_3}) = 1\}$.

La solution retournée par notre programme linéaire est un optimum global égal à 3.84 :

- $Match(G_1, G_2, G_3) \rightarrow \{(v_{1_2}, v_{2_1}) = 0.7\ , (v_{1_2}, v_{3_1}) = 0.52, (v_{2_1}, v_{3_1}) = 0.62, (v_{2_2}, v_{3_2}) = 1, (v_{2_3}, v_{3_3}) = 1\}$.

D'après ces résultats, nous observons que l'optimum global est meilleur que la somme des optima locaux.

3.1 Préparation des données

L'objectif de cette phase est de préparer les entrées du programme linéaire à savoir :

- N graphes ordonnés selon un ordre décroissant de leur nombre de noeuds. Cet ordre nous permettra de réduire le nombre de variables de décisions du programme linéaire ;
- N matrices de directions qui serviront pour les contraintes structurelles sur les graphes ;
- $\sum_{i=1}^{N-1}(N-i)$ matrices de similarités qui serviront pour le calcul de la somme de similarités et pour certaines contraintes.

Calcul des matrices de directions : Une matrice de directions notée Dir_i de taille $n_i \times n_i$ est définie pour chaque graphe G_i, $\forall i \in [1, N]$. Elle encode la direction des arcs dans les graphes comme suit :

$$Dir_i = \{dir_{i_{k,l}},\ \forall k \times l \in [1, n_i] \times [1, n_i]\}$$

$$dir_{i_{k,l}} = \begin{cases} 1 & \text{si } e_{i_{k,l}} \in E_i \\ -1 & \text{si } e_{i_{l,k}} \in E_i \\ 0 & \text{sinon} \end{cases}$$

Calcul des matrices de similarités : Une matrice de similarités $Sim_{i,j}$ de taille $n_i \times n_j$ est définie entre deux graphes différents G_i et G_j $\forall i \in [1, N-1]$ et $j \in [i+1, N]$. En général, pour N graphes en entrée, nous calculons $\sum_{i=1}^{N-1}(N-i)$ matrices de similarités. Avant de calculer leurs similarités, les éléments des graphes subissent une phase de pré-traitement qui consiste à extraire les différents termes puis à rechercher la racine de ces termes. Ensuite, nous calculons les mesures de similarités pour chaque matrice. Une mesure de similarités est la valeur maximale de deux mesures terminologiques et de deux mesures linguistiques. Les mesures terminologiques choisies sont Jaccard (Jaccard, 1912) et cosinus. Les mesures terminologiques (de catégorie sémantique) choisies sont Wup (Wu et Palmer., 1994) et Lin (Lin, 1998). Ces dernières utilisent le dictionnaire Wordnet.

Pour chaque graphe G_i et G_j, la mesure de similarité est calculée entre toutes les combinaisons des paires de noeuds v_{i_k} et v_{j_l} appartenant respectivement à G_i et G_j. La mesure de similarité est définie comme suit :

$$sim_{i_k,j_l} = max(Jacc(v_{i_k}, v_{j_l}), Cosinus(v_{i_k}, v_{j_l}), Wup(v_{i_k}, v_{j_l}), Lin(v_{i_k}, v_{j_l}))$$

Ainsi chaque matrice de similarité est définie comme suit :

$$Sim_{i,j} = \{sim_{i_k,j_l}, \forall k \in [1, n_i], \forall l \in [1, n_j]\}$$

3.2 Un Programme Linéaire pour le Matching des Graphes

Un programme linéaire est composé d'une fonction objective et de contraintes linéaires définies entre des variables de décisions. Les contraintes sont souvent équivalentes à des implications logiques. Nous nous référons aux travaux de (Plastria, 2002) qui expliquent comment modéliser des implications logiques sous la forme de contraintes linéaires comme le montre le théorème 1 :

Théorème 1. Soit x_i une variable 0-1 $\forall i$ appartenant à un ensemble fini I et soit y une variable 0-1, les deux expressions suivantes sont équivalentes :

$$\{\textbf{Si } \ x_i = 0 \ \ \forall i \in I \ \ \textbf{Alors } \ y = 0\} \Longleftrightarrow \{y \leq \sum_{i \in I} x_i\}.$$

3.2.1 Variables de décisions

LP4HM possède un seul type de variables de décision. Ce type exprime la possibilité ou pas d'avoir un matching entre deux noeuds appartenant à deux graphes différents.

Pour chaque graphe G_i et G_j, $\forall i \in [1, N-1], j \in [i+1, N]$, nous notons x_{i_k,j_l} une variable de décision binaire qui est égale à 1 s'il y a une correspondance entre le noeud v_{i_k} du graphe G_i et le noeud v_{j_l} du graphe G_j et 0 sinon.

Exemple 1. Nous avons 6 variables de décisions entre G_1 et G_2 : deux variables $x_{1_1,2_1}$ et $x_{1_2,2_1}$ dont les similarités attachées ne sont pas nulles et quatre variables $x_{1_1,2_2}$, $x_{1_1,2_3}$, $x_{1_2,2_2}$, $x_{1_2,2_3}$ dont les similarités attachées sont nulles (ne sont pas représentées sur la figure 1).

3.2.2 Contraintes linéaires

LP4HM possède deux catégories de contraintes : (i) des contraintes, notées MS, reliées à la cardinalité 1 :1 des correspondances (c'est-à-dire un élément du schéma source à au plus un seul élément correspondant dans le schéma cible) et à leur similarité qui doit être supérieure ou égale à un seuil global donné, (ii) des contraintes, notées GS, reliées à la structure hiérarchique des graphes. Ces dernières permettent d'imposer deux conditions pour avoir des hiérarchies strictes et des structures cohérentes dans le graphe intégré construit à partir des graphes en entrée.

MS1 (Cardinalité 1 :1 du Matching). Afin d'avoir des correspondances de cardinalité 1 :1, il faut que chaque noeud v_{i_k} du graphe G_i matche au plus avec un seul noeud v_{j_l} du graphe G_j $\forall i \in [1, N-1]$ et $\forall j \in [i+1, N]$ ce qui correspond à la contrainte suivante :

$$\sum_{l=1}^{n_j} x_{i_k,j_l} \leq 1, \quad \forall k \in [1, n_i]$$

Exemple 2. En appliquant MS1 entre G_1 et G_2, les contraintes suivantes sont générées :
- $x_{1_1,2_1} + x_{1_1,2_2} + x_{1_1,2_3} \leq 1$
  (le noeud v_{1_1} de G_1 peut matcher au plus avec un seul noeud de G_2) ;
- $x_{1_2,2_1} + x_{1_2,2_2} + x_{1_2,2_3} \leq 1$
  (le noeud v_{1_2} de G_1 peut matcher au plus avec un seul noeud de G_2) ;
- $x_{1_1,2_1} + x_{1_2,2_1} \leq 1$
  (le noeud v_{2_1} de G_2 peut matcher au plus avec un seul noeud de G_1) ;
- $x_{1_1,2_2} + x_{1_2,2_2} \leq 1$
  (le noeud v_{2_2} de G_2 peut matcher au plus avec un seul noeud de G_1) ;
- $x_{1_1,2_3} + x_{1_2,2_3} \leq 1$
  (le noeud v_{2_3} de G_2 peut matcher au plus avec un seul noeud de G_1).

MS2 (Seuil de Matching). La présence de cette contrainte "facultative" permettra de fixer un seuil pour les correspondances recherchées.

$$\forall i \in [1, N-1], \;\; j \in [i+1, N], \;\; \forall k \in [1, n_i], \; l \in [1, n_j]$$
$$sim_{i_k,j_l} \; x_{i_k,j_l} \geq seuil \; x_{i_k,j_l}$$

Exemple 3. En appliquant MS2 avec un $seuil = 0.5$ entre G_1 et G_2, les contraintes générées sont les suivantes :
- $0.35x_{1_1,2_1} \geq 0.5x_{1_1,2_1}$
  (si $x_{1_1,2_1} = 1$ alors $0.35 \geq 0.5$ (faux) donc $x_{1_1,2_1}$ doit être égale à 0) ;
- $0.7x_{1_2,2_1} \geq 0.5x_{1_2,2_1}$
  (si $x_{1_2,2_1} = 1$ alors $0.7 \geq 0.5$ (vrai) et si $x_{1_2,2_1} = 0$ alors $0 \geq 0$ (vrai)).

GS1 (Hiérarchie stricte). Les graphes en entrée sont composés de hiérarchies strictes. L'objectif est d'obtenir des hiérarchies intégrées strictes dans le graphe intégré. Ceci est d'une grande importance puisque nous envisageons de définir un entrepôt d'Open Data conformément à un schéma multidimensionnel à partir du graphe intégré. Afin de réaliser cet objectif, nous appliquons le principe suivant "si les noeuds fils matchent alors leurs parents doivent matcher mais si les parents matchent les fils ne sont pas obligés de matcher". La partie droite inférieure de la figure 2 illustre ce principe. $\forall i \in [1, N-1]$,

$j \in [i+1, N]$ et $\forall k \in [1, n_i]$, $l \in [1, n_j]$ la contrainte GS1 est comme suit :

$$x_{i_k,j_l} \leq x_{i_{pred(k)},j_{pred(l)}}$$

Exemple 4. En appliquant GS1 entre G_1 et G_2 , les contraintes générées sont :
- $x_{1_2,2_2} \leq x_{1_1,2_1}$
  (si les fils matchent, c'est-à-dire $x_{1_2,2_2} = 1$, les parents doivent matcher c'est-à-dire $x_{1_1,2_1}$ doit être égale à 1 pour que la contrainte soit valide. Si les parents matchent $x_{1_1,2_1} = 1$ la contrainte sera valide quelque soit la valeur de $x_{1_1,2_1}$ (1 ou 0) c'est-à-dire les fils peuvent ne pas matcher) ;
- $x_{1_2,2_3} \leq x_{1_1,2_1}$.

GS2 (La direction des arcs). L'objectif de cette contrainte est d'obtenir des arcs de directions non-conflictuelles entre les noeuds matchés. L'exemple de la figure 2 illustre ce problème. Dans la partie gauche de la figure 2, nous avons deux graphes G_1 et G_2 avec un ensemble de similarités. Dans la partie droite nous avons les scénarii d'intégration possible qui dépendent des directions des arcs reliant les noeuds. La partie droite supérieure correspond à des scénarii (avec un produit des directions des arcs égal à -1) qui peuvent générer un scénario conflictuel (a) et deux scénarii non conflictuels (b) et (c). La partie droite inférieure montre des scénarii (avec un produit de direction des arcs égale à 1) qui rejoignent la contrainte GS1. Afin d'obtenir des scénarios corrects d'intégration tout en considérant la structure, la contrainte GS2 est la suivante : $\forall i \in [1, N-1]$, $j \in [i+1, N]$ et $\forall k, k' \in [1, n_i]$ $\forall l, l' \in [1, n_j]$

$$x_{i_k,j_l} + x_{i_{k'},j_{l'}} + (dir_{i_{k,k'}} dir_{j_{l,l'}}) \leq 0$$

Exemple 5. En appliquant GS1 entre G_1 et G_2 une des contraintes générées est la suivante : $x_{1_1,2_2} + x_{1_2,2_1} + dir_{1_1,1_2} dir_{2_2,2_1} \leq 0$
En substituant $dir_{1_1,1_2} = 1$ et $dir_{2_2,2_1} = -1$ la contrainte devient : $x_{1_1,2_2} + x_{1_2,2_1} \leq 1$
Cette contrainte interdit d'avoir $x_{1_1,2_2}$ = 1 et $x_{1_2,2_1}$= 1 en même temps puisque si c'est le cas nous aurons $1 + 1 \leq 1$ ce qui n'est pas valide. Cela revient à dire que nous ne pouvons pas matcher en même temps v_{1_1} avec v_{2_2} et v_{1_2} avec v_{2_1}. Par contre cette contrainte permet d'avoir l'une des variables à 0 et l'autre à 1 ce qui donnera les cas b et c de la figure 2.

3.2.3 Le modèle résultant

La fonction objective de notre programme linéaire consiste à maximiser la somme des mesures de similarités de toutes les combinaisons de paires de noeuds. Le modèle résultant

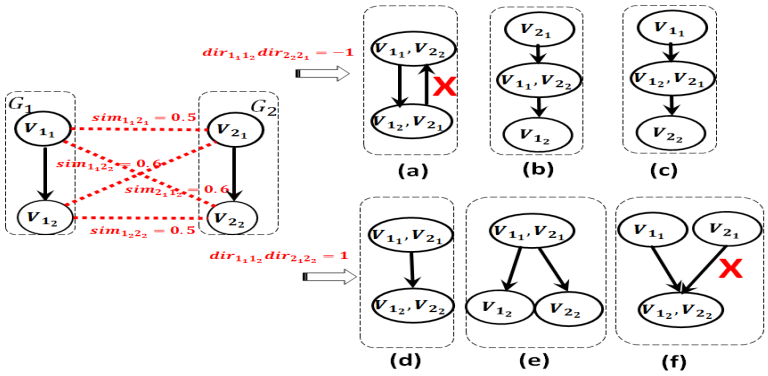


FIG. 2: La direction des arcs dans différents scénarios de matching

complet est comme suit :

$$
\begin{cases}
\max \quad \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} sim_{i_k,j_l}\ x_{i_k,j_l} \\
s.t. \quad \sum_{l=1}^{n_j} x_{i_k,j_l} \leq 1,\ \forall k \in [1,n_i] \qquad (MS1) \\
\qquad\qquad \forall i \in [1,N-1]\ \forall j \in [i+1,N] \\
sim_{i_k,j_l}\ x_{i_k,j_l} \geq seuil\ x_{i_k,j_l} \qquad (MS2) \\
\qquad\qquad \forall i \in [1,N-1]\ \forall j \in [i+1,N] \\
\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j] \\
x_{i_k,j_l} \leq x_{i_{pred(k)},j_{pred(l)}} \qquad (GS1) \\
\qquad\qquad \forall i \in [1,N-1]\ \forall j \in [i+1,N] \\
\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j] \\
x_{i_k,j_l} + x_{i_{k'},j_{l'}} - (dir_{i_{k,k'}} dir_{j_{l,l'}}) \leq 1 \qquad (GS2) \\
\qquad\qquad \forall i \in [1,N-1]\ \forall j \in [i+1,N] \\
\qquad\qquad \forall k,k' \in [1,n_i],\ \forall l,l' \in [1,n_j] \\
x_{i_k,j_l} \in \{0,1\}\ \ \forall i \in [1,N-1]\ \forall j \in [i+1,N] \\
\qquad\qquad \forall k \in [1,n_i],\ \forall l \in [1,n_j]
\end{cases}
$$

Ce modèle comporte :
- $\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} n_i n_j$ variables de décisions ;
- $\sum_{i=1}^{N-1} n_i\ (N-i)$ contraintes de type MS1 ;
- $\sum_{i=1}^{N-1} n_i - 1$ contraintes de type MS2 ;
- Au plus $\sum_{i=1}^{N-1} n_i - 1$ contraintes de type GS1 ;

TAB. 1: Performance du programme linéaire

Nombre de noeuds (Nombre de graphes en entrée)	Nombre de variables de décisions	Nombre de contraintes	Temps de résolution (sec)	Nombre de noeuds matchés
415 (9)	1745	119914	0,75	785
564 (10)	2160	184777	2,81	903
1119 (17)	7310	643807	9,28	3094
1735 (21)	15202	1496397	26,78	6654
3048 (34)	47533	601254	55,14	22971
3827 (38)	61110	7281554	222,25	30129
3973 (39)	64937	7849671	609,91	31943

– Au plus $\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} |E_i||E_j|$ contraintes de type GS2.

L'originalité de notre approche réside dans le fait de pouvoir, à partir d'un même modèle, résoudre deux problèmes connus en matching des schémas.

Le premier problème est celui du choix du seuil de similarité, nous avons constaté que toutes les approches de la littérature doivent fixer des seuils de similarités et paramétrer leur approches afin d'améliorer la qualité de leurs résultats. Nous avons alors défini deux stratégies d'intégration à partir de notre modèle. La première est complètement indépendante du seuil de similarité. En effet, cette stratégie n'utilise pas la contrainte MS2 (dépendante du seuil de similarité) mais uniquement les contraintes MS1, GS1 et GS2. La deuxième stratégie dépend du seuil de similarité globale qui est par défaut la médiane des similarités maximales dans les différentes matrices de similarités ou bien une valeur donnée par l'utilisateur. La première stratégie permet d'épargner un temps important de configuration et d'adaptation de l'outil en fonction des données. Cette stratégie est très avantageuse pour faire face à la diversité des Open Data ainsi qu'au manque d'expertise de l'utilisateur.

Le deuxième problème consiste à proposer des correspondances complexes de cardinalité n :m. La flexibilité de notre approche permet de proposer une solution à ce problème. En effet, il suffit de relaxer les variables de décisions binaires en variables fractionnaires dans l'intervalle [0,1]. Cela impacte en particulier la contrainte MS1 qui en variables binaires permet de matcher un noeud avec uniquement un seul noeud et en variables fractionnaires permet d'affecter à un noeud les noeuds les plus proches. Notre modèle relaxé, noté LP4HM(relaxé), permet de retourner des correspondances de cardinalité n :m. Deux stratégies d'intégration peuvent être définies pour ce modèle relaxé comme nous l'avons décrit pour le modèle LP4HM.

4 Expérimentations

Afin d'évaluer notre approche, nous avons conduit deux types d'expérimentations. La première étudie la performance du temps de résolution de notre modèle par rapport à la taille et au nombre de graphes d'Open Data. La deuxième compare la qualité du matching fournie par notre approche par rapport à la qualité fournie par les approches de référence que sont OMA++, BMatch et Similarity Flooding (SF). Les expérimentations ont été réalisées sur un ordinateur Dell (windows 8, Intel(R) Core i5, 2,30 Ghz processor, 8 Go RAM) et le modèle a été résolu par le solveur CPLEX en version académique.

4.1 Performance du Matching

Les expérimentations sur la performance ont été menées sur des graphes d'OD transformés à partir des sources tabulaires présentes dans ces liens [3]. Le tableau 1 détaille les résultats de ces expérimentations. Nous pouvons remarquer que malgré la taille importante du problème généré (nombre de variables de décisions et nombre de contraintes), LP4HM arrive à résoudre ces problèmes efficacement. En effet, la complexité du temps de résolution à une tendance polynomiale en $O(n^4)$ avec n le nombre des noeuds des graphes. Nous constatons que pour des graphes de grande taille par exemple 3973 noeuds pour 39 graphes en entrée le programme linéaire trouve une solution optimale en 609,91 sec $\rightsquigarrow$ 10,17 mn ce qui est un temps de résolution raisonnable. D'autant plus si nous comparons cette automatisation par rapport au temps qui serait nécessaire à un concepteur pour effectuer cette tâche manuellement. Il faut aussi noter que notre approche fournit une solution unique globalement optimale (sur les 39 graphes) ce qui là encore facilite le travail d'intégration en évitant la comparaison de plusieurs solutions localement optimales comme par exemple COMA++ ou SF.

4.2 Qualité du Matching

Afin d'étudier la qualité des matchings de notre approche, nous avons mené une étude comparative avec les outils COMA++, BMatch et RONDO (qui implémente Similarity Flooding) sur un benchmark existant orienté utilisateurs proposé par (Melnik et al., 2002) et disponible sur ce lien [4]. Le benchmark est composé de 9 tâches (schémas xml et schémas relationnels) et de 7 utilisateurs qui ont proposé des matching, dont la cardinalité est de n :m, pour chaque tâche. Le tableau 2 synthétise trois caractéristiques de ces tâches : (1) l'hétérogénéité des labels des éléments répartie sur trois variantes "faible" ($[0, 0.3[$), "moyenne" ($[0.3, 0.6[$) et "forte" ($[0.6, 1]$) ; les intervalles représentent l'estimation de l'hétérogénéité calculée en fonction de l'écart entre la médiane des maxima des distances terminologiques et la médiane des maxima des distances linguistiques que nous utilisons, (2) la structure des schémas qui est définie en fonction de la profondeur des schémas "plate" (profondeur <3) ou "imbriquée" (profondeur >=3), (3) le rapport entre les deux schémas calculé en divisant le nombre d'éléments du schéma le plus court sur le nombre d'éléments du schéma le plus long et réparti sur trois variantes qui sont faible ($[0.6, 1]$), moyen ($[0.3, 0.6[$), fort ($[0, 0.3[$).

Nous avons traduit les schémas du benchmark selon le format interne des graphes de notre approche. Ensuite nous avons calculé les mesures de précision, rappel, F-Measure, Acccuracy proposée par (Melnik et al., 2002) et HSR proposée par (Duchateau et Bellahsene, 2014) pour les approches suivantes : les deux stratégies indépendantes du choix du seuil de similarité de LP4HM et LP4HM(Relaxé), COMA++ (en combinant toutes les stratégies qu'ils proposent sauf celle de la réutilisation et de la fragmentation), BMatch avec le paramétrage par défaut et Similarity Flooding [5] avec le paramétrage par défaut de l'outil RONDO qui implémente cette approche. Les mesures Accuracy et HSR permettent d'évaluer la quantité de travail qu'un utilisateur donné pourrait économiser en utilisant les solutions automatiques proposées par les

3. `http://data.worldbank.org/country/united-kingdom`, `http://data.worldbank.org/indicator`, `https://www.gov.uk/government/statistical-data-sets/structure-of-the-agricultural-industry-in-england-and-the-uk-at-june`

4. `http://infolab.stanford.edu/~melnik/mm/sfa/`

5. Les mesures d'accuracy que nous avons trouvées sont légèrement différentes des mesures publiées dans (Melnik et al., 2002), nous pensons que cela provient du paramétrage par défaut.

TAB. 2: Caractéristiques des schémas des différentes tâches

	Hétérogénéité des éléments			Structure		Ecart entre schémas		
	faible	moyenne	forte	plate	imbriquée	faible	moyen	fort
Tâche 1	×			×		×		
Tâche 2		×			×	×		
Tâche 3			×		×	×		
Tâche 4	×				×	×		
Tâche 5			×		×	×		
Tâche 6			×		×	×		
Tâche 7			×	×			×	
Tâche 8			×	×			×	
Tâche 9			×	×				×

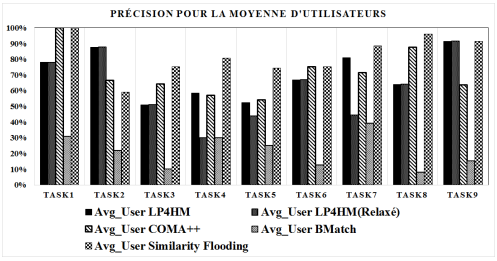
outils. La mesure HSR est plus significative que la mesure Accuracy puisque la première révèle mieux l'effort épargné comme elle implique le nombre d'éléments des schémas (Duchateau et Bellahsene, 2014). Alors que la deuxième qui se base uniquement sur la précision et le rappel devient non représentative si la précision est $\leq 50\%$ (Melnik et al., 2002).

Le tableau 3 résume les résultats pour la moyenne par utilisateur et par tâche de chacune des approches. Nous pouvons constater que les deux versions de notre approche sont les premières pour les mesures de rappel et de HSR. Sur la F-Measure, l'Accuracy et la précision Similarity Flooding occupe la première place. Cependant les résultats sont proches et nous rappelons que la configuration par défaut de SF admet un seuil de similarité de 1 ce qui explique des valeurs élevées de la précision et par conséquence de l'accuracy et la F-Measure. Il est à noter que les deux versions que nous avons utilisées sont indépendantes du seuil de similarité ce qui démarque notre solution par rapport aux trois approches COMA++, BMatch et SF qui utilisent toutes des seuils de similarité pour obtenir le meilleur de leur résultats. Cela épargne à l'utilisateur le processus fastidieux et difficile de recherche de la meilleure configuration de l'outil de matching. D'autre part, ces expérimentations confirment l'efficacité de la recherche d'une solution optimale pour le problème de matching. En effet, nous sommes les premiers sur les résultats de rappel malgré les choix variés des distances de similarités entre les approches.

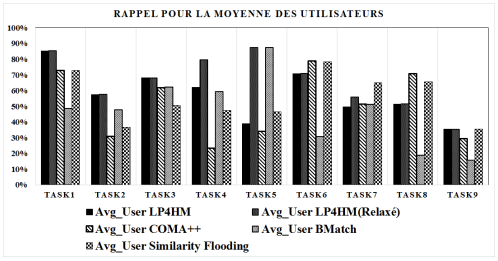
TAB. 3: Les différentes moyennes des mesures de qualité sur les 9 tâches et les 7 utilisateurs

	Précision (%)	Rappel (%)	F-Measure (%)	Accuracy (%)	HSR (%)
LP4HM	67	58	62	30	**81**
LP4HM(relaxé)	58	**66**	60	23	**81**
COMA++	72	50	58	32	76
BMatch	22	47	28	0	69
Similarity Flooding	**81**	55	**65**	**43**	80

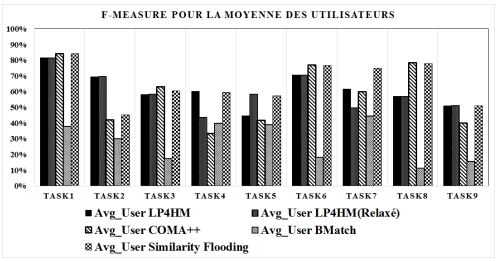
Concernant les résultats détaillés, les figures 3a–3e montrent respectivement les mesures de précision, rappel, F-Measure, Accuracy et HSR pour la moyenne des utilisateurs sur les différentes tâches. Pour la tâche 1 qui est la plus simple vu la faible hétérogénéité, sa structure plate, le faible écart entre les éléments des schémas, nous avons constaté que tous les utilisateurs à part le septième utilisateur ont proposé des correspondances de type 1 :1, les solutions des deux versions de notre approche étaient les mêmes et obtiennent des résultats de rappel et de précision de l'ordre de 80%. Pour la tâche 2, avec une structure imbriquée et une hété-



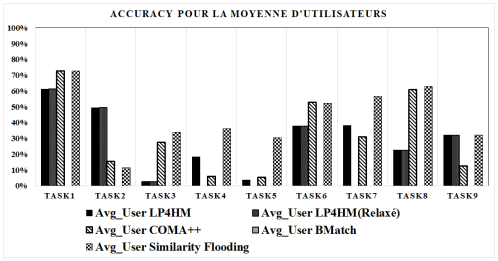
(a) Les résultats de la Précision



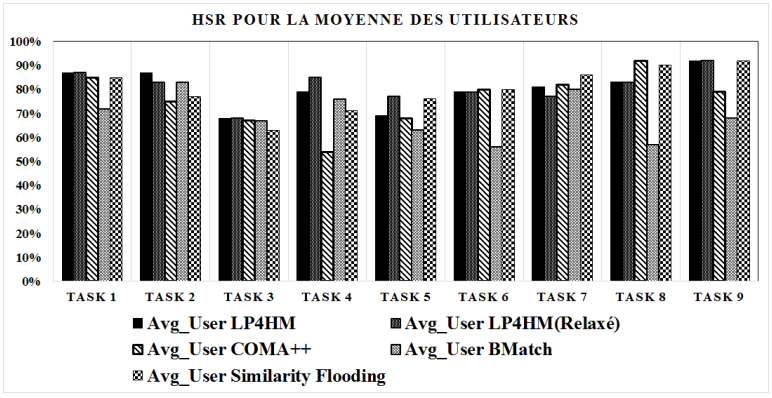
(b) Les résultats du Rappel



(c) Les résultats de la F-Measure



(d) Les résultats de l'Accuracy



(e) Les résultats du HSR

FIG. 3: Une comparaison des mesures de qualité pour les approches : LP4HM, LP4HM(Relaxé), COMA++, BMatch, Similarity Flooding

rogénéité moyenne, les utilisateurs avaient proposé des choix assez différents sur les parties similaires des schémas, toutefois nos résultats en moyenne pour la précision et le rappel ont coïncidé avec au moins la moitié des utilisateurs.

Pour les tâches 3, 4, 5 et 6 qui sont toutes de forte hétérogénéité, de structure imbriquée, de faible écart, les utilisateurs ont fait plusieurs matching de type 1 :n, la version LP4HM(relaxé) de notre approche obtient des résultats de rappel meilleurs que la version LP4HM et la plupart des autres approches mais une précision faible entre 30% et 60%. Enfin les tâches 7, 8 et en particulier la 9 n'est pas évidente pour les outils de matching puisque il y a un écart important

entre les éléments des schémas. Nous remarquons que les résultats de précision sont importantes. Par contre, les résultats de rappels ne dépassent pas les 50% pour toutes les approches.

Nous pouvons constater que expérimentalement la mesure HSR est plus significative que la mesure accuracy [6]. En effet, l'approche BMatch qui a des résultats nuls pour l'accuracy obtient de bons résultats de HSR. Pour conclure, notre approche dans ces deux variantes LP4HM et LP4HM(relaxé) montre expérimentalement sur différents utilisateurs et différentes tâches une bonne qualité de matching aussi bien sur des schémas de forte ou de faible hétérogénéité et de structure imbriquée ou plate. Ces dernières font partie des caractéristiques des Open Data.

5 Conclusion

Cet article présente une approche holistique pour l'intégration automatique des graphes d'Open Data. L'intégration se focalise sur des graphes hiérarchiques dans l'objectif de définir un schéma multidimensionnel permettant l'entreposage de ces Open Data. La solution proposée est un programme linéaire en variables 0-1 qui englobe des contraintes sur la structure des graphes et sur la cardinalité des correspondances. L'évaluation de notre approche montre un temps de résolution raisonnable en fonction du nombre et de la taille des graphes. La qualité de nos matchings est tout à fait satisfaisante également. En effet, nos comparaisons nous placent parmi les meilleures approches malgré (i) certains choix spécifiques au contexte des hiérarchies et (ii) la non-utilisation du seuil de similarité. En plus, la flexibilité de notre modèle, nous a permis de proposer des stratégies (relaxés ou non relaxés) qui s'adaptent à l'expertise des utilisateurs par rapport à la cardinalité des matchings. Pour nos travaux futurs, nous étudierons l'extension de notre modèle pour intégrer des graphes labellisés (ontologies).

Références

Aumueller, D., H.-H. Do, S. Massmann, et E. Rahm (2005). Schema and ontology matching with COMA++. In *Proceedings of the ACM International Conference on Management of Data*, SIGMOD '05, pp. 906–908.

Benharkat, A., R. Rifaieh, S. Sellami, M. Boukhebouze, et Y. Amghar (2007). PLASMA : A platform for schema matching and management. *IBIS 5*, 9–20.

Bernstein, P. A., J. Madhavan, et E. Rahm (2011). Generic schema matching, ten years later. *PVLDB 4*(11), 695–701.

Berro, A., I. Megdiche, et O. Teste (2013). Vers l'int égration multidimensionnelle d'open data dans les entrepôts de données. In *Actes des 9èmes journées francophones sur les Entrepôts de Données et l'Analyse en ligne, EDA' 2013*, pp. 95–104.

Berro, A., I. Megdiche, et O. Teste (2014). A content-driven ETL processes for open data. In *New Trends in Database and Information Systems II, Selected papers of the 18th East European Conference on Advances in Databases and Information Systems, ADBIS'14*, pp. 29–40. Springer.

Duchateau, F. et Z. Bellahsene (2014). Designing a benchmark for the assessment of schema matching tools. *Open Journal of Databases (OJDB) 1*(1), 3–25.

6. Nous avons arrondi toutes valeurs strictement négative d'accuracy à 0.

Duchateau, F., Z. Bellahsene, et M. Roche (2007). Bmatch : a semantically context-based tool enhanced by an indexing structure to accelerate schema matching. In *23èmes Journées Bases de Données Avancées, BDA' 2007*.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New Phytologist 11*(2), 37–50.

Lin, D. (1998). An information-theoretic definition of similarity. In *In Proceedings of the 15th International Conference on Machine Learning*, pp. 296–304. Morgan Kaufmann.

Malinowski, E. et E. Zimányi (2006). Hierarchies in a multidimensional model : From conceptual modeling to logical representation. *Data Knowl. Eng. 59*(2), 348–377.

Melnik, S., H. Garcia-Molina, et E. Rahm (2002). Similarity flooding : A versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th International Conference on Data Engineering, ICDE' 2002*, pp. 117–128.

Nash, S. G. et A. Sofer (1996). *Linear and Nonlinear Programming* (First ed.). McGraw-Hill Series in Industrial Engineering and Management Science. New York : McGraw-Hill Inc.

Plastria, F. (2002). Formulating logical implications in combinatorial optimisation. *European Journal of Operational Research 140*(2), 338 – 353.

Rahm, E. (2011). Towards large-scale schema and ontology matching. In Z. Bellahsene, A. Bonifati, et E. Rahm (Eds.), *Schema Matching and Mapping*, Data-Centric Systems and Applications, pp. 3–27. Springer Berlin Heidelberg.

Rahm, E. et P. A. Bernstein (2001). A survey of approaches to automatic schema matching. *The VLDB Journal 10*(4), 334–350.

Saleem, K., Z. Bellahsene, et E. Hunt (2007). Performance oriented schema matching. In *18th International Conference Database and Expert Systems Applications, DEXA' 2007*, pp. 844–853.

Shvaiko, P. et J. Euzenat (2005). A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, pp. 146–171. Springer Berlin Heidelberg.

Su, W., J. Wang, et F. Lochovsky (2006). Holistic schema matching for web query interfaces. In *10th International Conference on Extending Database Technology, EDBT' 2006*, pp. 77–94.

Wu, Z. et M. Palmer. (1994). Verb semantics and lexical selection. In *In 32nd. Annual Meeting of the Association for Computational Linguistics*, pp. 133–138.

Summary

In this paper, we propose a holistic approach to integrate Open Data graphs based on a 0-1 linear program. The latter aims to maximize the similarity profit for a reduced number of combinations of pair graphs constructed from several input graphs. The linear program encompasses constraints on the matching cardinality and on the graphs' structures. To our knowledge, our approach is the first to provide a global optimal solution for the holistic matching problem with a reasonable resolution time. The quality of our approach results are satisfactory compared to other approaches in the literature.