Laboratoire de Recherche en Informatique — U.M.R CNRS 8623 — Université de Paris-Sud

Intégration Sémantique de Données guidée par une Ontologie

THÈSE

présentée et soutenue publiquement le 07 décembre 2007

pour l'obtention du

Doctorat de l'Université Paris-Sud

Spécialité : informatique

par

Fatiha SAÏS

Composition du jury

Rapporteurs :	Daniel Kayser	LIPN – Université Paris Nord 13 (Professeur)
	Jean-Marc Petit	INSA – Lyon, LIRIS – Université Lyon (Professeur)
Examinateurs :	Patrice Buche	INRA – UR 1204 Mét@risk (Ingénieur de Recherche HDR)
	Christine Froidevaux	LRI – Université Paris-Sud (Professeur)
	Hélène Gagliardi	LRI – Université Paris-Sud (Maître de conférences)
Directrice :	Marie-Christine Rousset	LIG – Université Grenoble 1 (Professeur)
Co-directrice :	Nathalie Pernelle	LRI – Université Paris-Sud (Maître de conférences)
Membre invité :	Alain Léger	Orange – France Télécom R&D (Directeur de Recherche)

Université Paris-Sud, 91405 Orsay Cedex, France

Mis en page avec la classe thloria.

Table des matières

Introduction générale **vii**

1 Systèmes d'intégration de données . . . vii
2 Intégration sémantique de données . . . ix
3 Problématiques abordées . . . xi
4 Approche et principales contributions . . . xii
5 Plan du manuscrit . . . xiv

Partie I Réconciliation de données avec un schéma **3**

Introduction **5**

Chapitre 1
Présentation du contexte

1.1 Méthodes de réconciliation de schémas exploitant les données . . . 9
1.1.1 Problème d'alignement et différentes mesures de similarité . . . 10
1.1.2 iMAP : approche qui exploite les contraintes du domaine . . . 12
1.1.3 Approche supervisée pour la découverte d'appariements entre schémas 14
1.1.4 FCA-Merge : approche fondée sur l'analyse formelle de concepts . . . 16
1.1.5 Peuplement d'ontologies à partir de tableaux . . . 17
1.1.6 Synthèse . . . 19
1.2 Contexte applicatif : le projet e.dot . . . 21
1.2.1 La base de données Sym'Previus . . . 21
1.2.2 Chaîne de traitements des informations du Web . . . 23
1.2.3 Tâches d'acquisition et de pré-traitement des documents du Web . . . 25

Chapitre 2
Enrichissement sémantique d'informations structurées

2.1 Méthode d'enrichissement sémantique de tableaux 28
2.1.1 Présentation du format SML 28
2.1.2 Procédures d'appariement des données avec le contenu de l'ontologie . 30
2.1.3 Identification des colonnes du tableau 31
2.1.4 Identification des relations sémantiques présentées dans les tableaux . 33
2.1.5 Instantiation des relations sémantiques identifiées 38
2.1.6 Trace du contexte et des traitements 41
2.2 Algorithme d'enrichissement sémantique de tableaux 43
2.3 Génération automatique de la DTD SML à partir de l'ontologie 47
2.4 Découverte de relations candidates à l'alimentation d'un entrepôt thématique 48
2.4.1 Capture de l'information non reconnue dans des attributs supplémentaires 49
2.4.2 Représentation de tableaux par une relation générique 50
2.5 Interrogation des documents SML 51
2.5.1 Exemple d'interrogation 52
2.5.2 Exploitation des attributs supplémentaires 52
2.5.3 Exploitation des relations génériques 53
2.5.4 Interrogation par les vues pré-définies dans MIEL++ 53
2.6 Conclusion 55

Chapitre 3
Développement et expérimentations

3.1 Développement 57
3.2 Protocole d'évaluation des résultats de la méthode XTab2SML 59
3.2.1 Chaîne de traitements des documents évalués 59
3.2.2 Méthodologie d'évaluation 59
3.3 Présentation des résultats 61
3.3.1 Résultats de l'enrichissement sémantique 62
3.3.2 Résultats de la découverte de relations candidates 63
3.4 Conclusion 65

Partie II Réconciliation de données conformes à un même schéma 67

Introduction 69

Chapitre 4
État de l'art

4.1 Introduction de l'état de l'art . . . 74
4.2 Mesure de similarité pour la réconciliation de références . . . 76
4.2.1 Mesures fondées sur les caractères . . . 76
4.2.2 Mesures fondées sur les chaînes atomiques . . . 77
4.2.3 Mesures hybrides . . . 79
4.2.4 Réconciliation de références fondée sur la description non structurée des références . . . 80
4.2.5 Discussion . . . 81
4.3 Réconciliation de références et exploitation des relations entre données . . . 83
4.3.1 Méthodes locales de réconciliation de références . . . 83
4.3.2 Méthodes globales de réconciliation de références . . . 90
4.3.3 Discussion . . . 102
4.4 Réconciliation de références et exploitation de connaissances pour la réconciliation . . . 103
4.4.1 Méthodes supervisées de réconciliation de références . . . 103
4.4.2 Méthodes déclaratives de réconciliation de références . . . 107
4.4.3 Discussion . . . 111
4.5 Réduction du nombre de comparaisons de références . . . 112
4.6 Synthèse . . . 114

Chapitre 5
Modèle de données

5.1 Langages d'ontologies . . . 117
5.1.1 Langages RDF et RDFS . . . 117
5.1.2 Langage OWL . . . 119
5.1.3 Langage SWRL . . . 120
5.2 Le modèle de données RDFS+ . . . 122

5.2.1 Représentation du schéma et des axiomes du schéma 122
5.2.2 Représentation des données et de leurs axiomes 125

Chapitre 6
L2R : une méthode Logique pour la Réconciliation de Références

6.1 Problème de réconciliation de références considéré dans L2R 128
6.2 Traduction des axiomes du schéma et des données par des règles logiques . . 128
6.2.1 Traduction des axiomes sur les données 129
6.2.2 Traduction des axiomes sur le schéma 129
6.3 Méthode de raisonnement pour la réconciliation de références 131
6.3.1 Connaissances factuelles 131
6.3.2 Algorithme de réconciliation de références fondé sur la résolution . . . 132
6.3.3 Illustration de la méthode sur un exemple 137
6.4 Génération des dictionnaires de (non) synonymies 143
6.4.1 Dictionnaire de synonymies 144
6.4.2 Dictionnaire de non synonymies 145
6.4.3 Représentation des dictionnaires 145
6.5 Expérimentation 145
6.5.1 Présentation des jeux de données testés 146
6.5.2 Evaluation des résultats 147
6.6 Conclusion 150

Chapitre 7
N2R : une méthode Numérique pour la Réconciliation de Références

7.1 Problème de réconciliation de références considéré dans N2R 154
7.2 Comparaison des paires de références 155
7.2.1 Exploitation des connaissances du domaine pour le filtrage 155
7.2.2 Description commune des paires de références 156
7.2.3 Calcul du score de similarité des valeurs et des ensembles de valeurs . 157
7.3 Expression du calcul de similarité dans un système d'équations 159
7.4 Calcul de similarité informé par les connaissances du schéma 162
7.4.1 Exploitation des axiomes de fonctionnalité du schéma 162
7.4.2 Construction du système d'équations non linéaire 165
7.4.3 Résolution du système d'équations non linéaire 170
7.4.4 Illustration du calcul itératif sur un exemple 172
7.5 Algorithme de calcul itératif de la similarité des références 175
7.5.1 Présentation de l'algorithme général 175

7.5.2 Complexité de l'algorithme du calcul itératif de similarité 181
7.6 Expérimentations 182
7.6.1 Résultats de N2R obtenus sur le jeu de données *Cora* 183
7.6.2 Résultats de N2R obtenus sur le jeu de données FT_HOTELS 184
7.7 Conclusion 186

Conclusion et Perspectives **187**
7.1 Résumé des contributions et discussion 187
7.2 Travaux en cours et perspectives 190

Table des figures **194**

Liste des tableaux **196**

Annexe A
Preuve de convergence du calcul de similarité

A.1 Scores de similarité des paires de références bornés par 1 198
A.2 Fonction monotone de calcul de similarité 200
A.3 Convergence vers un point fixe 203

Annexe B
L'ontologie Sym'Previus

Annexe C
Document XTab

C.1 Exemple de document XTab 207
C.2 La DTD XTab 208

Annexe D
La DTD SML (Semantic Markup Language)

Annexe E
Requêtes d'interrogation de MIEL++

E.1 Requête sur la vue *Aliment/effet du facteur analysé* 211
E.2 Requête sur la vue *charge microbienne* 212

Annexe F
La description WSDL du service Web XTab2SML

Bibliographie **215**

Introduction générale

L'expansion du nombre de sources d'information disponibles sur le Web et la quantité de données gérées au sein des bases de données des entreprises a rendu indispensable l'intégration de données provenant de différentes sources. Dans le domaine des sciences de la vie, des centaines de sources de données médicales sont disponibles et des centaines de milliers de sources de données accessibles publiquement via le Web. Cette évolution quantitative des données et la diversité des formats des sources ont conduit les entreprises à consacrer 35 % de leur budget dédié aux technologies de l'information pour l'intégration de données. Ces sources de données sont le plus souvent *réparties*, *autonomes* et *hétérogènes*.

Les sources d'informations sont *réparties* : de plus en plus d'informations sont créées partout dans le monde et publiées sur le Web, de nombreuses entreprises ont des ramifications dans plusieurs pays et les états décentralisent leurs administrations. Elles sont *autonomes* car les sources de données sont conçues par différentes personnes, à différents moments et pour répondre à différents besoins applicatifs. Enfin, les sources d'informations sont *hétérogènes* : des logiciels différents sont utilisés pour créer et gérer les données (e.g. Oracle, O2, MySql), les données sont publiées dans des formats divers (e.g. HTML, PDF, images) et des modèles de données différents sont utilisés pour les représenter (e.g. modèles relationnel, objet, semi-structuré).

Un système d'intégration de données a pour rôle d'offrir à l'utilisateur une vue uniforme et une interrogation transparente des informations sans que l'utilisateur n'ait le souci de la provenance des informations ni de leur format d'origine.

1 Systèmes d'intégration de données

Le problème de combiner des sources de données hétérogènes et de les interroger via une seule interface de requête ne date pas d'aujourd'hui. Depuis l'arrivée et l'adoption des bases de données dans les années soixante, leur partage et leur combinaison sont naturellement devenus indispensables. Cette combinaison peut être effectuée de différentes façons et à différents niveaux de l'architecture du système. Deux approches principales pour la conception des systèmes d'intégration ont été définies en se fondant sur la localisation des données gérées par le système :

- *l'intégration virtuelle de données*, où la vue unifiée est virtuelle et les données restent stockées dans les sources d'origine. L'architecture type pour l'intégration virtuelle de données est l'architecture *médiateur* ;

- *l'intégration matérialisée de données*, où la vue unifiée des données est matérialisée et les données sont rapatriées des sources d'origine et stockées dans un *entrepôt* de données.

Dans l'approche médiateur, l'intégration de données est fondée sur la définition d'un schéma global unifiant les schémas hétérogènes des sources à intégrer et sur la description homogène et abstraite, du contenu des sources par des vues. Une première proposition d'architecture médiateur a été proposée dans [Wie92] ; celle-ci est représentée par trois couches, comme nous le montrons en Figure 1 : *médiateur*, *adaptateurs* et *sources*. Au niveau du *médiateur*, le schéma global fournit un vocabulaire unique pour l'expression des requêtes des utilisateurs et pour la description des contenus des sources par un ensemble de vues abstraites sur les sources. Les adaptateurs ou *wrappers* traduisent les requêtes exprimées en termes du vocabulaire du schéma global en des requêtes exprimées dans le langage des sources. De plus, les adaptateurs transforment les réponses aux requêtes en des réponses conformes au schéma global du médiateur.

Pour l'interrogation des données dans un médiateur, des mises en correspondance doivent être définies entre les relations du schéma global et les relations du schéma local. Il existe deux approches pour mettre en correspondance les relations du schéma global et celles du schéma local [Len02]. La première est l'approche *Global As View* (GAV) où les relations du schéma global sont exprimées en fonction des relations du schéma local (source) et la seconde est l'approche *Local As View* (LAV) où les relations du schéma local sont exprimées en fonction des relations du schéma global. Le contenu des sources est décrit par un ensemble de vues sur les relations du schéma global. Pour répondre à la requête de l'utilisateur, le médiateur exploite les vues abstraites qui décrivent le contenu des sources. La tâche du médiateur consiste, en effet, à reformuler les requêtes qui lui sont posées en termes du schéma global en des plans de requêtes exprimées en termes de vues sur les sources de données. Cette tâche de reformulation de la requête est appelée réécriture de requêtes en termes de vues. Les principaux problèmes de l'approche médiateur sont la réécriture des requêtes en termes de vues qui est complexe dans une approche LAV, dans la mise à jour du schéma global lors de l'ajout d'une source dans une approche GAV et dans la conception des adaptateurs.

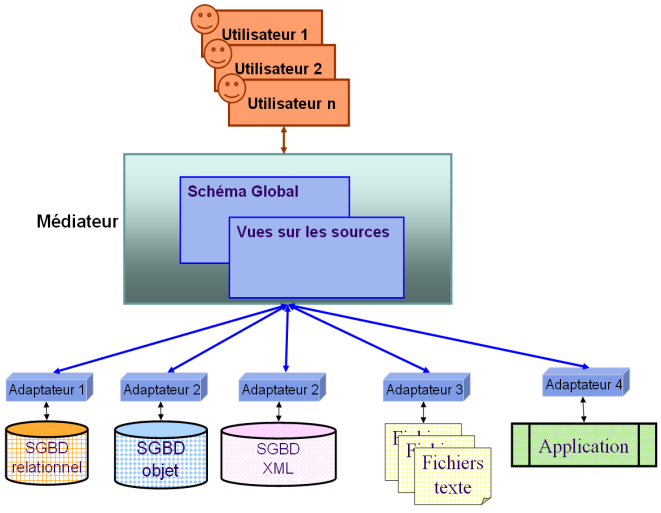

Fig. 1 – *Architecture d'un système médiateur*

L'approche entrepôt de données consiste à voir l'intégration comme la construction d'une base de données réelle appelée entrepôt, stockant des données intégrées provenant de différentes

sources. Dans le contexte d'une entreprise, les entrepôts de données sont mis en oeuvre pour fournir un support pour le processus décisionnel. Un système d'intégration suivant une approche entrepôt de données est constitué de trois niveaux, comme le montre la Figure 2 : le niveau de l'entrepôt de données, le niveau des sources et le niveau des *magasins de données*. L'intégration selon une approche entrepôt de données est fondée sur un schéma global de l'entrepôt fournissant une vue intégrée des sources. Une fois que le schéma de l'entrepôt est conçu, les données sont extraites à partir des sources, transformées au format de représentation des données de l'entrepôt par des *extracteurs*, elles sont éventuellement filtrées pour ne garder que les données pertinentes, et enfin stockées dans l'entrepôt. Les *magasins de données* correspondent à un ensemble de vues sur l'entrepôt qui peuvent être matérialisées ou abstraites. L'interrogation s'appuie sur des techniques classiques d'interrogation du domaine des bases de données. L'utilisateur interagit avec l'entrepôt pour une interrogation directe des données de l'entrepôt ou à travers les *magasins de données* soit pour effectuer de la fouille de données soit pour générer des rapports statistiques. Les principaux problèmes de cette approche sont l'extraction et l'intégration des données permettant d'assurer une bonne qualité des données, comme par exemple, l'absence de données redondantes et la fraîcheur des données. En effet, contrairement à une approche médiateur où les réponses aux requêtes proviennent directement des sources, dans le cadre d'un entrepôt les données matérialisées localement doivent être mises à jour pour assurer la fraîcheur des données.

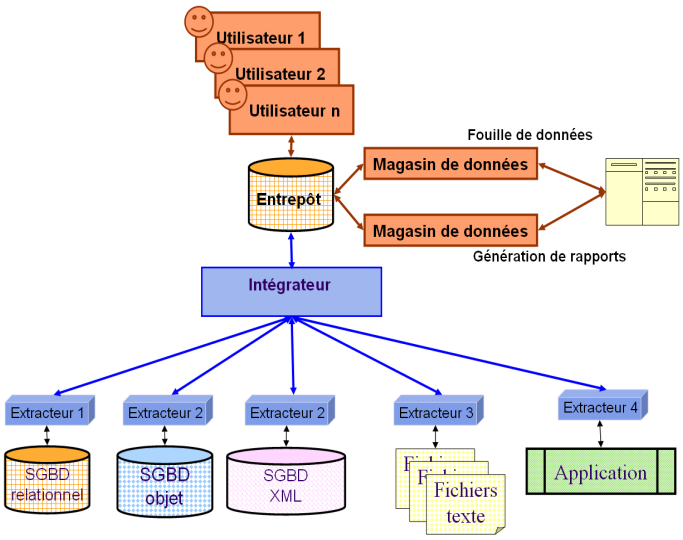

Fig. 2 – *Architecture d'un entrepôt de données*

Une approche mixte d'intégration de données est une approche qui combine à la fois l'approche médiateur et l'approche entrepôt. Il s'agit, par exemple, d'un système médiateur qui intègre plusieurs sources de données externes et qui exploite un entrepôt de données contenant des données conformes au schéma global du médiateur.

2 Intégration sémantique de données

Ce qui fait de l'intégration sémantique de données un défi aujourd'hui est le problème de l'*hétérogénéité* des sources de données qui se pose quelle que soit l'architecture du système d'intégration. L'hétérogénéité des sources de données est généralement classée en deux types : (i)

hétérogénéité des schémas et (ii) *hétérogénéité des données*. L'*hétérogénéité des schémas* ou de structures, apparaît, d'une part, lorsque des modèles de données différents sont utilisés pour décrire les données mais également lorsque des schémas décrits dans un même modèle sont différents. En effet, les choix considérés pour concevoir le schéma de la source de données, tels que les noms des relations, des attributs, des tags, des types de données et le degré de décomposition des attributs, diffèrent d'une source à une autre. La présence de variations dans les structures est inévitable puisque les humains pensent différemment et que les sources de données sont conçues pour des besoins applicatifs distincts. L'*hétérogénéité de données* apparaît lorsque différents vocabulaires et référentiels sont utilisés pour représenter les données, lorsque les informations sont incomplètes c'est-à-dire lorsque certains attributs ne sont pas renseignés, et lorsque les données contiennent des erreurs. Ce type d'hétérogénéité est également inéluctable lorsqu'on souhaite intégrer des données provenant de différentes sources. De plus, certaines sources de données sont alimentées par des informations extraites automatiquement à partir du Web ce qui peut engendrer de multiples erreurs dans les données. Ces erreurs peuvent être dues par exemple à une mauvaise segmentation des informations ou une association incorrecte des valeurs aux attributs.

Pour traiter l'hétérogénéité des schémas et des données, la majorité des travaux menés se contentent de l'exploitation des informations présentes dans les schémas et dans les données. Des techniques de comparaison syntaxique, telle que les mesures de similarité ont été utilisées. Des heuristiques ont également été utilisées, par exemple, le fait que des noms des attributs comportent des informations sur leur sémantique. Pourtant ces derniers peuvent contenir des abréviations, des concaténations de plusieurs noms d'attributs et des homonymes. Par exemple, deux entreprises qui souhaitent fusionner leur base de données ont dans leur schéma respectif des attributs homonymes qui possèdent des sens différents. Dans l'une des deux bases de données, l'attribut "*Dépenses*" représente la somme dépensée en Euro alors que dans l'autre base il correspond au nombre de produits achetés. Les mêmes noms d'attributs peuvent ne pas représenter la même information et n'ont donc pas la même sémantique. Le problème de l'hétérogénéité des schémas et des données ne peut donc pas être résolu simplement avec ce type de méthodes. C'est pour cela que rendre explicite la sémantique précise des données intégrées est essentiel pour avoir une intégration sémantiquement correcte des données. Notre approche est fondée sur l'explicitation et l'exploitation de la sémantique des données à intégrer et des schémas de ces données. Nous parlerons alors d'intégration sémantique de données.

Une première idée pour rendre explicite la sémantique des données et des éléments du schéma des sources de données est de la spécifier de manière exhaustive pour toutes les données et les éléments des schémas considérés. Seulement il est impossible d'envisager de définir complètement la sémantique ou le sens d'une donnée ou d'un élément du schéma dans une base de données [PKB93].

Les ontologies – qui sont définies comme « *Une spécification explicite et formelle d'une conceptualisation commune* », par Thomas Gruber [Gru93] – peuvent contribuer à la résolution du problème de l'hétérogénéité sémantique. En effet, elles offrent une description formelle des concepts et de leurs relations existant dans certains domaines. Un utilisateur ou concepteur peut grouper et déclarer explicitement la sémantique des concepts et des données fournis par les sources. Ainsi, en exploitant cette sémantique explicite et partagée, l'impact nuisible de l'hétérogénéité sémantique peut être réduit. Il s'agit dans ce cas d'une approche d'intégration de données fondée sur les ontologies. Une fois que les données des sources sont rendues conformes à l'ontologie, les requêtes peuvent être exprimées en termes de l'ontologie et les données peuvent être interrogées via une seule interface de requêtes. Dans le cas d'une intégration matérialisée des données, les

requêtes exprimées en termes de l'ontologie sont évaluées directement sur l'entrepôt ; dans le cas d'une intégration virtuelle des données ces requêtes sont réécrites en fonction des vues sur les sources et les extensions de ces vues sont ensuite intégrées et combinées pour pouvoir répondre à l'utilisateur. On note que, par la suite, on utilisera indifféremment les termes ontologie et schéma.

3 Problématiques abordées

Pour concevoir des systèmes d'intégration sémantique de données, nous nous sommes posé deux problèmes principaux qui sont indépendants de l'architecture choisie pour le système d'intégration (entrepôt ou médiateur).

Le premier problème se pose lorsqu'on souhaite *réconcilier des données avec un schéma* : ce qui consiste à déterminer quel élément du schéma est représenté par quel élément de la source de données. Nous considérons des données hétérogènes extraites à partir du Web et structurées par un ensemble d'éléments syntaxiques. Nous supposons disposer d'une ontologie contenant un ensemble de concepts organisés hiérarchiquement et un ensemble de relations n-aires entre les concepts de l'ontologie. L'ontologie contient par exemple une relation entre les concepts <*aliment*, *microorganisme*, *concentration*>. Les données extraites du Web doivent être décrites en utilisant le vocabulaire de cette ontologie afin que cette dernière puisse servir d'interface d'interrogation des sources de données intégrées. Pour atteindre cet objectif les deux problèmes suivants doivent être traités :

1. trouver des mises en correspondance entre les éléments structurels et les valeurs de la source d'une part et les concepts de l'ontologie d'autre part.
2. trouver les relations entre concepts représentées par un ensemble de valeurs (e.g. ensemble d'aliments, ensemble de microorganismes) dans les sources de données.

De plus, pour passer à l'échelle de grands volumes de données, cette méthode doit être la plus automatique possible et donc minimiser l'intervention humaine. Pourtant les variations syntaxiques et les informations incomplètes qui caractérisent les données extraites à partir du Web augmentent la difficulté du problème. Une méthode automatique risque parfois de conduire à des erreurs ou une absence d'interprétations. Pour éviter l'impact de ces dernières, le système doit fournir toutes les mises en correspondance trouvées accompagnées d'éléments des données originales afin de guider l'exploitation du résultat de la réconciliation.

Le second problème se pose lorsqu'on s'intéresse à l'intégration des sources de données conformes à un schéma où, à cause de l'hétérogénéité des données, des représentations différentes peuvent référer la même entité du monde réel, comme par exemple, une personne, un article ou un gène. Plus précisément, nous considérons qu'une donnée est définie par une *référence* – identifiant – et par une description. Il s'agira donc de définir une méthode qui permet de réconcilier des références différentes décrites relativement à un même schéma et ainsi de déterminer celles qui réfèrent la même entité du monde réel et celles qui réfèrent des entités différentes du monde réel. Ce problème de *réconciliation de références* peut surgir dans différentes applications qu'elles soient en ligne comme lors de l'intégration de plusieurs sources de données dans un médiateur ou hors ligne comme lors du nettoyage d'une source de données contenant des redondances.

Dans un contexte où des variations syntaxiques, des erreurs et des informations incomplètes sont présentes dans les descriptions des références, une méthode de réconciliation de références doit pouvoir tirer profit de toutes les informations disponibles : les caractéristiques des sources de données, les connaissances du domaine, telles que les dépendances fonctionnelles ou les disjonctions entre données ainsi que les descriptions d'autres références qui leur sont liées. De plus, la méthode de réconciliation de références doit être la plus automatique possible pour se plier aux contraintes des applications telles que le traitement de volumes importants de données, une exécution en ligne de la réconciliation de références ou une adaptation facile en cas d'ajout de nouvelles sources.

4 Approche et principales contributions

Pour traiter les deux problèmes dégagés dans la section précédente nous avons proposé une approche dans laquelle l'ontologie a un rôle central. Cette approche consiste à exploiter en plus des informations syntaxiques présentes dans les données, les connaissances du domaine déclarées explicitement dans l'ontologie. Dans notre approche, l'exploitation du contenu de l'ontologie a deux rôles : (i) homogénéisation des données hétérogènes en les décrivant relativement au contenu d'une même ontologie en les enrichissant par les concepts, les termes et les relations du domaine représentés dans l'ontologie. Nous parlons donc de l'enrichissement sémantique des données ; (ii) renforcer la tâche de réconciliation par des connaissances du domaine. Ceci nous a conduit à proposer une méthode de réconciliation de références fondée sur la sémantique du domaine.

De plus, le fait que l'approche soit guidée par l'ontologie assure une certaine généricité des tâches de la réconciliation de données avec un schéma et de la réconciliation de références. En effet, en cas de changement ou de mise à jour du domaine il suffit de mettre à jour ou de modifier l'ontologie.

Réconciliation de données avec un schéma. Nous avons formulé le problème de réconciliation de données avec un schéma sous forme d'un problème d'enrichissement sémantique de données structurées représentant des tableaux (ensemble de lignes et de colonnes) collectés et extraits à partir du Web. Nous nous sommes focalisés sur les tableaux car, dans le cadre applicatif concernant la veille scientifique du risque microbiologique dans les aliments que nous présentons plus en détail en Section 1.2, les experts du domaine ont constaté que les tableaux sont une représentation synthétique des données et que ces dernières sont souvent pertinentes. Nous supposons que les tableaux extraits sont représentés dans une collection de documents XML dont des tags purement syntaxiques sont utilisés pour représenter la structure logique des données (lignes, colonnes). Ce format de document est nommé XTab pour *Xml Tables*.

Pour enrichir les données représentant les tableaux par la sémantique du domaine contenue dans l'ontologie, nous exploitons la structure logique des données. Nous mettons également en correspondance les données avec le contenu de l'ontologie en nous appuyant sur des mesures de similarité syntaxique entre les données des tableaux d'une part et les concepts, les termes et les relations du domaine représentés dans l'ontologie d'autre part. Le résultat final de l'enrichissement sémantique, est représenté dans des documents semi-structurés représentés en XML, pour lesquels nous avons spécifié la nature et l'organisation des éléments, en utilisant une Définition de Type de Documents (DTD) que nous avons nommée SML (*Semantic Markup Language*). Dans un document SML, la plupart des tags et des valeurs sont des concepts, des termes ou des relations contenus dans l'ontologie.

Les principales contributions concernant le problème de réconciliation de données avec un schéma sont résumées dans les points suivants :

- la proposition d'une méthode générique et automatique d'enrichissement sémantique d'informations structurées représentant des tableaux ;
- la définition d'une DTD pour représenter de manière flexible le résultat de l'enrichissement sémantique d'informations représentant des tableaux ;
- la découverte de relations candidates à l'alimentation d'un entrepôt de données ou candidates à l'enrichissement d'une ontologie ;
- le développement d'un prototype, publié sous forme d'un service Web nommé XTab2SML, mettant en oeuvre cette méthode d'enrichissement sémantique.

Ces résultats concernant l'enrichissement sémantique d'informations structurées ont été publiés dans [SGHP05, GHPS05a, GHPS05b, GHPS06, GHM+05]

Réconciliation de références dont les descriptions sont conformes à un même schéma. Nous proposons une approche de réconciliation de références fondée sur la sémantique du schéma et des données. Cette sémantique déclarée explicitement au niveau du schéma est, d'une part, traduite sous forme d'un ensemble de règles d'inférence permettant un calcul symbolique (Vrai | Faux) des réconciliations et des non réconciliations entre références. D'autre part, elle est exploitée par une mesure de similarité *informée* qui capture les connaissances sur le schéma dans un calcul numérique de similarité des références exprimé dans un système d'équations non linéaire. L'approche de réconciliation de références que nous proposons combine une méthode Logique pour la Réconciliation de Références que nous appelons L2R et une méthode Numérique de Réconciliation de Références (valeurs réelles de similarité, dans [0..1]) que nous appelons N2R. La méthode logique permet d'inférer des décisions certaines de réconciliation et de non réconciliation, ainsi que des synonymies et des non synonymies entre valeurs, grâce aux règles logiques traduisant la sémantique des connaissances déclarées explicitement dans l'ontologie. La méthode numérique permet de calculer un score de similarité pour les paires de références, par la résolution itérative d'un système d'équations, pour des références pour lesquelles la méthode logique n'a pas permis d'inférer leur réconciliation ou leur non réconciliation. Dans les deux méthodes, les décisions de (non) réconciliation ou les modifications des scores de similarité sont propagées à d'autres paires de références, ce qui fait que les deux méthodes peuvent être qualifiées de globales.

En résumé, les principales contributions concernant le problème de réconciliation de références conformes à un même schéma sont :

- la proposition d'une méthode logique de réconciliation de références (L2R) exploitant les connaissances du domaine ;
- la proposition d'une méthode numérique de réconciliation de références (N2R) exploitant les connaissances du domaine ;
- La preuve de convergence du calcul itératif de la similarité par la méthode numérique ;
- l'implémentation d'un prototype (LN2R) mettant en oeuvre la méthode logique et numérique de réconciliation de références.

Ces résultats concernant la réconciliation de références ont été publiés dans [SPR07b, SPR07a, SPR07c, PS07].

5 Plan du manuscrit

Ce manuscrit est organisé en deux parties. La une première partie traite de la réconciliation de données avec un schéma et la seconde partie concerne la réconciliation de références conformes à un même schéma.

Partie I : Réconciliation de données avec un schéma

Chapitre 1 : Présentation du contexte. Ce chapitre présente tout d'abord un survol des travaux les plus représentatifs pour le problème de réconciliation de schémas exploitant les données. Ensuite, nous présentons le cadre applicatif de ce travail qui s'est inscrit dans le cadre du projet eDot (entrepôt de Données ouvert sur la toile).

Chapitre 2 : Enrichissement sémantique d'informations structurées. Ce chapitre est dédié à la présentation de la méthode d'enrichissement sémantique d'informations structurées représentant des tableaux. Cette méthode est automatique, flexible et guidée par une ontologie. Nous présentons également l'utilisation du résultat pour découvrir des relations candidates à l'enrichissement d'un entrepôt et des relations candidates à l'enrichissement d'une ontologie.

Chapitre 3 : Développement et expérimentations. Dans ce chapitre, nous présentons le prototype XTab2SML qui met en oeuvre la méthode d'enrichissement sémantique. Nous présentons également les résultats d'expérimentation dans le domaine du risque alimentaire.

Partie II : Réconciliation de références conformes à un même schéma

Chapitre 1 : État de l'art. Ce chapitre passe en revue les méthodes de réconciliation de références abordées dans la littérature scientifique. Ces techniques sont classées selon deux critères : l'exploitation des relations entre données et l'exploitation des connaissances du domaine. Nous présentons également les mesures de similarité les plus connues et des méthodes permettant de réduire le nombre de comparaisons de références.

Chapitre 2 : Modèle de données RDFS+. Ce chapitre est dédié à la présentation de notre modèle de données RDFS+ choisi pour décrire les données après une brève présentation des langages de représentation d'ontologie sur lesquels notre modèle RDFS+ est fondé.

Chapitre 3 : Méthode Logique de Réconciliation de Références (L2R). Ce chapitre présente la méthode logique de réconciliation de références, qui exploite la sémantique du schéma et des données en la traduisant sous forme d'un ensemble de règles d'inférence. Ces règles permettent d'inférer des réconciliations et des non réconciliations certaines entre références.

Chapitre 4 : Méthode Numérique de Réconciliation de Références (N2R). Ce chapitre présente la méthode numérique de réconciliation de références qui calcule des scores de similarité pour les références par la résolution itérative d'un système d'équations non linéaire. Ce calcul numérique de la similarité capture la sémantique du schéma par une fonction particulière d'agrégation des scores de similarité.

Première partie

Réconciliation de données avec un schéma

Introduction

Le développement et la popularité du Web ont beaucoup contribué à l'expansion de la quantité d'informations disponibles. Par conséquent, les méthodes traditionnelles de recherche d'informations, telles que la navigation et la recherche d'informations par mots clés, sont devenues de plus en plus inappropriées pour la recherche d'informations précises sur le Web et inadaptées aux traitements automatiques de ce gisement d'informations.

Plusieurs organismes de recherche et de nombreux industriels souhaitent aujourd'hui exploiter ces données disponibles. Ils s'intéressent à ces informations pour compléter, par exemple, des bases de données déjà existantes et de rendre les réponses aux requêtes des utilisateurs les plus complètes et les plus précises possible.

Pour réaliser cet objectif, il est possible de s'appuyer sur les applications permettant l'interrogation de données stockées dans des sources hétérogènes et distribuées (e.g. collections de pages Web, données semi-structurées, données relationnelles, etc). Ces diverses sources peuvent être exploitées dans des systèmes d'intégration de données tels que les médiateurs ou les entrepôts de données.

L'approche *entrepôts de données* est au centre du projet **e.dot** *(Entrepôt de Données Ouvert sur la Toile)* dont l'objectif est le développement de technologies, permettant de construire et de maintenir des entrepôts thématiques de qualité le plus automatiquement possible. Dans le projet e.dot, le domaine d'application choisi est le risque microbiologique dans les aliments, domaine qui présente des enjeux majeurs. En effet, les lois sur la sécurité alimentaire étant de plus en plus strictes, la quantité d'analyses effectuées tous les jours par l'industrie alimentaire entraîne des dépenses importantes qui peuvent être limitées si des outils sont développés pour mieux connaître le comportement des microorganismes dans les aliments. Il sera alors possible de prévenir le risque de contamination au lieu de constater les épisodes de crises. C'est dans le contexte de ce projet que s'inscrit le travail que nous présentons dans cette partie.

Pour pouvoir construire ou alimenter un entrepôt thématique par des données concernant le domaine d'application et pour pouvoir offrir aux utilisateurs une vue uniforme un schéma et un vocabulaire communs doivent être utilisés pour décrire et stocker les données. Ceci est possible grâce à l'utilisation d'une ontologie du domaine d'application comme un vocabulaire commun pour la description des données mais également comme interface de requêtes. Nous donnons une présentation synthétique du projet e.dot au chapitre 1.

Indépendamment du choix de l'architecture (entrepôt ou médiateur) du système d'intégration, les problèmes d'hétérogénéité des schémas et des données doivent être résolus. Dans cette partie de la thèse, nous nous focalisons sur le problème d'hétérogénéité de schémas, problème que nous posons entre des données collectées sur le Web ou fournies par des fournisseurs de contenus et l'ontologie du domaine. Les infomrmations que nous considérons représentent des *tableaux* extraits de documents HTML, PDF ou Excel collectés ou fournis. Le choix de se focaliser sur la collecte et le traitement de documents contenant des tableaux est justifié pour deux raisons.

Le première est d'ordre sémantique : d'après les experts du domaine d'application, les tableaux présents dans les documents du Web contiennent généralement des informations synthétiques et fiables. La seconde raison est d'ordre opérationnel : les tableaux sont une représentation structurée des informations, ce qui rend leur extraction et leur interprétation plus facile par rapport aux textes bruts. Les tableaux extraits sont ensuite représentés dans un format uniforme qui est le format XTab pour *XML Tables*. Il s'agit d'une représentation en XML permettant de décrire des structures génériques de tableaux indépendantes du domaine et du format du document initial. Dans un document XTab, un tableau est représenté par son titre, les titres de ses colonnes et par un ensemble de lignes, elles-mêmes composées d'une ensembles de valeurs organisées en cellules.

Pour pouvoir intégrer les données représentées dans les tableaux puis les interroger, il est important de les représenter de la manière la plus homogène possible. Pour ce faire, nous réconcilions ces données avec le contenu de l'ontologie du domaine. Par analogie avec les données relationnelles, un tableau peut être vu comme une relation ayant comme attributs les titres des colonnes et comme instances les valeurs contenues dans les cellules.

Pour le cas particulier de nos données présentées dans des tableaux nous formulons le problème de *réconciliation de données avec une ontologie* comme un problème d'*enrichissement sémantique* où nous cherchons à déterminer quel(s) élément(s) de l'ontologie est représenté par quel(s) élément(s) du tableau.

L'ontologie représente le vocabulaire du domaine d'application, sous la forme d'une taxonomie de termes et de concepts, ainsi que les relations sémantiques entre les différents concepts du domaine. Ainsi, pour enrichir les données avec la sémantique du domaine déclarée dans l'ontologie, nous proposons une méthode qui permet de mettre en correspondance le contenu du tableau avec l'ontologie en exploitant la structure logique des données. En effet, nous exploitons l'organisation des données en colonnes pour identifier des concepts et leur organisation en lignes pour identifier des relations sémantiques.

Nous disposons d'une ontologie Sym'Previus sur le risque alimentaire. Nous montrons en Figure 3, un extrait de la taxonomie de concepts et de termes représentés dans cette ontologie Sym'Previus. En Figure 4, nous montrons deux exemples de tableaux contenant des données traitant du risque alimentaire et que nous souhaitons enrichir par le contenu de l'ontologie du domaine.

Dans un contexte où les données sont hétérogènes, parfois erronées, et où les informations sont incomplètes, une méthode d'enrichissement sémantique de données des tableaux par le contenu d'une ontologie est confrontée à plusieurs difficultés. À travers l'exemple des tableaux $T1$ et $T2$ de la Figure 4, nous allons illustrer les problèmes auxquels est confrontée une méthode d'enrichissement sémantique de données contenues dans des tableaux grâce à une ontologie :

- ▷ *Différentes conventions et différents vocabulaires.* Lorsque les données des tableaux sont produites par différentes personnes et pour des objectifs différents, les conventions et le vocabulaire utilisés pour représenter les données peuvent être différents. Par exemple, dans dans le tableau $T2$ de la Figure 4 la colonne représentant des microorganismes est nommée *Micro-Orga* alors qu'elle est nommée *Germ* dans le tableau $T1$. De plus, les unités de mesure de certains facteurs sont parfois différentes. Par exemple, les valeurs des colonnes *Temperature* dans les tableaux $T1$ et $T2$ ne sont pas données selon la même unité : dans le premier tableau il s'agit de températures données en degrés *Fahrenheit* tandis qu'elles sont données en degrés *Celsius* dans le second tableau. Il est clair qu'il est impossible de concevoir une ontologie qui contienne toutes les conventions et tous les vocabulaires utilisés

pour décrire les données d'un domaine.

▷ *Informations ambiguës.* Lors de la mise en correspondance des données des tableaux avec le contenu de l'ontologie, il est possible de trouver plusieurs mises en correspondance. Par exemple, pour la valeur "*Cheese : Cheddar*" du tableau $T1$ le terme *Cheese* et le terme *Cheddar* de l'ontologie de la Figure 3 sont tous deux des mises en correspondance possibles pouvant être obtenues par la mesure de leur similarité.

▷ *Informations incomplètes et erronées.* Lorsque les données des tableaux sont collectées à partir du Web et extraites automatiquement à partir de documents hétérogènes, certaines informations peuvent être non renseignées comme, par exemple, la valeur du pH de la troisième ligne du tableau $T2$. De plus, à cause d'une mauvaise segmentation des données ou à cause d'une mauvaise identification de la structure tabulaire (lignes, colonnes, titres), des informations peuvent être parfois erronées comme, par exemple, la valeur de la température de la deuxième ligne du tableau $T1$ où apparaît une information qui ne représente nullement une température mais plutôt un nom plus spécifique du microorganisme *Escherichia coli*.

▷ *Informations non identifiables.* Il arrive souvent que des informations contenues dans les tableaux ne soient pas suffisamment significatives ce qui rend leur identification difficile, comme, par exemple, le titre "*Proc. T*" de la quatrième colonne du tableau $T1$ qui ne peut pas être mis facilement en correspondance avec le contenu de l'ontologie.

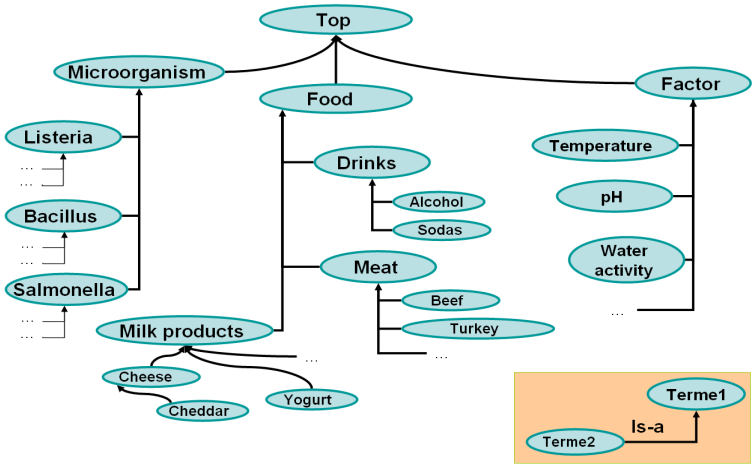

FIG. 3 – Extrait de la taxonomie de termes de l'ontologie du risque alimentaire Sym'Previus

Les éléments structurels, tels que les titres de colonnes, ne sont donc pas toujours significatifs des concepts de l'ontologie. En se fondant uniquement sur ces éléments structurels il est parfois impossible de trouver une mise en correspondance avec le contenu de l'ontologie. C'est pour cela que nous proposons une méthode d'enrichissement sémantique qui met en correspondance, en priorité, les valeurs représentées dans les cellules du tableau avec le contenu de l'ontologie pour identifier le concept que ces valeurs représentent. Dans le cas où les valeurs ne permettent pas d'identifier le concept, comme pour les valeurs numériques, les titres des colonnes sont alors

Exemple – Tableaux de données. Soient $T1$ et $T2$ deux tableaux qui contiennent les résultats d'expérimentation de la croissance des microorganismes dans les aliments.

T1 :

Article	**Germ**	**Temperature**	**Proc. T**
Cheese : cheddar	Listeria	26.6	87
Meat : beef	Escherichia coli	E.coli O157 :H7	90
Yogurt	Listeria monoctyogenes	44.6	86

T2 :

Food	**Micro-Orga**	**Temperature**	**pH**
Cheese	Listeria monoctyogenes	-0.4	4.3
Salad	Salmonella spp.	4.0	4.0
Crab	Bacillus cereus	4.0	

FIG. 4 – *Exemple de tableaux de données à intégrer*

comparés avec l'ontologie pour identifier un éventuel concept qui y soit représenté. En se fondant sur l'ensemble de concepts associés aux différentes colonnes du tableau, une deuxième étape permettant de mettre en correspondance cet ensemble de termes avec les relations sémantiques contenues dans l'ontologie est réalisée afin d'identifier celles qui sont représentées dans le tableau.

La méthode automatique d'enrichissement sémantique des données des tableaux avec l'ontologie peut parfois échouer dans l'identification des concepts ou des relations sémantiques qui sont représentés dans les tableaux, soit à cause de la façon dont les données sont représentées soit parce que l'ontologie ne représente que partiellement le domaine d'application. Dans ce cas, les informations non identifiées sont tout de même conservées et représentées par des éléments génériques (attributs ou relations) dans le résultat final de l'enrichissement sémantique. Ces informations imparfaitement identifiées pourront non seulement être exploitées lors de l'interrogation mais également pour découvrir des nouvelles relations qui seront proposées à l'expert du domaine pour enrichir l'ontologie.

Dans cette thèse nous proposons une méthode d'enrichissement sémantique de données contenues dans des tableaux avec l'ontologie du domaine, méthode complètement automatique, flexible et qui conserve toutes les mises en correspondance trouvées. Nous proposons également l'exploitation du résultat de la méthode d'enrichissement sémantique pour la découverte de relations sémantiques candidates à l'enrichissement d'un entrepôt thématique ou de l'ontologie du domaine.

Chapitre 1

Présentation du contexte

Sommaire

1.1 Méthodes de réconciliation de schémas exploitant les données . 9
1.1.1 Problème d'alignement et différentes mesures de similarité 10
1.1.2 iMAP : approche qui exploite les contraintes du domaine 12
1.1.3 Approche supervisée pour la découverte d'appariements entre schémas 14
1.1.4 FCA-Merge : approche fondée sur l'analyse formelle de concepts . . 16
1.1.5 Peuplement d'ontologies à partir de tableaux 17
1.1.6 Synthèse 19
1.2 Contexte applicatif : le projet e.dot 21
1.2.1 La base de données Sym'Previus 21
1.2.2 Chaîne de traitements des informations du Web 23
1.2.3 Tâches d'acquisition et de pré-traitement des documents du Web . 25

Dans ce chapitre nous présentons le contexte scientifique et applicatif de notre travail sur la réconciliation de données avec un schéma. En Section 1.1 nous donnons un bref état de l'art des méthodes les plus proches de notre approche de réconciliation. Il s'agit, en effet, des approches qui s'appuient sur les données ou les instances pour découvrir les appariements entre schémas. Puis, en Section 1.2, nous présentons le cadre applicatif de notre travail qui est le projet e.dot.

1.1 Méthodes de réconciliation de schémas exploitant les données

Dans cette section, nous présentons un ensemble d'approches permettant d'aligner des schémas ou des ontologies, c'est à dire de trouver des similarités ou des correspondances entre des éléments de deux schémas ou ontologies qui sont sémantiquement liées. De nombreux travaux se sont intéressés au problème de l'automatisation de cette opération d'alignement et nous pouvons citer Shvaiko et Euzenat [SE05], Rahm et Bernstein [RB01] ou encore Kalfoglou et Schorlemmer [KS05] qui permettent d'avoir une vue d'ensemble. Les correspondances trouvées permettent d'effectuer différentes tâches relatives à la manipulation de données hétérogènes exprimées dans des schémas ou des ontologies distincts. L'objectif peut être d'intégrer ces données, que ce soit grâce à un adaptateur dans le cadre d'un système d'intégration de type médiateur, ou que ce soit grâce à un extracteur dans le cadre de la construction d'un entrepôt de données. Ces correspondances peuvent également permettre d'effectuer des échanges entre des pairs dans des

applications Pair-à-Pair où les bases de données sont distribuées et autonomes ou encore de fusionner des ontologies.

Les mesures de similarité utilisées par les approches de réconciliation de schémas diffèrent en fonction des connaissances qu'elles peuvent prendre en compte. La plupart des travaux utilisent des connaissances que l'on peut qualifier de terminologiques. Il s'agit de connaissances textuelles apparaissant sur les étiquettes, descriptions ou commentaires associés à un élément comme un concept ou une relation d'une ontologie ou comme un attribut appartenant au schéma d'une base de données. Un grand nombre de travaux utilisent également des connaissances structurelles et sémantiques internes ou externes sur les éléments. La richesse de ces connaissances dépend fortement du modèle de données dans lequel l'ontologie ou le schéma est décrit. Il peut s'agir d'une ontologie représentée en OWL-DL ou d'une simple taxonomie. Une opération d'alignement peut également être appliquée sur des schémas tel qu'un schéma relationnel ou un schéma XML pour lesquels il n'existe pas de sémantique explicite qui soit associée aux éléments. De plus, la richesse de cette information structurelle et sémantique dépend également de l'expressivité effectivement exploitée par les experts chargés de la définition de l'ontologie.

Les approches diffèrent également sur les types de relations de mise en correspondance qu'elles peuvent trouver. En général, les approches ne cherchent que des relations d'équivalence entre des couples d'entités appartenant à des ontologies distinctes. Ces relations découvertes sont souvent associées à un degré de confiance. D'autres approches plus récentes essaient de découvrir d'autres types de relations telles que la subsomption, le recouvrement ou même la disjonction.

Notre étude porte sur la mise en correspondance d'un schéma global exprimé sous la forme d'une taxonomie de termes et d'un ensemble de relations sémantiques avec des données semi-structurées dans le cas particulier où ces données se présentent sous forme d'un tableau de données. Ce tableau peut être considéré comme une table relationnelle dont on peut découvrir le schéma afin de pouvoir l'exprimer dans le vocabulaire du schéma de référence. L'information dont nous disposons pour réaliser cette traduction est constituée du titre du tableau de données s'il existe, des titres de colonnes s'ils existent, et des données apparaissant dans ces colonnes. L'exploitation des données qui, dans le cas général, peut permettre de confirmer ou d'éliminer certains appariements incorrects est, dans les cas où l'information sur les ou l'un des schémas est faible, un atout considérable. Nous allons donc nous focaliser dans cet état de l'art sur les approches qui exploitent les données ou les instances pour aligner deux ontologies ou deux schémas.

Nous présentons tout d'abord brièvement le problème de l'alignement d'ontologie et les différents types d'approches d'alignement. Puis, nous nous focalisons sur les approches qui exploitent les instances pour trouver des correspondances entre les éléments des ontologies.

1.1.1 Problème d'alignement et différentes mesures de similarité

Soient deux ontologies o1 et o2, un alignement de o1 et o2 est constitué d'un ensemble d'appariements sachant qu'un appariement est défini par le tuple $< id, e1, e2, n, R >$ suivant [Euz04] :

- ⋆ id est un identifiant de l'appariement trouvé.
- ⋆ $e1$ et $e2$ sont des éléments des deux schémas ou ontologies o1 et o2 telles qu'une propriété, un concept, ou encore une relation.
- ⋆ n est un degré de confiance associé à un appariement trouvé entre e1 et e2

⋆ R est une relation (d'équivalence $\equiv$, de subsomption $\leq$, de disjonction $\perp$ ou de recouvrement $\sqcap$)

Les travaux d'alignement peuvent être catégorisés en fonction des différentes mesures de similarité utilisées. Les catégories que nous présentons dans la Figure 1.1 ci-dessous résument les différentes catégories présentées dans [RB01] [SE05] et [Bac06].

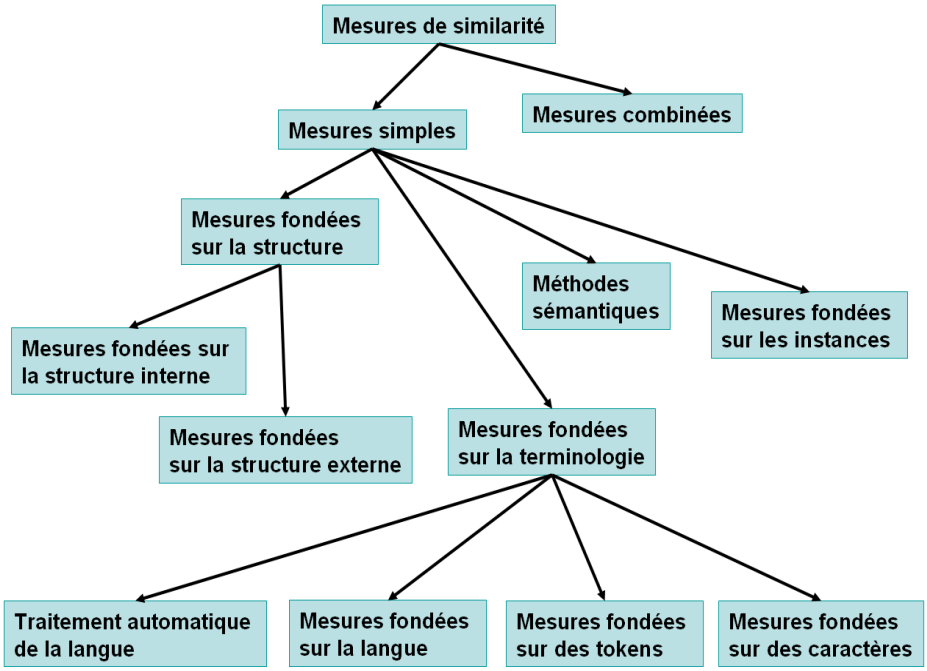

FIG. 1.1 – Une classification des mesures de similarité pour l'alignement

Les *mesures basées sur la structure interne* recherchent des appariements entre des éléments en analysant les éléments pris isolément et sans tenir compte de leurs relations avec d'autres éléments [RB01, SE05]. Ces mesures prennent en compte des informations telles que les attributs, les cardinalités, ou les types de données. Les *mesures basées sur la structure externe* recherchent des correspondances entre des éléments en analysant les relations entre éléments telles que les relations de subsomption ou les relations partie/tout, l'idée étant que si deux éléments sont similaires, leurs voisins peuvent également être similaires.

Les *méthodes terminologiques* se fondent sur la comparaison des chaînes de caractères, des termes ou bien des textes qui peuvent apparaître sur les noms des concepts ou des propriétés, ou sur toute étiquette, commentaire ou description apparaissant dans l'ontologie. Ces méthodes peuvent se diviser en deux catégories : les méthodes qui comparent des termes en se basant sur les caractères ou les tokens (cf. Section 4.2) et les méthodes qui utilisent des connaissances linguistiques telles que la lemmatisation ou encore l'élimination des mots vides ou qui utilisent des ressources linguistiques externes telles que des lexiques ou des thesaurus comme Wordnet [wor95].

Les *mesures basées sur les instances* ou *méthodes extensionnelles* exploitent l'extension des concepts ou des relations, et donc les ensembles d'instances. Si ces ensembles ont une partie commune, des mesures telles que la distance de Hamming ou la mesure de similarité de Jaccard [Jac01] peuvent être utilisées. La mesure de Hamming permet de compter le nombre d'éléments différents de deux ensembles. La mesure de Jaccard est le rapport entre l'intersection des ensembles et leur union. De plus, les valeurs apparaissant dans les descriptions des instances peuvent être comparées grâce à des techniques terminologiques ou grâce à des caractérisations fondées sur des contraintes telles que des domaines de valeur ou des patterns spécifiques (comme ceux que l'on peut définir pour les codes postaux ou les numéros de téléphone.

Enfin, les méthodes dites *sémantiques* ou *fondées sur un modèle* considèrent le problème d'alignement comme le calcul de la satisfiabilité d'une théorie exprimée en logique propositionnelle qui représente la fusion de deux ontologies et ce pour toutes les correspondances possibles. Dans le cas où les ontologies sont exprimées en logique de description, [SE05] envisagent la possibilité de réaliser un test de subsomption entre chaque paire de concepts.

1.1.2 iMAP : approche qui exploite les contraintes du domaine

iMAP [DLD+04] est une approche qui permet de découvrir semi-automatiquement des appariements entre des schémas relationnels en utilisant des instances décrites dans la base de données. Ces appariements sont éventuellement complexes c'est-à-dire qu'ils peuvent faire correspondre à un attribut d'un schéma une combinaison d'attributs de l'autre schéma. Il peut s'agir de concaténations telles que $address = concat(city, state)$ ou de fonctions telles que $room - price = room - rate * (1 + tax\text{-}rate)$.

Exemple (extrait de [DLD+04]) : soient deux schémas S et T de deux bases de données relationnelles concernant des maisons mises en vente. Supposons que ces schémas contiennent les tables suivantes :

Location	Price($)	Agent-id
Atlanta,GA	360,000	32
Raleigh, NC	430,000	15

TAB. 1.1 – *Schéma S : Houses*

Id	Name	City	State	Fee-rate
32	Mike brown	Athens	GA	0.03
15	Jean Laup	Raleigh	NC	0.04

TAB. 1.2 – *Schéma S : Agents*

Area	List-price	Agent-address	Agent-name
Denver,CO	550,000	Boulder, CO	Laura Smith
Atlanta, GA	370,800	Athens, GA	Mike Brown

TAB. 1.3 – *Schéma T : Listings*

L'objectif des auteurs est de retrouver, sur cet exemple, les correspondances suivantes : l'attribut *Location* du schéma S est équivalent à l'attribut *Area* du schéma T, l'attribut *Price*

multiplié par l'attribut 1 + *Fee-rate* de S est équivalent à l'attribut *List-price* du schéma T, l'attribut *Agent-address* est la concaténation des attributs *City* et *State*.

Les appariements complexes sont courants mais leur recherche est bien plus difficile que dans les cas des appariements 1-1. En effet, dans ce cas, le nombre d'appariements n'est pas borné par le produit du nombre d'attributs de chaque schéma. D'autre part il existe une infinité de fonctions permettant de combiner des attributs.

L'architecture d'iMAP est organisée de la façon suivante :

⋆ **Un générateur d'appariements** : il prend les deux schémas en entrée et renvoie pour chaque attribut d'un schéma l'ensemble des appariements candidats combinant des attributs de l'autre schéma. Ce générateur est composé d'un ensemble de modules de recherche spécialisés. Le système dispose par exemple d'un module de recherche numérique capable de produire des expressions arithmétiques utilisant les attributs du schéma source. Un module de recherche textuel est capable d'analyser les propriétés textuelles des attributs des deux schémas et envisage leur concaténation. Un autre module de recherche est spécialisé dans la conversion automatique de valeurs exprimées dans différentes unités de mesure. Ce module est capable, par exemple, de traduire des *livres* en *kilogrammes*. Le générateur dispose de huit modules de recherche spécialisés.

⋆ **Un estimateur de la similarité** : il permet d'évaluer chaque appariement candidat fourni par le générateur d'appariements. Cet estimateur utilise deux modules d'évaluation. L'un de ces modules permet d'évaluer une similarité syntaxique entre deux chaînes de caractères. L'autre module utilise un classifieur Bayesien. D'autres modules d'évaluation ont été envisagés mais ne sont pas encore implémentés.

⋆ **Un sélectionneur** : il choisit les meilleurs appariements candidats en vérifiant que certaines contraintes d'intégrité ne sont pas violées. Par exemple, un attribut représentant la ville de localisation d'une maison ne peut pas être mis en correspondance avec deux attributs d'un autre schéma. En effet, une maison n'a qu'une seule ville de localisation. Le sélectionneur choisit donc le meilleur ensemble d'appariements qui ne viole pas ce type de contrainte.

iMAP exploite également des connaissances du domaine permettant d'améliorer les résultats obtenus. Il s'agit de contraintes du domaine qui représentent une information telle que : « les lits et les noms de personnes n'ont pas de lien sémantique ». Ces attributs ne peuvent alors pas être combinés pour former un appariement. Les contraintes peuvent également concerner les valeurs d'un attribut donné. Il est ainsi possible de déclarer que la valeur moyenne du nombre de chambres des maisons ne peut excéder 10.

iMAP mémorise les appariements trouvés pour détecter des formes de combinaison arithmétique qui se répètent. À partir d'appariements tels que $price = pr * (1 + 0.06)$, le système est capable d'extraire le template VARIABLE∗(1+CONSTANT).

Ce système a été testé sur huit bases de données de quatre domaines distincts. Les auteurs ont pu montrer que la précision des appariements 1-1 allait de 68 % à 92% pour l'un des domaines mais qu'elle était nettement plus faible si des connaissances du domaine additionnelles n'étaient pas représentées et exploitées. Les résultats concernant les appariements complexes sont plus faibles (environ moitié moins) ce qui n'est pas étonnant compte tenu de la difficulté.

L'approche permet de fournir à l'utilisateur, qui examine et corrige les appariements trouvés, des explications concernant la présence ou l'absence d'un appariement mais également concer-

nant l'ordre des appariements trouvés. Ces explications indiquent les générateurs d'appariements utilisés, les valeurs des scores de similarité des différents évaluateurs et les contraintes d'intégrité exploitées par le sélectionneur.

Cette approche permet donc de générer des appariements éventuellement complexes entre deux schémas. Elle utilise différents modules de recherche d'appariements candidats qui peuvent varier en fonction du type de données de l'attribut et qui exploitent les données contenues dans les tables. Les meilleurs appariements sont expliqués et proposés à l'utilisateur qui doit les corriger. En revanche, cette approche ne permet pas de rechercher des appariements entre les relations sémantiques qui peuvent être exprimées dans les tables relationnelles. Les appariements générés ne concernent en effet que les attributs.

De plus, cette approche est une approche semi-automatique qui permet à un utilisateur *expert* de comprendre les décisions des différents modules d'iMAP pour corriger les appariements proposés. Il ne s'agit pas d'une approche dont l'objectif est d'être complètement automatique et donc de conserver des appariements éventuellement ambigus ou incomplets.

1.1.3 Approche supervisée pour la découverte d'appariements entre schémas

Dans [XE03], les auteurs proposent une approche de découverte d'appariements entre un schéma source et un schéma cible en s'appuyant sur des méthodes d'apprentissage supervisé. Dans un premier temps, la méthode cherche des appariements entre les deux schémas source et cible d'une part et les concepts d'une ontologie de domaine conçue manuellement par un expert d'autre part. Dans un deuxième temps en exploitant les relations de type *sorte-de* et *partie-de* entre les concepts de l'ontologie, la méthode cherche à identifier différents appariements. Une variété de méthodes d'alignement sont utilisées pour trouver des appariements potentiels pour lesquels des degrés de confiance dans [0..1] sont calculés. Pour trouver ces appariements potentiels, les auteurs exploitent non seulement les éléments du schéma mais également les données qui leur sont associées.

Les auteurs distinguent deux types d'éléments de schéma : les *éléments objets* qui représentent des entités complexes (e.g. Maison, Agent) et les *éléments valeurs* qui représentent des valeurs de base (e.g. Email, Fax). Si un lien existe entre un élément objet et un élément valeur il est alors appelé *attribut* et si un lien existe entre deux éléments objets il est alors appelé *relation*.

Soient S et T deux schémas à mettre en correspondance. Un appariement entre les éléments s et t peut être de deux types : (i) *direct*, si les éléments s et t représentent le même ensemble de valeurs ou d'objets. (ii) *indirect*, lorsqu'il n'existe pas d'appariement *direct* entre s et t et lorsque les données du schéma cible peuvent être dérivées à partir des données du schéma source grâce à des opérations de manipulation, par exemple, *adresse = Composition(rue, ville, code postal)*.

Pour trouver les appariements, différentes techniques de mise en correspondance sont utilisées :

- ⋆ **Relations terminologiques**. On considère la décomposition de chaque nom d'élément de schéma en un ensemble de tokens. Puis, le thesaurus Wordnet [wor95] est utilisé pour chercher des synonymies, des relations de subsomption et des hyperonymies. Un arbre de décision de type C4.5 [Qui93] est alors généré afin de classer les tokens appartenant à un ensemble d'exemples positifs et négatifs. Cette classification permet de calculer un degré de confiance $conf_1$ pour chaque appariement potentiel en mesurant, par exemple, le rapport entre le nombre d'exemples classés comme étant synonymes et le nombre d'exemples classés comme étant non synonymes.

⋆ **Comparaison des caractéristiques des valeurs**. Pour trouver des appariements potentiels entre les éléments des schémas, différentes caractéristiques sur les valeurs des attributs peuvent être comparées comme, par exemple, la longueur des chaînes de caractères, le ratio des caractères alphabétiques/numériques, la variance des valeurs numériques, etc. Un arbre de décision de type C4.5 est utilisé pour classer un ensemble d'exemples positifs et négatifs selon ses différentes caractéristiques, ce qui permet ensuite de calculer un degré de confiance $conf_2$.

⋆ **Comparaison des valeurs attendues pour les attributs**. Il s'agit d'un indice important pour l'appariement des éléments des schémas. On considère une ontologie simple représentant les concepts du domaine ainsi que les différentes relations entre ces concepts. À chaque concept est associé un ensemble d'expressions régulières permettant de mettre en correspondance des valeurs qui peuvent instancier le concept. Ensuite, en utilisant des techniques d'extraction, présentées dans [ECJ+99], on peut extraire des instances à partir des ensembles de données associés aux *éléments valeurs* des schémas source et cible. En exploitant les expressions régulières et les relations déclarées entre les concepts de l'ontologie on peut découvrir les appariements directs et indirects pour les éléments du schéma.
Considérons tout d'abord C_s (resp. C_t) l'ensemble de concepts qui se concatènent et qui s'apparient avec un élément s (resp. t) du schéma source (resp. cible). On calcule les ratios r_s et r_t qui représentent le pourcentage des valeurs des éléments s ou t qui s'apparient avec les valeurs associées aux concepts C_s ou C_t respectivement. Ensuite, au lieu de considérer un appariement *simple* entre les valeurs des éléments et celles des concepts on considère la mise en correspondance entre la concaténation de certaines valeurs des éléments avec celles des concepts et vise-versa. Sur ces appariements plus complexes les ratios r_s et r_t sont calculés en fonction du pourcentage des valeurs appariées en utilisant la concaténation.
Si $C_s = C_t$ alors l'appariement entre les éléments s et t est direct, sinon il s'agit d'un appariement indirect dont l'opération est définie selon les cas suivants :

- Si $C_s \supset C_t$ ($C_s \subset C_t$) alors l'opération est une *Décomposition* (*Composition*).
- S'il existe un concept $c_s \in C_s$ et un concept $c_t \in C_t$ tel que les valeurs de c_s (resp. c_t) sont déclarées comme un sous ensemble de valeurs de c_t (resp. c_s) alors l'opération est une *Union* (resp. *Sélection*).

Pour l'ensemble de ces appariements potentiels, un degré de confiance $conf_3$ est calculé comme suit : (i) si les valeurs des ratios r_s et r_t sont supérieures à un seuil donné alors $conf_3(s,t) = 1$, sinon (ii) on construit deux vecteurs v_s et v_t dont les composantes représentent les ratios r_s et r_t calculés pour tous les concepts $c_s \in C_s$ et $c_t \in C_t$, puis on calcule $conf_3$ en utilisant la mesure *cosinus* [Aiz03] comme suit :
$conf_3(s,t) = cos(v_s, v_t) \times (r_s + r_t)/2$

Une fois que toutes ces comparaisons ont été effectuées et donc que des appariements directs et indirects potentiels ont été trouvés, un degré de confiance est calculé pour chacun d'entre eux par la combinaison des différents degrés de confiance $conf_1$, $conf_2$ et $conf_3$ calculés aux étapes précédentes. Puis, relativement à un seuil donné th_{conf} on détermine les appariements les plus pertinents.

Cette approche permet donc de générer des appariements éventuellement complexes entre deux schémas. Elle s'appuie sur différentes techniques d'alignement : techniques terminologiques,

techniques exploitant les caractéristiques des données et techniques de comparaison des valeurs des attributs. Toutefois, cette approche ainsi que iMAP [DLD+04] ne permet pas de trouver des appariements pour les relations sémantiques qui peuvent être représentées entre les éléments des schémas. De plus, il s'agit donc d'une approche supervisée ce qui la rend inadaptée à un contexte où toutes les tâches sont complètement automatiques comme, par exemple, dans le cadre du projet e.dot (cf. Section 1.2).

1.1.4 FCA-Merge : approche fondée sur l'analyse formelle de concepts

FCA-Merge [SM01] est un outil qui permet de fusionner deux ontologies en exploitant les instances de leurs concepts. La méthode se fonde sur la construction d'un treillis de Galois qui représente le résultat de la fusion des deux ontologies. Ce treillis devra ensuite être corrigé par un utilisateur expert.

Nous allons tout d'abord rappeler brièvement ce qu'est un *treillis de Galois*. L'utilisation de treillis de Galois pour l'analyse formelle de concepts (*Formal Concept Analysis*) est une approche théorique de regroupement conceptuel des données permettant d'identifier des concepts. Elle a été initiée par R. Wille au début des années 80 [Wil82]. Le treillis de Galois ou de concepts est constitué de couples d'ensembles fermés d'instances et d'attributs. Plus précisément, le contexte formel d'un treillis de Galois est défini par un triplet (G, M, I) où G est un ensemble d'instances, M un ensemble d'attributs, et I une relation binaire telle que $I \subseteq G \times M$. Le couple (g, m) signifie que l'instance g possède l'attribut m.
Soit $A \subseteq G$, l'intension de A est l'ensemble A' tel que $A' = \{m \in M \mid \forall g \in A, (g, m) \in I\}$
Soit $B \subseteq M$, l'extension de B est l'ensemble B' tel que $B' = \{g \in G \mid \forall m \in B, (g, m) \in I\}$

Un concept formel de (G, M, I) est une paire (A, B) de $G \times M$ telle que A'=B et B'=A. Les concepts sont organisés en hiérarchie grâce à une relation de subsomption $\leq$:

$$(A1, B1) \leq (A2, B2) \; ssi \;\; A1 \subseteq A2 \;\; (ou \;\; B1 \supseteq B2)$$

Dans le cadre de FCA-Merge, G est un ensemble de documents et M est l'ensemble des concepts de l'ontologie. Pour chaque ontologie O_i, une étape d'analyse linguistique des documents de G permet de générer un contexte formel (G, M_i, I_i) tel que l'on a $(g, m) \in I_i$ si le document g contient des instances du concept m.

Soit G un ensemble de cinq documents et soient deux ontologies $O1$ et $O2$ qui contiennent respectivement un ensemble de concepts $M1$ et $M2$. L'étape d'analyse linguistique va permettre, par exemple, d'aboutir aux relations $I1$ et $I2$ des deux contextes formels $(G, M1, I1)$ et $(G, M2, I2)$ décrits en table 1.4 et table 1.5.

I1	**vacation**	**hotel**	**event**	**concert**	**root**
doc1	1	1	1	1	1
doc2	1	1	1	1	1
doc3	1	1	1	0	1
doc4	0	0	1	1	1
doc5	0	1	1	1	1

TAB. 1.4 – *Le contexte formel (G,M1,I1)*

Ensuite, un treillis de concepts est construit à partir de l'union de ces deux contextes. Seuls les concepts formels équivalents ou généralisant au moins l'un des concepts des ontologies originales

I1	hotel	accommodation	musical	root
doc1	1	1	1	1
doc2	1	1	0	1
doc3	1	1	1	1
doc4	0	0	1	1
doc5	1	1	1	1

TAB. 1.5 – *Le contexte formel (G,M2,I2)*

seront présentés à l'utilisateur. Cela évite, en effet, de proposer un grand nombre de concepts très spécialisés. Dans l'exemple, le concept d'hôtel de $O1$ génère le même concept formel que le concept d'hôtel de $O2$. Aussi il est suggéré de les fusionner. En revanche, des concepts formels, qui ne sont générés que par un seul concept d'une ontologie, sont inclus dans l'ontologie résultat sans validation, le concept étant considéré comme distinct des autres concepts. Dans les cas où les concepts formels sont constitués de plusieurs concepts, l'utilisateur doit décider si un nouveau concept doit être créé ou si le concept formel représente une relation. Ainsi, pour un concept formel dont l'intention est {*Hotel*, *Event*}, l'utilisateur doit choisir de créer un concept *HotelEvent* ou une nouvelle relation *OrganizesEvent* de signature {*Hotel*, *Event*}.

Cette approche semi-automatique permet donc de découvrir des appariements entre des concepts de deux ontologies en se fondant sur leurs instances communes. Elle permet également de découvrir de nouveaux concepts ou de nouvelles relations. Dans cette approche, une instance est un document que l'on peut identifier par exemple par une URL. Il est donc possible dans une telle approche de former des ensembles d'instances communes à deux concepts de deux ontologies distinctes. Le problème que nous traitons ne permet pas d'identifier les instances communes aussi facilement puisqu'il s'agit de données contenues dans les tableaux des documents.

1.1.5 Peuplement d'ontologies à partir de tableaux

La méthode de peuplement d'ontologies proposée par Pivk et al. [PCS04] est une méthode qui permet de décrire les tableaux en termes de leur syntaxe et de leur sémantique. L'originalité de cette approche réside dans la génération de *Frames Logiques* à partir de tableaux contenus dans des documents HTML. L'utilisation des Frames Logiques permet de décrire la sémantique des tableaux dans un modèle théorique en offrant une description explicite de la signification du contenu des cellules et de la dimension fonctionnelle des cellules des tableaux.

La méthode est fondée sur un modèle cognitif [Hur01] de tableaux qui est composé des cinq dimensions : (i) *Graphique*, elle représente la description de l'image du tableau au niveau des pixels des traits et du texte ; (ii) *Physique*, elle représente la description des cellules relativement à la position dans le tableau ; (iii) *Structurelle*, elle représente l'organisation des cellules et des relations de navigation dans le tableau qu'elle soient verticales ou horizontales ; (iv) *Fonctionnelle*, elle correspond à l'ensemble des cellules permettant d'accéder aux cellules contenant les données ; et (v) *Sémantique*, elle représente la description de la sémantique du tableau et des relations entre les différentes interprétations du contenu des cellules. Dans le travail de Pivk et al. la dimension graphique n'est pas considérée car on ne traite pas les tableaux représentés dans des images. Ce modèle est instancié suivant différentes étapes.

Les tableaux représenté en HTML sont, dans un premier temps, soumis à une phase de nettoyage et de normalisation où l'on corrige la structure des tableaux en supprimant tous les

Tour Code			DP9LAX01AB	
Valid			01.05.-30.09.04	
Class/Extension			Economic	Extended
Adult	P	Single Room	35,450	2,510
	R	Double Room	32,500	1,430
		Extra Bed	30,550	720
Child	I	Occupation	25,800	1,430
	C	No Occupation	23,550	720
	E	Extra Bed	22,900	360

TAB. 1.6 – *Exemple de table*

groupements de cellules.

Dans un deuxième temps, on cherche à assigner à chaque cellule du tableau un type fonctionnel qui peut être soit un type A-Cell pour une cellule représentant un attribut, soit un type I-Cell pour une cellule représentant des instances. Une heuristique est utilisée pour assigner ces types aux différentes cellules : les cellules se situant en bas à gauche sont de type I-Cell et celles qui se situent en haut à droite sont de type A-Cell. Pour ces cellules une probabilité de 1 leur est assignée. En ce qui concerne le reste des cellules, la probabilité d'être de type I-Cell est obtenue en calculant la proportion des tokens composant la valeur d'une cellule qui sont du type[1] DATE, NUMBER ou CURRENCY. La probabilité d'être de type A-Cell est obtenue par le complémentaire de la probabilité d'être de type I-Cell. En s'appuyant sur la distance des types associés aux tokens des valeurs, dans la hiérarchie de types, il est possible de mesurer la similarité entre les cellules. Pour un tableau avec r lignes et s colonnes, la distance Δ_{cols} entre les s colonnes du tableau est calculée par la somme des distances entre la dernière colonne et les $s-1$ colonnes précédentes. Par analogie, la distance Δ_{rows} entre les r lignes du tableau est calculée. L'orientation du tableau est obtenue selon les trois cas suivants :

- Si $\Delta_{cols} > \Delta_{rows}$ alors l'orientation est verticale ;
- Si $\Delta_{cols} < \Delta_{rows}$ alors l'orientation est horizontale ;
- Si $\Delta_{cols} = \Delta_{rows}$ alors l'orientation est verticale.

Dans un troisième temps, le tableau est soumis à une phase de découverte des *unités logiques* et des *régions*. Une *unité logique* est produite par une séparation horizontale dans le cas d'une orientation verticale et par une séparation verticale dans le cas d'une orientation horizontale du tableau. Par exemple, le tableau 1.6 contient trois unités logiques : la première correspond aux deux premières lignes, la deuxième correspond à la troisième ligne et la troisième correspond au reste du tableau. Une *région* dans le tableau correspond à un rectangle appartenant à une seule unité logique et contenant des cellules ayant le même type fonctionnel (i.e. I-Cell ou A-Cell).

Dans un quatrième temps, les auteurs proposent de construire ce qu'ils nomment le *modèle fonctionnel du tableau* (FTM) qui représente une hiérarchisation des régions découvertes à l'étape précédente. Il s'agit d'organiser les régions de façon à mettre celles qui contiennent des cellules de types I-Cell au niveau des feuilles de l'arbre et de mettre les autres régions à des niveaux plus

[1] Une hiérarchie de types est donnée dans [PCS04] et elle permet d'organiser les différents types qui peuvent être associés aux tokens composant les valeurs des cellules, par exemple, le type ALPHA-NUM est plus général que les types ALPHA et NUMBER

proches de la racine de l'arbre. Puis, à chaque nœud du FTM (ie. une région) on cherche à associer une étiquette sémantique obtenue soit par la recherche d'un hyperonyme approprié correspondant à un généralisant dans WordNet [wor95] soit par l'interrogation du service GoogleSets [goo02] afin d'obtenir un synonyme.

La dernière étape de cette méthode consiste à transformer le FTM enrichi sémantiquement en un *Frame Logique* qui permet de représenter un tableau sous la forme d'un ensemble de méthodes. Un frame logique est défini par un nom et par l'ensemble de méthodes permettant d'obtenir les valeurs des cellules. Par exemple, dans le tableau 1.6, nous pouvons avoir la méthode : $Price(PersonType, RoomType) \Rightarrow$ NUMBER dont le nom est *Price*, le domaine est NUMBER et les paramètres sont *PersonType* et *RoomType*. Afin de générer le Frame Logique, on crée une méthode m pour chaque région R correspondant à des cellules de type I-Cell. Puis, on considère toutes les régions correspondant à des cellules de type A-Cell et se situant dans les nœuds antécédents de la région R comme des paramètres de la méthode m.

Les noms des méthodes sont obtenus par la combinaison des nom des A-Cell et des étiquettes sémantiques associées aux régions représentées par des nœuds parents du FTM. Les domaines de valeurs des méthodes correspondent aux types des tokens des cellules représentées par la région.

Le *Frame Logique* idéal représentant le tableau 1.6 est le suivant :

```
Tour[Code => ALPHANUMERIC
    DateValid => DATE
    EconomicExtension(PersonClass, RoomClass) => LARGE_NUMBER
    ExtendedExtension(PersonClass, RoomClass) => LARGE_NUMBER]
```

L'approche proposée par Pivk et al. [PCS04] s'appuie sur des ressources linguistiques externes pour l'enrichissement sémantique d'une partie du contenu du tableau. L'approche d'enrichissement sémantique que nous proposons est différente. En effet, comme nous allons le voir dans le chapitre 2, notre approche est fondée sur une ontologie de domaine et n'utilise pas de ressources externes. De plus, les relations sémantiques que nous cherchons à identifier sont pré-définies dans l'ontologie. Contrairement à [PCS04], dans notre approche nous ne occupons pas de la tâche de reconnaissance et de normalisation de la structure du tableau, ce qui permet ainsi de séparer cette étape de l'étape d'enrichissement sémantique.

1.1.6 Synthèse

Dans cette section nous avons présenté quelques approches de réconciliation de schémas qui exploitent non seulement les éléments des schémas mais également les données. Ce type d'approche consiste à se servir des appariements entre les données pour soit renforcer le degré de confiance dans les appariements potentiels, soit trouver de nouveaux appariements. L'exploitation des données offre une alternative intéressante lorsque les informations représentées par les éléments du schéma sont limitées ou encore absentes, ce qui est souvent le cas des données semi-structurées et des données extraites automatiquement à partir du Web.

Différentes techniques sont utilisées pour trouver les appariements entre schémas. L'approche iMAP s'appuie sur des mesures de similarité syntaxiques et sur un ensemble de contraintes du domaine d'application. La méthode proposée dans [PCS04] s'appuie sur un ensemble d'heuristiques, pour reconnaître la structure logique des données, et sur l'utilisation de ressources linguistiques externes telles que Wordnet pour l'enrichissement sémantique des données. Dans [XE03] la méthode utilise l'apprentissage supervisé appliqué sur des arbres de décision ainsi que la mesure

de similarité *cosinus* afin de trouver les différents appariements. Dans FCA-Merge les appariements sont obtenus grâce à l'analyse formelle des concepts et en s'appuyant sur l'intersection des ensembles d'instances des concepts. Le choix de ces différentes techniques est induit d'une part par la nature des entrées traitées et du format des sorties souhaité, d'autre part par les caractéristiques du contexte applicatif dans lequel elles sont utilisées.

Les entrées des méthodes proposées différent : dans iMAP [DLD+04] et dans [XE03] les entrées sont des schémas relationnels instanciés, dans FCA-Merge [SM01] un ensemble de documents et un ensemble de concepts constituent les entrées et dans la méthode proposée dans [PCS04] l'entrée est un tableau représenté en HTML. En effet, les méthodes de réconciliation de schémas exploitant les instances peuvent être adaptées à certains types de schémas mais pas à d'autres, car la différence des caractéristiques des entrées exploitées influent sur l'applicabilité de ces méthodes. Il peut s'agir par exemple d'une différence dans le niveau de structuration des schémas (e.g. tableaux vs. documents XML), de la pertinence des éléments du schéma (e.g. titre de colonne d'un tableau vs. attribut dans un schéma relationnel), de la proportion des données représentées dans les schémas ainsi que de la qualité des données (e.g. redondances, erreurs).

La proposition d'une méthode de réconciliation de schémas dépend aussi des caractéristiques des sorties souhaitées. Les méthodes proposées dans [DLD+04] et [XE03] permettent de fournir à la fois des appariements simples $1:1$ mais également des appariements plus complexes $n:m$. Dans la méthode proposée dans [PCS04] la sortie est différente, il s'agit d'un *Frame Logique* représentant la sémantique du contenu du tableau traité. Les caractéristiques des sorties d'une méthode de réconciliation jouent un rôle dans le choix des techniques utilisées pour résoudre le problème mais aussi dans la complexité du problème lui-même (e.g. appariements $1:1$ vs appariements $n:m$).

Dans le cadre de cette thèse nous avons proposé une approche de réconciliation de schémas fondée sur les données et guidée uniquement par une ontologie de domaine. Cette méthode permet de trouver des appariements entre le contenu d'un tableau donné en XML et le contenu d'une ontologie. Les appariements que notre méthode permet de trouver sont de deux types : (i) entre une colonne du tableau et un concept de l'ontologie et (ii) entre un sous ensemble de colonnes du tableau et un ensemble de concepts liés par une relation sémantique. Ces appariements peuvent être découverts automatiquement, entre autres, grâce aux appariements plus fins que nous cherchons entre les valeurs des cellules du tableau et les termes de l'ontologie. Avoir comme entrée des tableaux, comme dans [PCS04], nous oblige à interpréter d'abord le schéma du tableau avant de procéder à l'appariement. De plus, à la différence de [SM01] nous ne disposons pas d'appariements entre données, cependant nous utilisons des mesures de similarité syntaxiques pour les calculer.

Nous pouvons caractériser notre approche de réconciliation comme **non supervisée**, car contrairement à [XE03] aucun échantillon de données étiquetées n'est utilisé, comme approche guidée uniquement par une **ontologie**, car à l'inverse de [PCS04, XE03] aucune ressource linguistique externe (e.g. Wordnet) n'est exploitée et enfin comme une approche complètement **automatique** car contrairement à [DLD+04, SM01] aucune validation manuelle n'est utilisée.

1.2 Contexte applicatif : le projet e.dot

Notre travail concernant l'enrichissement sémantique de données structurées s'inscrit dans le cadre du projet e.dot (*Entrepôts de Données Ouverts sur la Toile*). Le projet e.dot [e.d04] a été soutenu par le Ministère de l'Industrie, et labellisé RNTL en mai 2002. Il a réuni trois équipes de recherche IASI/LRI, INRIA-Futurs, MIA/INRA et la start-up Xyleme. Son objectif est de développer des entrepôts de données dans des domaines spécifiques en intégrant de manière automatique des informations découvertes sur le Web avec des données privées obtenues de fournisseurs de contenus. Il s'agit de permettre une véritable exploitation de l'immense gisement d'informations qu'est le Web. Le projet est fondé sur une très grande utilisation de XML et des nouveaux standards développés autour de ce modèle, *XQuery* pour l'interrogation, *XML Schéma* et les *DTD*s pour la définition de types ainsi que *SOAP/WSDL* pour la définition de services Web.

Pour la spécification du système, la validation des approches et des développements, le projet *e.dot* s'est appuyé sur la construction et l'exploitation d'un entrepôt de données spécialisé dans le domaine du risque alimentaire dont l'ontologie a été conçue dans le cadre du projet Sym'Previus [Sym99]. Ce domaine particulier a été choisi car il s'agit d'un bon exemple pour montrer les limites des constructions manuelles d'entrepôts, limites dues à :

- ⋆ la dispersion de l'information sur une multitude de serveurs (nombreux laboratoires) ;
- ⋆ l'évolution rapide des données déjà répertoriées ;
- ⋆ la nécessité de découvrir rapidement des informations nouvelles apparaissant sur le Web.

De nombreux autres domaines, ayant des caractéristiques semblables, auraient pu être choisis comme domaine d'application : les dépêches d'agences de presse, les publications de bilans d'entreprise, la biologie, l'aéronautique, etc.

Le projet e.dot est décomposé en quatre lots :
- ⋆ Lot 1 : Spécification d'un entrepôt de données pour le risque alimentaire.
- ⋆ Lot 2 : Acquisition des données pertinentes du Web.
- ⋆ Lot 3 : Organisation et structuration de l'entrepôt.
- ⋆ Lot 4 : Validation auprès des utilisateurs.

La chaîne de traitement des documents dans le projet e.dot est implémentée sous la forme d'un ensemble de services Web qui s'enchaînent : la sortie d'un service correspond à l'entrée d'un autre.

Notre travail concernant l'enrichissement sémantique des informations structurées fait partie du lot 3 du projet. Nous traitons les documents XML fournis par les services d'acquisition du lot 2.

1.2.1 La base de données Sym'Previus

Le projet e.dot s'est appuyé sur les résultats du projet Sym'Previus [Sym99] soutenu par le Ministère de la Recherche et le Ministère de l'Agriculture et qui a commencé à la fin de l'année 1999. Son objectif est la mise en œuvre d'un outil d'aide à l'expertise en microbiologie prévisionnelle tenant compte des conditions de croissance des microorganismes mais aussi des paramètres liés à la nature des aliments ainsi qu'aux procédés de transformation.

Son originalité tient au fait qu'il s'appuie à la fois sur des modèles statistiques et sur une base de données contenant des résultats expérimentaux, provenant d'environ 700 publications scientifiques choisies comme les plus pertinentes. À l'issue de ce projet, une base de données relationnelle, une base de graphes conceptuels [Sow84, CM92] et un Moteur d'Interrogation Élargie (MIEL) ont été conçus. De plus, une ontologie décrivant les connaissances du domaine du risque alimentaire a été mise en œuvre. Nous présentons dans les deux sections suivantes l'ontologie et le moteur d'interrogation MIEL.

L'ontologie du domaine du risque alimentaire

A l'issus du projet Sym'Previus, une *ontologie du domaine du risque alimentaire* a été conçue afin d'avoir un modèle de données commun représentant le vocabulaire utilisé pour décrire les informations et les relations sémantiques des données à exploiter.

Taxonomie. Il s'agit d'une structure permettant d'organiser un ensemble de termes dans une hiérarchie. Cette taxonomie est définie par un couple $(T, \preceq)$ où T représente les termes du domaine constituant le vocabulaire commun utilisé pour décrire et interroger les données du domaine d'application. Il s'agit plus précisément d'un ensemble fini et non-vide de termes pertinents du domaine du risque alimentaire. La taxonomie contient au total 428 termes (e.g. $food$, $microorganism$, $factor$, $meat$, $milk$). La relation $\preceq$ est une relation de spécialisation qui est réflexive et transitive sur T. Par exemple, dans la partie taxonomie donnée en introduction en Figure 3, nous avons $Listeria \preceq Microorganism$ interprétée comme "$Listeria$ est une spécialisation de $Microorganism$" ou bien que "$Microorganism$ est une généralisation de $Listeria$". Nous avons donné un extrait de cette taxonomie en Figure 3 dans l'introduction de cette partie.

Dans la base de données Sym'Previus, les termes de l'ontologie sont utilisés pour spécifier le schéma de la base de données relationnelle et de la base de graphes conceptuels.

Schéma relationnel. Il représente un ensemble de *relations sémantiques* du domaine du risque alimentaire portant sur des concepts représentés par des termes déclarés dans la partie taxonomie de l'ontologie.
Relation sémantique. Une relation sémantique est représentée par une signature composée du nom de la relation et d'un ensemble de concepts. Les concepts apparaissant dans les signatures des relations sont appelés *attributs* de la relation. La signature d'une relation sémantique r est notée comme suit :
$r(attribut_1, attribut_2, \ldots, attribut_n)$. Nous notons $attrs(r)$ l'ensemble des attributs de la signature de la relation sémantique r déclarée dans le schéma relationnel.

Le schéma relationnel de l'ontologie Sym'Previus contient une trentaine de relations sémantiques. Par exemple (cf. Figure 1.2) :

$foodFactorMicroorganism(Food, Factor, Microorganism)$ est une relation sémantique qui exprime pour un aliment et pour une valeur d'un facteur expérimental (e.g. température) le nom d'un microorganisme (e.g. Listeria) qui est capable d'y croître.

Dans l'ontologie du risque alimentaire Sym'Previus la distinction entre les concepts et les termes n'est pas explicite. Nous considérons les *termes* apparaissant comme attributs de relation sémantique comme étant des *concepts*.

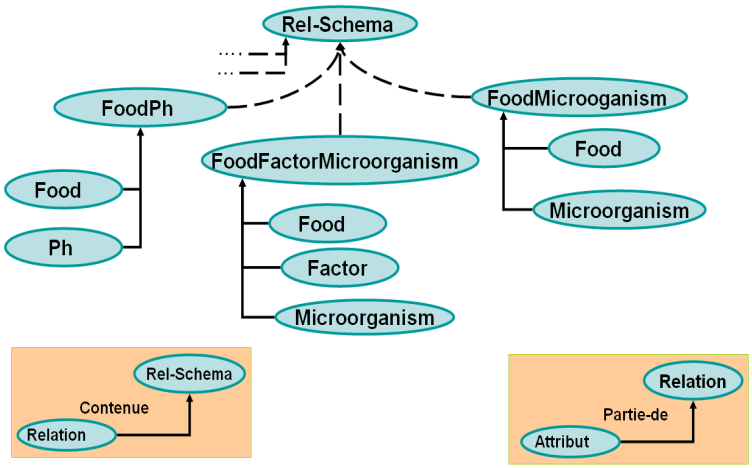

FIG. 1.2 – Extrait du schéma relationnel de l'ontologie du risque alimentaire Sym'Previus

L'ontologie Sym'Prévius a été fournie sous la forme d'un document XML. Nous donnons dans l'Annexe B la DTD et un extrait de cette ontologie.

Le Moteur d'Interrogation ELargie (MIEL)

Le moteur MIEL réalisé par Buche et al. [BDBHH04] est un système d'interrogation flexible et unifié de données. Il joue le rôle d'un médiateur intégrant différentes sources de données et décrites selon le même schéma. En effet, pour répondre aux requêtes de l'utilisateur exprimées en fonction du contenu de l'ontologie, le moteur MIEL exploite simultanément deux bases de données distinctes (cf. architecture fonctionnelle d'e.dot en Figure 1.3) :

- ⋆ une base de données relationnelle dans laquelle est stockée l'information structurée et
- ⋆ une base de connaissances exprimée sous forme de graphes conceptuels dans laquelle est stockée l'information faiblement structurée.

L'interrogation se fait par le biais d'une interface graphique dans laquelle l'utilisateur exprime sa requête en spécifiant les valeurs qu'il souhaite rechercher pour certains attributs. Dans le système MIEL toute requête est traduite par deux requêtes dans le langage d'interrogation de la base de données relationnelle et l'autre dans le langage d'interrogation de la base de graphes conceptuels.

Une extension du système MIEL a été effectuée dans le système MIEL++ [BDBHT06]. Grâce à MIEL++ il est possible d'interroger les données collectées du Web et stockées dans l'entrepôt thématique conçu dans le projet e.dot en combinaison avec les deux sources de données existantes relationnelle et graphes conceptuels, voir la Figure 1.3.

1.2.2 Chaîne de traitements des informations du Web

En vue d'obtenir des données provenant du Web qui soient pertinentes pour les domaines d'application et exploitables par le moteur d'interrogation, une chaîne de traitements des documents a été établie. Cette chaîne de traitements des informations du Web, montrée dans la Figure 1.3, est composée des étapes suivante : dans un premier temps, on collecte (*crawling*) les documents potentiellement pertinents par rapport au domaine visé représenté par l'ontologie

Sym'Previus. Ces documents sont de nature hétérogène : HTML, Pdf, etc. On les soumet ensuite à une étape de *filtrage* pour recueillir les documents les plus pertinents [Mez05]. Ces derniers sont soumis à une phase *d'extraction d'informations structurées* et de transformation en un format XML représentant la structure de ces informations par des tags purement syntaxiques. Enfin, dans le but d'avoir des données homogènes décrites selon le même vocabulaire, nous appliquons une procédure d'enrichissement sémantique de ces données par le contenu de l'ontologie du domaine. C'est cette étape d'enrichissement sémantique que nous allons décrire dans cette partie de la thèse. Il s'agit de décrire les données tabulaires par les concepts, les termes et les relations contenus dans l'ontologie. Le résultat de cette étape est représenté dans des documents conformes à un format nommé SML (*Semantic Markup Language*) que nous présentons dans le chapitre 2.

Il faut remarquer que les informations provenant des fournisseurs de contenus (sites bibliographiques, documents Excel) peuvent être soumis aux mêmes traitements. Ils sont alors pris en compte par l'étape d'extraction.

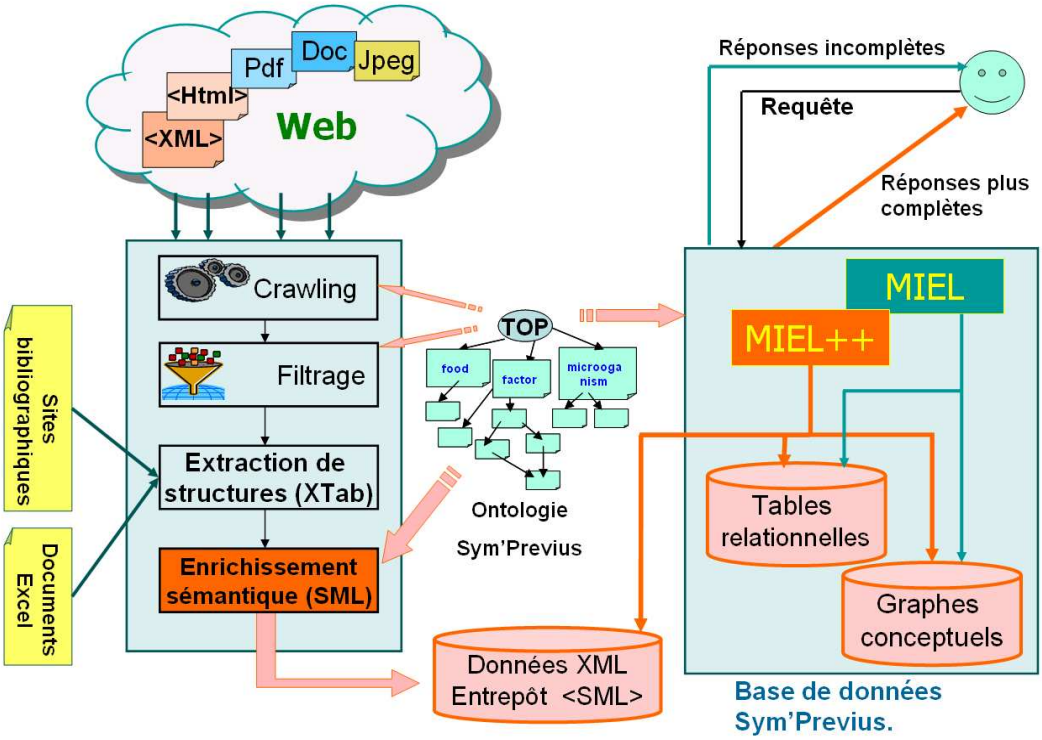

FIG. 1.3 – Une architecture fonctionnelle du projet e.dot

Après l'étape d'enrichissement sémantique, les informations sont prêtes pour être exploitées, par exemple dans le cadre d'une application d'intégration de données du Web avec des données déjà stockées : pour répondre à une requête d'un utilisateur, le système interroge simultanément l'ensemble des sources de données déjà existantes (bases de données relationnelle et graphes conceptuels) en combinaison avec les données SML provenant du Web.

Comme nous le montrons dans l'architecture fonctionnelle du projet e.dot présentée en Figure 1.3, l'ontologie a un rôle central dans la plupart des étapes du traitement des informations du

Web : commençant par l'acquisition, le filtrage, l'enrichissement sémantique et l'interrogation des données enrichies et stockées dans l'entrepôt.

1.2.3 Tâches d'acquisition et de pré-traitement des documents du Web

Dans les sous-sections suivantes nous présentons brièvement comment les informations sont collectées, filtrées, validées puis uniformisées et représentées en XML afin de les préparer pour l'enrichissement sémantique par le vocabulaire contenu dans l'ontologie.

Acquisition et filtrage des données du Web

La chaîne d'acquisition des données du Web, dans le cadre du projet e.dot, s'est focalisée, dans un premier temps, sur la découverte et la réception de documents *HTML* pertinents. Par documents pertinents, il s'agit de tout document contenant des informations centrées sur la thématique du risque alimentaire et en particulier des résultats expérimentaux concernant l'impact de germes sur certaines catégories d'aliments.

Dans un deuxième temps, on s'intéresse à l'acquisition de documents PDF. Pour prospecter le Web à la recherche de documents *PDF*, un module de *crawl* spécifique, s'appuyant sur le moteur de recherche *Google* [GC], a été mis en œuvre. Ce module parcourt le Web à la recherche de documents potentiellement intéressants, c'est-à-dire, une publication scientifique au format PDF, incluant au moins un tableau contenant des données expérimentales et éventuellement des références bibliographiques, un résumé, une introduction, un paragraphe "*Matériels et méthodes*" et une conclusion.

La chaîne d'acquisition et de filtrage des documents du Web se décompose en deux parties :

1. Un module de recherche et de collecte des documents HTML du Web, appelé "*crawler*" : ce module parcourt automatiquement le Web, de lien en lien, et rapatrie chaque nouveau document *contenant une liste de termes de l'ontologie du domaine.* Ce module a été étendu afin d'intégrer un filtrage des documents HTML en utilisant des techniques issues du domaine de l'intelligence artificielle, permettant une caractérisation multicritères des pages à retenir (pour plus de précisions sur cette partie, voir le travail de Mezaour [Mez05]).

2. Un module axé sur la découverte et la récupération des documents *PDF* du Web. Il a été en effet constaté que les informations pertinentes pour la construction de l'entrepôt e.dot étaient en grande majorité dans des documents de ce type, ce qui justifie des développements spécifiques aux documents de ce format.

Extraction et représentation unifiée d'informations structurées (XTab)

Afin de fournir une entrée unifiée au module d'enrichissement sémantique, une DTD (*Définition de Type de Document*) a été définie (cf. Annexe DTD XTab C.2) pour représenter de manière uniforme les tableaux extraits à partir de documents hétérogènes. La phase d'extraction et de conversion consiste à remplir les différents champs des documents XML se conformant au format XTab.

Les données contenues dans les tableaux sont tout d'abord représentées en XML en utilisant des tags purement syntaxiques (e.g. *line*, *cell*, *title*, *column-title*) qui sont indépendants du domaine d'application. Les tableaux, une fois extraits à partir des documents d'origine (HTML, PDF ou Excel), sont automatiquement représentés en XML où le contenu du tableau est représenté par une liste d'éléments représentant les lignes et chaque ligne est représentée par une liste

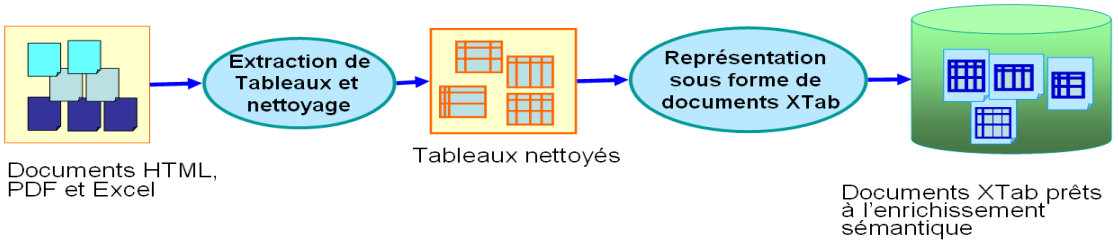

FIG. 1.4 – Le module Any2XTab

d'éléments représentant les cellules du tableau. De plus, lorsque le titre du tableau et les titres des colonnes ont pu être extraits, ils sont également représentés dans le document XTab. Des structures de tableaux plus complexes (e.g. colonnes groupées, tableaux croisés) nécessitent des heuristiques, telles que celles utilisées dans [PCS04], pour extraire les données et les représenter ensuite dans le format XTab. Ce format XTab a été défini dans le cadre du projet e.dot [e.d04]. Ce traitement est effectué par le module Any2XTab présenté dans la Figure 1.4.

Products	**pH values**
Cultivated mushroom	5.00
Crab	6.60

TAB. 1.7 – *approximative pH of some food products*

Dans la Figure 1.5, nous montrons la représentation XTab du tableau donné en Figure 1.7.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?> <table>
    <title>
        <table-title>approximative pH of some food products</table-title>
        <column-title>Products</column-title>
        <column-title>pH values</column-title>
    </title>
    <nb-col>2</nb-col>
    <content>
        <line>
            <cell>cultivated mushroom</cell>
            <cell>5.00</cell>
        </line>
        <line>
            <cell>crab</cell>
            <cell>6.60</cell>
        </line>
    </content>
</table>
```

FIG. 1.5 – *Représentation XTab du tableau 1.7*

Chapitre 2

Enrichissement sémantique d'informations structurées

Sommaire

2.1 Méthode d'enrichissement sémantique de tableaux 28
2.1.1 Présentation du format SML 28
2.1.2 Procédures d'appariement des données avec le contenu de l'ontologie 30
2.1.3 Identification des colonnes du tableau 31
2.1.4 Identification des relations sémantiques présentées dans les tableaux 33
2.1.5 Instantiation des relations sémantiques identifiées 38
2.1.6 Trace du contexte et des traitements 41
2.2 Algorithme d'enrichissement sémantique de tableaux 43
2.3 Génération automatique de la DTD SML à partir de l'ontologie 47
2.4 Découverte de relations candidates à l'alimentation d'un entrepôt thématique 48
2.4.1 Capture de l'information non reconnue dans des attributs supplémentaires 49
2.4.2 Représentation de tableaux par une relation générique 50
2.5 Interrogation des documents SML 51
2.5.1 Exemple d'interrogation 52
2.5.2 Exploitation des attributs supplémentaires 52
2.5.3 Exploitation des relations génériques 53
2.5.4 Interrogation par les vues pré-définies dans MIEL++ 53
2.6 Conclusion 55

L'objectif de l'enrichissement sémantique est d'exploiter une ontologie du domaine pour identifier, de façon plus ou moins précise, les données apparaissant dans un tableau. Il s'agit, tout d'abord, de traiter le contenu de chaque cellule en mettant en correspondance sa valeur avec les termes de l'ontologie. Ensuite, en s'appuyant sur les termes identifiés dans les cellules du tableau et éventuellement sur les titres de colonnes et le titre du tableau, il s'agit d'identifier le concept de l'ontologie représenté dans chacune des colonnes. Enfin, sachant qu'un tableau est créé initialement pour représenter des relations entre différentes entités, nous souhaitons retrouver puis représenter une ou plusieurs relations sémantiques entre les données d'une même ligne. Ces relations sont identifiées grâce à la liste des relations sémantiques décrites dans l'ontologie par leur signature et grâce aux concepts de l'ontologie préalablement identifiés dans les

colonnes du tableau. Enrichir les données représentées dans un tableau mis au format XTab, signifie donc décrire chaque ligne du tableau par les relations sémantiques et les termes présents dans l'ontologie.

Dans notre méthode d'enrichissement sémantique, nous nous interdisons de rejeter des données que nous n'avons pas pu identifier avec certitude afin de ne pas perdre d'informations susceptibles d'intéresser l'utilisateur. Pour garder trace des traitements ayant été effectués sur le contenu du tableau lors de leur mise en correspondance avec l'ontologie ainsi que l'estimation du degré d'identification (exacte, partielle ou nulle) nous avons défini une série d'indicateurs associés à chaque information (relation, attribut ou valeur). Ces indicateurs peuvent être exploités lors de l'interrogation.

Ce chapitre est organisé de la manière suivante : en Section 2.1, nous présentons la méthode automatique d'enrichissement sémantique, la génération automatique de la DTD des documents SML en Section 2.3. En Section 2.2 nous donnons l'algorithme qui implémente cette méthode d'enrichissement sémantique. Ensuite, en Section 2.4 nous proposons une étude des différents types de résultats que nous obtenons par la méthode d'enrichissement sémantique. En Section 2.5 nous montrons comment les données enrichies sémantiquement peuvent être interrogées et comment cette interrogation peut être guidée par les indicateurs de traitements que nous conservons. Enfin, nous concluons ce chapitre en Section 2.6.

2.1 Méthode d'enrichissement sémantique de tableaux

Nous présentons, dans cette section, notre méthode d'enrichissement sémantique d'informations contenues dans des tableaux et représentées dans le format XTab. La première originalité de cette méthode est que l'unique source de connaissances utilisée pour l'enrichissement sémantique est l'ontologie du domaine. La seconde originalité est que la mise en correspondance entre le contenu des tableaux et le contenu de l'ontologie est complètement automatique. L'inconvénient d'une telle approche automatique et non supervisée est de conduire à des ambiguïtés ou à une interprétation partielles ou à de mauvaises mises en correspondance trouvées. Cependant, nous proposons de conserver toutes les interprétations afin de permettre leur exploitation lors de l'interrogation. Cette flexibilité est la troisième originalité de notre méthode.

Nous décrivons tout d'abord le format de document SML que nous avons défini pour représenter le résultat de l'enrichissement sémantique. Ensuite, nous donnons les trois étapes principales de notre méthode. Enfin, nous présentons l'algorithme XTab2SML qui implémente notre méthode d'enrichissement sémantique.

2.1.1 Présentation du format SML

Le format de documents SML (*Semantic Markup Language*) est un format fondé sur le langage XML que nous avons défini pour représenter le contenu d'un tableau enrichi par les concepts, les termes et les relations sémantiques déclarés dans l'ontologie du domaine d'application. Pour représenter un tableau dans un document SML, nous gardons la structure en lignes du tableau de la même façon que pour les documents XTab. Cependant, les lignes ne sont plus représentées par un ensemble de cellules mais par un ensemble de relations sémantiques de l'ontologie identifiées dans les lignes du tableau. Chaque relation identifiée dans le tableau est représentée par l'imbrication d'éléments correspondant à l'ensemble des attributs appartenant à sa signature dans le schéma relationnel de l'ontologie. En résumé, dans un document SML, la plupart des tags et des valeurs possibles sont des relations, des concepts ou des termes construits à partir de l'ontologie du domaine (e.g. l'ontologie Sym'Previus).

Nous avons défini le type des documents SML dans une DTD (*Document Type Definition*). Dans cette DTD, nous spécifions l'organisation et les types des différents éléments et attributs XML représentant un document SML. Tous les documents SML produits par notre méthode sont conformes à cette DTD. Un document SML est donc composé d'un ensemble d'éléments XML organisés de manière arborescente. Dans un document SML nous trouvons principalement les éléments suivants :

table. C'est l'élément racine du document SML qui englobe les éléments *title* et *content*.

content. C'est l'élément représentant le contenu du tableau qui est structuré sous forme d'un ensemble de lignes *rowRel* qui contient un ensemble de relations sémantiques identifiées dans le tableau (e.g. *foodPh*). Une relation est représentée par l'ensemble de ses attributs décrits dans sa signature dans le schéma relationnel (e.g. *food*, *ph*). Chaque attribut représente l'ensemble de valeurs d'une colonne du tableau (ie. ensemble de cellules), pour chacune d'entre elles nous recherchons un terme de l'ontologie ayant une sémantique proche de la valeur. Si un tel terme est trouvé nous le représentons dans un tag *ontoVal*. Par exemple, pour la valeur *cultivated mushroom* nous avons trouvé le terme *mushroom* comme terme correspondant.

title. Pour chaque tableau nous conservons des informations le concernant qui seront utilisées lors de l'exploitation du document SML : le titre du tableau, les titres et le nombre de colonnes du tableau.

En Annexe D, nous donnons la DTD SML définie pour le domaine du risque alimentaire représentée dans l'ontologie Sym'Previus. En Section 2.3 nous montrons comment la DTD SML peut être générée automatiquement à partir de l'ontologie du domaine.

Exemple très préliminaire

Considérons, comme exemple, le tableau 1.7 où nous pouvons constater que l'organisation en lignes des valeurs reflète l'existence d'une relation sémantique entre les valeurs de chaque ligne. La deuxième ligne du tableau peut être interprétée comme : « *Le crabe est un aliment dont le pH vaut 6.60* ». Supposons que la relation sémantique nommée *foodPH* qui représente les liens entre les aliments et leur valeur de pH soit déclarée dans l'ontologie.

L'objectif de la méthode d'enrichissement sémantique est de traduire un tableau représenté dans un document XTab, tel que celui présenté en Figure 1.5, en un tableau représenté dans un document SML tel que celui présenté en Figure 2.1. Dans le document SML, la relation sémantique *foodPh*, que nous pouvons identifier dans les lignes du tableau, est représentée par ses deux attributs *food* et *Ph*, puis instanciée par les valeurs des cellules du tableau.

Pour instancier la relation, nous cherchons à associer un ou plusieurs termes de la taxonomie à chacune des valeurs du tableau. Si une des valeurs n'apparaît pas telle quelle dans la taxonomie, nous utilisons alors des techniques d'appariement de chaînes de caractères afin de trouver des termes similaires.

Dans le tableau de l'exemple 1.7, la valeur de la première colonne *crab* appartient telle quelle à la taxonomie. Cependant, la valeur *cultivated mushroom* n'existe pas ainsi dans la taxonomie ; néanmoins, nous proposons de lui associer le terme *mushroom* grâce à l'utilisation d'une procédure d'appariement. Cette valeur est représentée dans le document SML par un tag $< ontoVal >$ tout en conservant la valeur d'origine *cultivated mushroom* dans le tag $< originalVal >$. Ainsi,

```
<table>
    <title>
        <table-title>approximative pH of some food products </table-title>
        <column-title> Products </column-title>
        <column-title>pH values</column-title>
    </title>
    <content>
        <rowRel>
            <foodPh>
                <food>
                    <ontoVal>mushroom</ontoVal>
                    <originalVal> cultivated mushroom </originalVal>
                </food>
                <ph>
                    <originalVal>5.00</originalVal>
                </ph>
            </foodPh>
        </rowRel>
        <rowRel>
        <foodPh>
            <food>
                <ontoVal>crab</ontoVal>
                <originalVal>crab</originalVal>
            </food>
            <ph>
                <originalVal>6.60</originalVal>
            </ph>
        </foodPh>
    </rowRel>
    </content>
</table>
```

FIG. 2.1 – *Une représentation simplifiée en SML du tableau XTab de la Figure 1.5.*

la valeur d'origine peut être montrée à l'utilisateur dans le résultat d'une requête, même si cette requête est formulée en fonction des termes de l'ontologie.

2.1.2 Procédures d'appariement des données avec le contenu de l'ontologie

Lors de la comparaison des données du tableau avec le contenu de l'ontologie, un prétraitement est effectué sur les valeurs du tableau, les termes et les concepts pour lesquels nous cherchons des appariements. Tout d'abord, toute valeur, tout concept et tout terme est représenté dans un ensemble de mots. Ensuite, il s'agit de supprimer tous les *mots vides* et d'appliquer une procédure de lemmatisation à tous les mots considérés dans la comparaison.

Mot vide. Un *mot vide* désigne tout élément d'un texte n'ayant pas de référence concrète ou notionnelle dans la réalité et dont le rôle est d'assurer la liaison ou la cohérence entre les principales parties d'un texte. Il s'agit des articles, des conjonctions, des adjectifs non qualificatifs, des prépositions et de la ponctuation.

Lemmatisation. Il s'agit d'une procédure qui permet de ramener un mot portant des marques de flexion (par exemple, la forme conjuguée d'un verbe) à sa forme de référence (dite *lemme*). Autrement dit, il s'agit, dans le cadre d'une analyse de texte par ordinateur ou d'un traitement automatique de la langue de manière générale, de reconnaître un mot par rapport à sa forme de

base, et cela, quelle que soit la forme sous laquelle il apparaît dans un texte.

Exemple : lemmatisation(sèches) = sec.

Afin de mesurer la similarité syntaxique entre les données du tableau et le contenu de l'ontologie, nous utilisons une procédure d'appariement entre des chaînes de caractères représentées sous la forme d'ensembles de mots lemmatisés desquels nous avons supprimé les mots vides. Cette procédure est fondée sur des critères syntaxiques telle que les mesures de similarité syntaxiques présentées en Section 4.2 du chapitre 4.

Nous différencions la procédure d'appariement utilisée lors de l'identification des colonnes de celle utilisée lors de l'instantiations des relations sémantiques. En effet, le critère de similarité doit être plus strict lorsqu'une valeur est utilisée pour permettre de trouver des mises en correspondance entre les valeurs des colonnes et les attributs des relations sémantiques. Nous verrons que lors de l'instantiation des relations, plusieurs termes de l'ontologie peuvent être associés à une même valeur.

2.1.3 Identification des colonnes du tableau

L'identification des relations sémantiques représentées dans le tableau est effectuée en deux étapes. La première, présentée dans cette section, consiste à identifier un concept de la taxonomie qui représente les valeurs de chaque colonne du tableau. La seconde étape que nous présentons dans la section suivante va consister, compte tenu des concepts associés à chacune des colonnes du tableau, à reconnaître les relations sémantiques qui sont représentées dans les données organisées en colonnes.

L'identification des colonnes du tableau est fondée sur deux éléments d'information : le contenu des colonnes, utilisé en priorité puis les titres des colonnes qui sont utilisés dans le cas où le contenu des colonnes ne nous permet pas de leur associer un concept de l'ontologie représenté par un terme que nous nommons A-Term.

Définition – A-Term. Un A-Term est un terme déclaré dans la taxonomie et qui apparaît au moins une fois comme attribut dans la signature d'une relation sémantique déclarée dans le schéma relationnel de l'ontologie. L'ensemble de tous les A-Term est noté ATS et est défini comme suit :

$$ATS = \{a \mid \exists\, r\ tq\ a \in\ attrs(r)\}$$

En vue d'associer un A-Term à chaque colonne du tableau, nous exploitons tout d'abord le contenu de la colonne. Dans un premier temps, nous mettons en correspondance [2] chaque valeur de la colonne avec les termes contenus dans la taxonomie afin d'identifier le terme le plus proche. Soit $TS(c)$ l'ensemble des termes associés aux valeurs d'une colonne c. Pour désigner le A-Term le plus représentatif des valeurs contenues dans une colonne c du tableau nous retenons l'ensemble des A-Term qui satisfont les deux conditions suivantes :

1. t est un généralisant commun dans la taxonomie relativement à la relation $\preceq$, d'un sous-ensemble non vide $G(t, c)$ de l'ensemble $TS(c)$ de termes associés aux valeurs de la colonne

[2] Nous appliquons des mesures de similarité fondées sur les tokens 4.2 sur les ensembles de mots lemmatisés et auxquels les mots vides ont été enlevés.

c ;

2. Nous introduisons une notion de seuil qui doit être satisfait par la proportion de termes associés aux valeurs d'une colonne qui soient généralisés par un A-Term. Soit th une valeur réelle appartenant à l'intervalle [0..1]. L'ensemble de tous les A-Term vérifiant cette contrainte est noté $ATCandidat(c, th)$ qui est défini comme suit :

$$ATCandidat(c, th) = \{t \mid t \in TS(c) \; et \; \frac{|G(t, c)|}{|c|} \geq th\}$$

où $|c|$ représente le nombre de valeurs contenues dans la colonne c.

Parmi ces A-Term candidats, nous retenons le A-Term le plus spécifique qui subsume l'ensemble le plus grand de termes associés aux valeurs de la colonne. Ce A-Term le plus représentatif, pour une colonne c et un seuil th donnés, noté $ATRep(c, th)$, est défini comme suit :

$$ATRep(c, th) = \{t \mid t \in ATCandidat(c, th), \; \neg\exists \; t' \; tq \; t' \in ATCandidat(c, th)$$
$$et \; |G(t', c)| > |G(t, c)|, \neg\exists \; t'' \; tq \; t'' \in ATCandidat(c, th)$$
$$et \; |G(t'', c)| = |G(t, c)| \; et \; t'' \preceq t\}$$

Par construction, pour un seuil supérieur à 0,5, il ne peut y avoir qu'un A-Term représentatif. De manière générale, si l'ensemble $ATRep$ contient plusieurs A-Terms nous gardons uniquement le premier trouvé. On peut noter que les expérimentations que nous avons effectuées ont montré que lorsque le seuil est assez élevé l'ensemble $ATRep$ contient zéro ou un seul A-Term représentatif.

Si pour une colonne, nous n'avons pas pu trouver de A-Term, en exploitant les valeurs, nous exploitons le titre de la colonne s'il est disponible. Nous le comparons syntaxiquement avec les A-Terms contenus dans l'ontologie. Soit $TS(titre)$ l'ensemble représentant le A-Term le plus proche du titre de la colonne.

Nous exploitons en priorité les valeurs des colonnes car, dans le cas où nous arrivons à identifier un grand nombre de valeurs, le A-Term identifié est souvent pertinent. De plus, l'exploitation du titre peut conduire à une mauvaise association. Par exemple, dans le titre de colonne "*Listeria Qty*", le A-Term *Microorganism* est identifié dans ce titre cependant la colonne ne représente pas les noms de microoganismes mais plutôt leur quantité.

Si pour une colonne, aucun A-Term n'a pu être identifié ni par l'exploitation des valeurs ni par l'exploitation du titre, nous gardons tout de même les informations de la colonne dans le document SML. Nous associons, à cette colonne, un A-Term générique que nous nommons *attribute*.

Une fois qu'un A-Term contenu dans l'ontologie ou un A-Term générique a été associé à chacune des colonnes du tableau nous construisons alors le *schéma du tableau*.

Definition –schéma du tableau. Le schéma d'un tableau T est l'ensemble de couples, de la forme (c, t), représentant pour chacune des colonnes c du tableau T le A-Term t le plus représentatif qui lui est associé pour un seuil th donné. Cet ensemble noté $tabSch(T)$ est défini comme suit :

$$tabSch(T) = \{(c,t)|t \ \in ATRep(c,th) \ ou \ [t \in TS(title(c)) et \ ATRep(c,th) = \emptyset] \ et$$
$$[(t = attribute) \ et \ ATRep(c,th) = \emptyset \ et \ TS(title(c)) = \emptyset]\}$$

où $title(c)$ représente le titre de la colonne c dans le tableau.

Pour des raisons de simplicité, nous référons à chaque colonne par son rang d'apparition dans le tableau.

Exemple. Dans le tableau 2.1, les termes *crab* et *chicken* contenus dans l'ontologie ont été associés aux valeurs *ground crab* et *chicken*. Si le seuil *th* est de 0.5, le A-Term le plus spécifique qui subsume ces termes est le A-Term *Food*. Pour la seconde colonne, aucun A-Term n'a été identifié car elle ne contient que des valeurs numériques et le titre n'a pas non plus permis de trouver un A-Term ; un A-Term générique *attribute* lui est donc associé. Les A-Terms *lipid* et *calorie* ont été associés aux deux dernières colonnes grâce à leur titre.

Products	**Item**	**Lipids**	**Calories**
whiting with lemon	100 g	7.8 g	92 kcal
ground crab	150 g	11.25 g	192 kcal
chicken	250 g	18.75 g	312 kcal

TAB. 2.1 – *Nutritional Composition of some food products*

Le schéma du tableau $T1$ représenté en Table 2.1 est donc :

$$tabSch(T2) = \{(1, food), \ (2, attribute), \ (3, lipid), \ (4, calorie)\}$$

2.1.4 Identification des relations sémantiques présentées dans les tableaux

Nous présentons dans cette section comment nous identifions la ou les relation(s) sémantique(s) représentée(s) dans le tableau. Nous utilisons le schéma du tableau que nous avons obtenu à l'étape précédente. Cette identification est réalisée par la comparaison des A-Terms associés aux colonnes et les attributs apparaissant dans les signatures des relations sémantiques déclarées dans l'ontologie du domaine. Une relation sémantique est considérée comme représentée dans un tableau si l'ensemble de ces attributs est complètement ou partiellement inclus dans l'ensemble des A-Term contenu dans le schéma de ce tableau. Un appariement exact entre le schéma du tableau et la signature d'une relation sémantique est évidemment le cas idéal. Toutefois, dans la plupart des cas, nous obtenons plusieurs appariements possibles entre des sous ensembles des A-Terms contenus dans le schéma du tableau et les signatures des relations sémantiques. De plus, certaines relations sémantiques ne sont parfois appariées que partiellement : tous les attributs de leur signature n'ont pas pu être mis en correspondance avec les A-Terms du schéma du tableau. En effet, cela est dû au fait que des informations, collectées et extraites automatiquement à partir du Web, peuvent être implicites, ou représentées par des constantes dans le titre ou dans le texte environnant le tableau ou que les créateurs des données n'utilisent pas les mêmes attributs pour représenter les relations sémantiques.

Dans la suite nous montrons les différents cas de figures qui peuvent se présenter lors de l'identification des relations sémantiques représentées dans le schéma du tableau. Nous allons voir également comment ils sont représentés en SML.

Relations complètement représentées

Une relation est considérée comme ***complètement représentée*** dans un tableau si chaque attribut de sa signature est égal ou subsume un A-Term distinct du schéma du tableau. Soit l'ensemble noté $CInTab(T)$ l'ensemble représentant des relations sémantiques qui sont complètement représentées dans un tableau T et que nous définissons comme suit :

$$CInTab(T) = \{r \mid \forall a \in attrs(r), \exists (c,t) \in tabSch(T)\ tq\ (t \preceq a)\}$$

Supposons que les relations sémantiques *foodLipid*, *foodCalorie* et *foodPh* soient déclarées dans l'ontologie et que les deux relations *foodLipid* et *foodCalorie* exprime "*le nombre de lipides (resp. de calories) contenues dans* ***100****g d'aliment*" car les experts du domaine ont considéré que le poids est normalisé. Dans le tableau présenté dans la table 2.1, les relations sémantiques ont été identifiées de la façon suivante :

- ⋆ *foodLipid*, est complètement représentée par les valeurs de la première et de la troisième colonne.
- ⋆ *foodCalorie*, est complètement représentée par les valeurs de la première et de la quatrième colonne.

Dans ce même exemple, nous montrons comment les attributs supplémentaires sont ajoutés aux relations représentées dans le document SML. Un attribut supplémentaire est un attribut qui est associé à une colonne du tableau pour laquelle aucun A-Term n'a été identifié. Par exemple, la seconde colonne *Item* du tableau 2.1, n'a pas été identifiée et par conséquent ne participe à aucune relation complètement identifiée, nous représentons alors les valeurs de cette colonne par un attribut générique que nous ajoutons à toute relation identifiée dans le tableau. Dans un tel cas, la représentation des informations des colonnes non identifiées dans des attributs supplémentaires permer de guider l'utilisateur et de le conduire vers de meilleures interprétations des données. Considérons le cas où l'attribut générique n'a pas été ajouté, nous interprétons alors les données de la troisième ligne du tableau comme : "***100g*** *de poulet correspondent à 312 calories*". En revanche, lorsque l'attribut générique représentant les valeurs de la deuxième colonne est ajouté à chacune des relations, l'interprétation, par exemple, de la troisième ligne serait plutôt : "***250g*** *de poulet correspond à 312 calories*".

En Figure 2.2 nous montrons la representation SML des relations complètement identifiées *foodLipid* et *foodCalorie* avec un attribut supplémentaire :

Afin qu'un moteur d'interrogation des documents SML puisse s'adapter et être informé de la présence d'attributs supplémentaires dans le tableau, un attribut XML *additionalAttr* est ajouté à chaque ligne du tableau. Il prend deux valeurs possible "*yes*" ou "*no*".

Afin de garder trace du type de l'identification des relations dans le tableau, nous ajoutons à chaque élément du document SML représentant une relation, un attribut XML nommé *relType* qui indique si la relation a été identifiée complètement ou partiellement.

Nous représentons également des relations partiellement identifiées dans le tableau, dans le but de garder toute information pouvant intéresser l'utilisateur. En effet, des attributs peuvent être absents, d'une part parce qu'ils peuvent être représentés par des valeurs constantes et donc non exprimés dans le titre du tableau, d'autre part si les experts ne formulent pas la même liste

```
<table>
    ...
    <content>
        <rowRel additionalAttr="yes">
            <foodLipid relType ="completeRel">
                <food>...</food>
                <lipid> ... </lipid>
                <attribute> ...</attribute>
            <foodLipid>
            <foodCalorie relType ="completeRel">
                <food>...</food>
                <calorie> ... </calorie>
                <attribute> ...</attribute>
            </foodCalorie>
        <rowRel>
        ...
    <content>
<table>
```

FIG. 2.2 – *Représentation SML de relations complètement identifiées*

d'arguments pour une relation sémantique[3].

Une relation est donc partiellement représentée dans le tableau s'il existe au moins deux attributs inclus dans le schéma du tableau, au moins deux, parce que nous considérons la relation binaire comme la relation minimale qui puisse être instanciée.

Une relation sémantique est donc ***partiellement représentée*** dans un tableau si elle n'est pas complètement représentée et si au moins deux des attributs de sa signature sont égaux ou subsument deux A-Terms distincts du schéma du tableau. Soit $PInTab(T)$ l'ensemble des relations sémantiques qui sont partiellement représentées dans un tableau T et que nous définissons comme suit :

$$PInTab(T) = \{r \mid \neg(r \in CInTab(T))\ et\ (\exists a \in attrs(r)\ tq\ \exists(c,t) \in tabSch(T)\ et\ t \preceq a)\ et$$

$$(\exists a' \in attrs(r)\ tq\ \exists(c',t') \in tabSch(T)\ et\ t' \preceq a')\ et\ (c \neq c')et(a \neq a')\}$$

Nous considérons deux cas où les relations peuvent être partiellement identifiées.

Relations partiellement représentées avec des attributs Null

Il s'agit du cas où un attribut de la relation sémantique n'a été apparié à aucun A-Term du schéma du tableau. Par exemple, dans le tableau 2.1, la relation sémantique $foodAmountLipid$, définie dans l'ontologie par l'ensemble des attributs $food$, $amount$ et $lipid$, est partiellement représentée dans le schéma du tableau, car l'attribut $amount$ n'est pas représenté tel quel dans $tabSch(T2)$ et aucun A-Term du schéma du tableau n'est subsumé par l'attribut $amount$. Dans la Figure 2.3, nous montrons la représentation SML de la relation $foodAmountLipid$:

Notons que lorsqu'une relation est partiellement représentée, les attributs n'ayant pas été identifiés dans le schéma du tableau sont représentés dans le document SML par des tags vides tels que $< amount \quad attrType = "Null"/ >$. Dans ce cas, dans le document SML, la valeur de l'attribut XML $relType = "partialNull"$ indique qu'il s'agit d'une relation partiellement identifiée avec un attribut non reconnu et donc Null.

[3](ex : Nom de l'expérimentateur, temps, ...) Cela pose le problème de la discrimination entre les attributs obligatoires et optionnels d'une relation sémantique

```
<table>
    <content>
        <rowRel dditionalAttr="yes">
        ...
            <foodAmountLipid relType ="partialNull">
                <food attrType="Normal">...</food>
                <amount attrType="Null"/>
                <lipid attrType ="Normal"> ... </lipid>
                <attribute attrType="generic"> ...</attribute>
           <foodAmountLipid>
        <rowRel> ...
    <content>
<table>
```

FIG. 2.3 – *Représentation SML de relation partiellement identifiée avec des attributs Null*

Dans cet exemple, l'information du tableau qui correspond à l'attribut *amount* et qui n'a pas été identifiée est en fait représentée dans l'attribut supplémentaire. Si lors de l'interrogation, l'utilisateur souhaite que tous les attributs (identifiés, Null ou supplémentaires) soient affichés, il pourra alors mieux interpréter le résultat, en faisant, par exemple, le lien entre l'attribut supplémentaire et l'attribut Null *amount*.

Relations partiellement représentées avec des valeurs constantes

Il s'agit du cas où le créateur du tableau a choisi de représenter la valeur d'un attribut de relation sémantique non pas par une colonne mais par une constante contenue dans le titre du tableau. Voici un exemple de tableau illustrant ce cas de figure.

Products	**Doubling time (h)**
Minced meat	30^1
Cured raw pork	3.6^1
Frankfurters	9^1

TAB. 2.2 – *Doubling times of Listeria monocytogenes in foodstuffs*

Supposons que le schéma calculé du tableau $T3$ présenté en 2.2 soit le suivant :

$$tabSch(T3) = \{(1, food), (2, factor)\}$$

Dans ce schéma du tableau, le relation *foodFactorMicroorganism* est partiellement représentée : les attributs *food* et *factor* sont représentés dans le schéma du tableau. L'attribut *Microorganism* n'est représenté dans aucune des colonnes du tableau. Cependant nous pouvons identifier la constante *Listeria Monocytogenes* apparaissant dans le titre du tableau : "*Doubling time of* ***Listeria Monocytogenes*** *in foodstuffs*" comme une sorte de microorganisme (par la relation $\preceq$). Cette constante représente un terme de l'ontologie subsumé par un attribut de relation sémantique non identifié dans les colonnes du tableau. Il s'agit en effet d'une information importante à conserver car elle permet d'instancier un attribut de relation sémantique non identifié dans le tableau. Nous associons alors la constante à l'attribut non identifié *microorganism*. Cette constante est dupliquée dans toutes les occurrences de la relation, ie. dans toutes les lignes du tableau. En Figure 2.4 nous montrons la représentation SML de la relation $foodFactorMicroorganism$, partiellement identifiée et où l'attribut $Microorganism$ est instancié par la constante "*Listeria Monocytogenes*".

```
<table>
    ...
    <content>
        <rowRel additionalAttr="no">
            ...
            <foodFactorMicroorganism relType ="partialConst">
                <food attrType="Normal">...</food>
                <factor attrType="Normal"> ... </factor>
                <microorganism attrType ="Const"> listeria monocytogenes </microorganism>
            <foodFactorMicroorganism>
            ...
        <rowRel>
        ...
    <content>
<table>
```

FIG. 2.4 – *Représentation SML d'une relation partiellement identifiée avec un attribut représenté par une constante*

Dans le document SML, la valeur de l'attribut XML $relType = "partialConst"$ indique qu'il s'agit d'une relation partiellement identifiée avec un attribut reconnu dans une constante dans le titre du tableau.

Un attribut XML *attrType* est ajouté à tout attribut de relation sémantique permettant d'indiquer le type de l'attribut. Nous avons trois types possibles pour les attributs : (i) un attribut identifié dans le schéma du tableau $attrType = "Normal"$ (ii) un attribut identifié dans une constante dans le titre du tableau $attrType = "Const"$ ou encore (iii) un attribut non identifié dans le schéma du tableau $attrType = "Null"$.

On note que les attributs supplémentaires sont également ajoutés aux relations qui sont partiellement identifiées. En effet, certains de ces attributs supplémentaires peuvent correspondre à des attributs de relations sémantiques non reconnus dans les colonnes du tableau. De plus, ces attributs supplémentaires peuvent être considérés comme informations contextuelles pouvant aider l'utilisateur à avoir une interprétation plus précise du résultat.

Dans le cas où aucune relation sémantique n'a été identifiée ni complètement ni partiellement, nous présentons le contenu du tableau dans une relation générique nommée *relation*. Ainsi, nous conservons les liens sémantiques existants entre les valeurs des lignes du tableau même si ces liens n'ont pas été identifiés. Cette relation contient un ensemble d'attributs génériques *"attribute"*, correspondant à chaque colonne du tableau. Dans ce cas, il sera quand même possible d'interroger les documents SML par une liste de mots-clés. Plus précisément, une requête sur La *Listeria* pourra avoir pour réponse des valeurs en lien avec la *Listeria* accompagnées d'informations contextuelles comme les titres des colonnes même si celles-ci n'ont pas été identifiées. En Figure 2.5, nous montrons un exemple de relation générique présentée en SML.

Il est possible d'avoir des attributs de relations sémantiques qui soient identifiés dans les colonnes du tableau mais que leur combinaison ne permette pas d'identifier même partiellement une relation sémantique. C'est pour cela que nous avons introduit un nouvel indicateur de traitement *indMap* qui conserve les noms des A-Terms associés aux colonnes bien que la relation n'ait pas été reconnue.

```
<table>
    ...
    <content>
        <rowRel>
            <relation>
                <attribute attrMatch="notFound">  </attribute>
                <attribute attrMatch="food"> ... </attribute> ...
            </relation>
        </rowRel>
            ...
    </content>
</table>
```

FIG. 2.5 – *Représentation d'une relation générique en SML*

2.1.5 Instantiation des relations sémantiques identifiées

Pour instancier les relations identifiées par les valeurs du tableau, nous recherchons pour chaque valeur le ou les termes de l'ontologie qui lui sont proches. Nous appliquons une procédure d'appariement entre chaque valeur et les termes de l'ontologie en utilisant, d'une part, les opérateurs d'appariement fondés sur un critère syntaxique, présentés en Section 2.1.2, d'autre part, une procédure d'appariement utilisant le Web comme ressource externe. Nous avons choisi d'utiliser l'approche non supervisée PANKOW (*Pattern-based Annotation through Knowledge On the Web*) [CHS04] où des patterns sont utilisés pour catégoriser des noms propres considérés comme représentant des instances par rapport à une ontologie. Cette méthode exploite un ensemble de patterns linguistiques dont certains ont été définis par [Hea92] tel que : $< concept >$ *instance of* $< concept >$ ou $< concept >$ *such as* $< instance >$. Ces patterns sont appliqués sous la forme de requêtes évaluées sur la moteur de recherche Google[4]. Les auteurs retiennent comme critère d'évaluation du pattern, le nombre de pages dans lesquelles le pattern apparaît. Nous avons utilisé la même approche sur les données des tableaux même si elles ne sont pas nécessairement des noms propres. Nous avons appliqué un pattern général "$< valeur >$ *is a* $< terme >$" pour découvrir des relations de spécialisation entre les valeurs des tableaux et les termes de l'ontologie en utilisant le corpus du Web. Étant donnée une valeur, nous instantions le pattern pour chaque terme de l'ontologie et nous sélectionnons le meilleur terme compte tenu du nombre de pages Web trouvées. A cause de la spécificité de notre domaine, le nombre de pages peut être très faible. Ainsi, lorsque nous essayons d'associer la valeur "ice cream" à chaque terme de l'ontologie, le pattern "ice cream is a dessert" a été trouvé dans 12900 pages seulement. Heureusement, pour le pattern "ice cream is a microorganism" aucune page n'a été trouvée ! On note que le terme *dessert* ne peut pas être trouvé par un critère syntaxique.

Dans le format SML il est possible d'associer plusieurs termes pouvant être trouvés par les procédures d'appariement pour une valeur du tableau donnée. Si pour une valeur *val* du tableau, plusieurs termes de l'ontologie ont été trouvés comme similaires alors dans le document SML nous ajoutons autant d'éléments $ontoVal$ que de termes de l'ontologie trouvés. Par exemple, pour la valeur "*whiting with lemon*" deux termes de l'ontologie ont été trouvés "*whiting fillets*" et "*green lemon*" ce qui est représenté en SML par :

[4] en utilisant l'API fournie par Google.

```
<table>
...
    <content>
    ...
        <ontoVal> green lemon </ontoVal>
        <ontoVal> whiting fillets </ontoVal>
        <originalVal> whiting with lemon </originalVal>
    ...
    </content>
</table>
```

Dans le cas où, pour une valeur du tableau, aucun terme de l'ontologie n'a été trouvé par les procédures d'appariement, nous représentons alors uniquement la valeur d'origine par l'élément $originalVal$. Ceci est toujours le cas pour les valeurs numériques.

Cas des listes de valeurs dans une cellule. Jusque là, nous n'avons considéré qu'une seule valeur par cellule. Une cellule du tableau peut contenir une liste de valeurs de même type. Une liste de valeurs peut être segmentée en un ensemble de mots si les séparateurs sont distinguables.

Products	pH values
Fresh meat and fish	4.60
Cultivated mushroom	5.00
Crab, Prawns, lobster	6.60

TAB. 2.3 – *approximative pH of some food products*

Supposons que nous ayons dans un tableau une colonne aliments où les cellules contiennent des listes d'aliments et que le A-Term $food$ ait été associé à cette colonne. Pour prendre en compte ce cas de listes de valeurs et éviter les ambiguïtés, nous encapsulons l'attribut $food$ dans un élément ***foodList***, quelle que soit la structure de la valeur. Le format que nous avons défini permet de garder la valeur d'origine dans un élément XML $originalValList$ que l'on ait pu ou pas segmenter cette liste sans perdre d'informations.

Certaines listes utilisent des séparateurs de valeurs distingués, tels que ';', ',' et '-', qui pourraient permettre de segmenter la liste en plusieurs valeurs. Nous représentons cette configuration en SML en ajoutant au tag $foodList$ autant de tags $food$ qu'il y a de valeurs segmentées.

Cependant, la segmentation des listes est souvent difficile et nécessite des méthodes avancées de TALN (Traitement Automatique du Langage Naturel). Dans le cas, par exemple, de "Fresh meat and Fish", on ne sait pas s'il faut segmenter {"Fresh meat", "Fish"} ou distribuer le modifieur et obtenir {"Fresh meat", "Fresh Fish"}. Le format permet donc de considérer la liste comme une seule valeur et d'y associer les les termes de l'ontologie qui lui sont proches. Nous représentons les termes trouvés dans des tags XML ***ontoValList*** auxquels nous ajoutons les indicateurs de traitement ($indMap$, $attrMatch$). En Figure 2.7, nous montrons la représentation SML des listes de valeurs dans le cas où la segmentation a pu être effectuée et en Figure 2.8, nous donnons la représentation SML dans le cas où cette segmentation n'a pas pu être réalisée.

```
<table>
    ...
        <content>
        ...
            <rowRel additionalAttr="no">
                <foodPh>
                    <foodList>
                        <food ...>  ... </food>
                    </foodList>
                    <ph>    ...  </ph>
                </foodPh>
                ...
             </rowRel>
            ...
        </content>
</table>
```

FIG. 2.6 – *La représentation SML d'une relation avec prise en compte de listes de valeurs*

```
...
   <rowRel>
      <foodPh>
          <foodList>
              <food ...>
                  <ontoVal>Fresh meat</ontoVal>
                  <ontoVal>Sea food and fish </ontoVal>
              </food>
              <food ...> ... </food>
              <originalValList> Fresh meat and fish</originalValList>
          </foodList>
          <ph ...>... </ph>
      </foodPh>
   </rowRel>
...
```

FIG. 2.7 – *Liste de valeurs avec segmentation*

```
...
  <rowRel>
     <foodPh>
          <foodList>
              <ontoValList attrMatch="food"> Fresh meat </ontoValList>
              <ontoValList attrMatch="food"> Fish </ontoValList>
              <originalValList>Fresh meat and fish</originalValListe>
          </foodList>
          <ph ...> ... </ph>
     </foodPh>
 </rowRel>
...
```

FIG. 2.8 – *Liste de valeurs sans segmentation*

Exemple plus complet de document SML.

Nous présentons en Figure 2.9 le document SML obtenu pour l'enrichissement sémantique du tableau présenté en 2.1. Dans ce tableau, quatre relations sémantiques ont été identifiées : deux ont été complètement identifiées (*foodLipid* et *foodCalorie* pour la première, troisième et quatrième colonne ; et deux relations ont été partiellement identifiées *foodAmountLipid* et

foodAmountCalorie avec l'attribut *Amount* représenté par un attribut *Null*.

La représentation SML d'une relation est composée d'un ensemble d'attributs apparaissant dans la signature de la relation sémantique décrite dans le schéma relationnel de l'ontologie (e.g. *foodLipid*(*food*, *lipid*)). Chaque attribut correspond au *A*-Term le plus représentatif des valeurs de chaque colonne ou au *A*-Term ayant été identifié dans le titre de la colonne. Un ensemble de termes représentés dans les tags XML *ontoVal* est associé à chaque valeur du tableau. Ainsi, le terme *crab* a été associé à la valeur *ground crab* alors que trois termes différents ont été associés à la valeur *whiting with lemon* : il s'agit de *whiting Provencale*, *green lemon* et de *whiting fillets*. La valeur d'origine est conservée et représentée dans tag XML *originalVal*.

Nous montrons dans la Figure 2.9 une partie du document SML qui a été automatiquement généré à partir du document XTab de la figure Figure 2.1.

```
<table>
    <title>
        <table-title>Nutritional Composition of some food products </table-title >
        <column-title> Product </column-title>
        <column-title>Item</column-title>
        <column-title>lipids</column-title>
        <column-title>calories</column-title>
    </title>
    <column-nb> 4 </column-nb>
    <content>
        <rowRel additionalAttr="yes">
            <foodLipid relType="completeRel">
                <food attrType="Normal">
                    <ontoVal> whiting Provencale </ontoVal>
                    <ontoVal>green lemon </ontoVal>
                    <ontoVal> whiting fillets </ontoVal>
                    <originalVal> whiting with lemon </originalVal>
                </food>
                <lipid  attrType="Normal">
                    <originalVal> 7.8 g</originalVal>
                </lipid>
                <attribute attrMatch="notFound" attrType="generic">
                    <originalVal> 150 g </originalVal>
                </attribute>
            </foodLipid>
            <foodCalorie relType="completeRel"> ... </foodCalorie>
            <foodAmountLipid relType="partialNull"> ...</foodAmountLipid>
        </rowRel> ...
    </content>
</table>
```

FIG. 2.9 – *Représentation SML de la composition nutritionnelle des aliments*

2.1.6 Trace du contexte et des traitements

L'aspect automatique et générique de notre méthode d'enrichissement sémantique nous a conduit à une représentation flexible du résultat par le format SML. Dans un document SML, il est possible de représenter pour un même tableau plusieurs relations sémantiques qui sont soit complètement identifiées soit partiellement identifiées. Il est également possible de représenter des informations bien qu'elles soient non identifiées par des attributs supplémentaires ou par des relations génériques. Pour pouvoir exploiter la flexibilité du format SML, nous avons défini un ensemble d'indicateurs de traitement et de contexte. Ces derniers peuvent être vus comme

des méta-données[5] pouvant être exploitées soit pendant l'interrogation des documents SML soit par une procédure d'extraction d'informations candidates (e.g. des valeurs et des relations) à l'enrichissement de l'ontologie du domaine. Pour l'interrogation, en exploitant les informations véhiculées par ces indicateurs, un moteur d'interrogation peut s'adapter aux types d'informations représentées dans le document SML et ainsi proposer à un utilisateur d'autres réponses et des éléments de contexte en cas d'insatisfaction.

Nous présentons dans cette section les indicateurs de traitement qui sont représentés dans les documents SML. Ils sont représentés par des attributs XML ajoutés à différents niveaux du document : au niveau des lignes du tableau, au niveau d'une relation sémantique, au niveau d'un attribut de relation sémantique ou au niveau d'un terme de l'ontologie. Voici les cinq indicateurs de traitement que nous avons définis :

Indicateur additionalAttr. Il s'agit d'un indicateur qui représente des informations sur la structure des relations sémantiques. Il indique la présence ou l'absence d'attributs supplémentaires représentant les données des colonnes du tableau pour lesquelles aucun attribut de relation sémantique n'a été identifié. Il est ajouté aux éléments $< rowRel >$ du document SML. Par exemple, dans la table de la Figure 2.1, cet indicateur permet à un moteur d'interrogation d'utiliser l'attribut supplémentaire associé à la colonne *Item*.

Indicateur relType. Il s'agit d'un indicateur qui représente le type d'identification d'une relation sémantique : une relation complètement identifiée, une relation partiellement identifiée avec un attribut en constante ou avec un attribut *Null* ou encore une relation non identifiée. Il est ajouté à tout élément représentant une instance de cette relation dans le document SML. Il prend des valeurs dans l'ensemble : { *completeRel*, *partialNull*, *partialConst*, *generic*}.

Indicateur attrType. Il s'agit d'un indicateur qui représente le type d'identification d'un attribut de relation sémantique : un attribut complètement identifié, un attribut identifié par une constante dans le titre du tableau, un attribut non identifié dans le tableau ou encore un attribut associé à une colonne non identifiée. Il est ajouté aux éléments représentant les attributs des relations sémantiques dans le document SML. Il prend des valeurs dans l'ensemble : {$Normal$, $Null$, $Const$, $generic$}.

Indicateur attrMatch. Il s'agit d'un indicateur qui représente le A-Term associé à une colonne du tableau. Cet indicateur est intéressant car il arrive parfois qu'un attribut de relation sémantique soit identifié dans le tableau mais il n'est pas suffisant pour reconnaître la relation[6].

Indicateur indMap. Il s'agit d'un indicateur qui représente le type de la procédure d'appariement – *mapping*– utilisée pour trouver la mise en correspondance entre la valeur du tableau et les termes de l'ontologie. Il est ajouté aux éléments $< ontoVal >$, représentant dans le document SML un terme de l'ontologie trouvé comme similaire à la valeur d'origine d'une cellule du tableau. Plusieurs procédures d'appariement peuvent être utilisées dans la méthode d'enrichissement sémantique, e.g. des mesures de similarité syntaxique [CRF] ou des mesures de similarité sémantique [CHS04]. Ainsi, cet indicateur peut être utilisé pour évaluer l'impact de l'utilisation d'une procédure d'appariement sur la précision et la pertinence des termes associés aux valeurs

[5] Une méta-donnée est une donnée utilisée pour décrire ou apporter une précision sur une autre donnée.

[6] On rappelle que il est nécessaire d'identifier au minimum deux attributs pour identifier une relation.

du tableau. Par conséquent, il est possible de moduler le degré de confiance dans l'enrichissement relativement à la procédure d'appariement.

2.2 Algorithme d'enrichissement sémantique de tableaux

L'objectif de l'algorithme d'enrichissement sémantique que nous proposons est de mettre en correspondance les données représentées dans une structure de tableau avec le contenu de l'ontologie du domaine visé. Le résultat de cet algorithme que nous avons nommé $XTab2SML$ est un document SML dans lequel les lignes du tableau sont représentées par un ensemble de relations sémantiques identifiées dans le tableau. De plus, les valeurs des cellules du tableau sont représentées par le terme ou les termes de l'ontologie ayant été détectés comme étant similaires à ces valeurs.

Avant de décrire l'algorithme XTab2SML, nous présentons tout d'abord ses paramètres :

- ⋆ $Onto$, est un paramètre representant le contenu de l'ontologie que nous exploitons pour l'enrichissement sémantique. Nous considérons des parties de cette ontologie de la forme $\{taxonomie, schemaRel\}$: la taxonomie de termes et de concepts ainsi que le schéma relationnel contenant les signatures de différentes relations sémantiques.

- ⋆ $XTab$, est un paramètre représentant le document XTab que nous souhaitons enrichir. Nous considérons qu'un document XTab est sous forme d'un ensemble : $\{tabTitre, colonnesTitles, valeurs\}$ où $tabTitre$ correspond au titre du tableau, $colonnesTitles$ correspond aux titres des colonnes et $valeurs$ correspond aux valeurs des cellules organisées par colonne.

- ⋆ p, est une valeur réelle dans l'intervalle [0..1] représentant le pourcentage minimum des valeurs auxquelles des termes de l'ontologie sont associés dans chaque colonne.

- ⋆ th_t, est une valeur réelle dans l'intervalle [0..1] représentant le seuil des scores de similarité utilisé lors de l'identification des A-Terms dans les colonnes du tableau.

- ⋆ th_v, est une valeur réelle dans l'intervalle [0..1] représentant le seuil des scores de similarité utilisé lors de l'instantiation des relations.

Dans l'algorithme XTab2SML, pour comparer le contenu du tableau avec celui de l'ontologie, nous utilisons des mesures de similarité telles que la mesure de Jaccard [Jac01] (cf. Section 4.2 du chapitre 4) ainsi que la procédure d'appariement inspirée de du système PANKOW. Nous référons à la mesure qui permet de calculer un score de similarité, dans l'intervalle [0..1], entre deux chaînes de caractères s et s' par $similarite(s, s')$

Nous utilisons deux seuils de similarité : (i) un seuil élevé th_t pour trouver des mises en correspondance entre les valeurs du tableau et les titres des colonnes d'une part et le contenu de l'ontologie d'autre part ; (ii) un seuil moins élevé th_v qui est utilisé lors de l'instantiation des relations sémantiques où l'on cherche à associer à chaque valeur du tableau un ou plusieurs termes de l'ontologie.

Algorithm 1 `XTab2SML : ENRICHISSEMENT SÉMANTIQUE DE TABLEAUX`

Onto : l'ontologie du domaine d'application.
XTab : le tableau XTab à enrichir.
Inputs : p : le pourcentage minimum des valeurs à identifier par des termes.
th_t : un seuil à utiliser pendant l'identification des A-Terms.
th_v : un seuil à utiliser pendant l'instantiation des relations.

Output : *docSml* : un document XML conforme à la DTD SML

1: $docSml \leftarrow \emptyset$;
2: $tabSch \leftarrow ConstructionSchemaTab(XTab, Onto, th_t, p)$;
3: $Rels \leftarrow IdentificationRelations(tabSch, Onto.schemaRel)$;
//Cas où aucune relation n'a été identifiée, une relation générique est alors créée
4: **If** $(Rels = \emptyset)$ **Then**
5: $relGen \leftarrow CreationElementXML(''relation'')$;
6: $R \leftarrow InstantiationRelationDansTableau(relGen, XTab.Vals, TabSch, th_v)$;
7: $docSml \leftarrow AjoutRelation(R, docSml)$;
8: **Else If** $(Rels \neq \emptyset)$ **Then**
9: **Foreach** $(R_i \in Rels)$ **Do**
10: $R \leftarrow InstantiationRelationDansTableau(R_i, XTab.valeur, tabSch, Onto.taxonomie, th_v)$;
11: $docSml \leftarrow AjoutRelation(R, docSml)$;
12: **End For**
13: **End If**
14: **Return** *docSml* ;

Construction du schéma du tableau

Cette tâche est effectuée par la primitive $constructionSchemaTab(XTab, Onto, th, p)$. Il s'agit pour un tableau XTab donné de trouver pour chaque colonne un A-Term représenté dans l'ontologie. Si aucun A-Term n'a été trouvé alors un attribut générique lui est associé cf. instruction 19 de l'algorithme 2. Cette primitive est décrite dans l'algorithme 2 ci-dessous.

Identification des relations.

Cette tâche est effectuée par la primitive $identificationRelationTabSch(tabSch, Onto.schemaRel)$ cf. instruction 3 de l'algorithme 1. Il s'agit de mettre en correspondance les signatures des relations du schéma relationnel avec le schéma du tableau construit par l'algorithme 2 afin de pouvoir identifer les relations sémantiques qui sont complètement ou partiellement représentées dans le schéma du tableau. La méthode $InTab(R, TabSch)$ permet de vérifier si au moins deux attributs de la relation R ont été associés aux colonnes du tableau. La méthode $RechercheConstantesAttrRelation(R, XTab.tabTitle)$ cherche dans le titre du tableau les éventuelles constantes qui soient similaires aux attributs de la relation R non identifiés dans le schéma du tableau.

Algorithm 2 CONSTRUCTION DU SCHEMA DU TABLEAU

Inputs : $XTab$: le tableau XTab
$Onto$: l'ontologie du domaine d'application
p : le pourcentage minimum des valeurs à identifier par des termes.
th_t : un seuil à utiliser pendant l'identification des A-Terms.

Output : $tabSch$: le schéma du tableau sous la forme $\{(C_1, ATerm_1),\ldots, (C_n, ATerm_n)\}$

1: $ATS \leftarrow ExtractionATermsOntologie(Onto.schemaRel)$;
2: **Foreach** $(C_i \in XTab.valeurs)$ **Do**
3: $j \leftarrow 1$;
4: $ATermOk \leftarrow$ Faux ;
5: $ATerm \leftarrow$ "" ;
6: **While** $(ATermOk =$ Faux et $j < ATS.taille)$ **Do**
7: $TS(C_i) \leftarrow CalculEnsembleATermsCorrespondants(ATS[j], th_t)$;
8: **If** $((ColonneSatisfaitPourcentage(C_i, TS(C_i), p)))$ **Then**
9: $ATermOk \leftarrow$ Vrai ;
10: **Else If** $(similarite(XTab.colonnesTitres[i], ATS[j]) > th_t)$ **Then**
11: $ATerm \leftarrow$ ATS[j] ;
12: **End If**
13: $j \leftarrow j + 1$;
14: **End While**
15: **If** $(ATermOk = Vrai)$ **Then**
16: $tabSch[C_i] \leftarrow ATerm$;
17: **Else**
18: $tabSch[C_i] \leftarrow'' attribute''$;
19: **End If**
20: **End For**
21: **Return** $tabSch$;

Instantiation des relations dans le tableau

Cette tâche est effectuée par la primitive $instantiationRelationDansTableau$ ($relGen$, $XTab.Vals$, $TabSch$) cf. instruction 10 de l'algorithme 1. Il s'agit, pour une relation identifiée complètement ou partiellement dans le tableau, de rechercher pour toute valeur des colonnes qu'elle représente, le terme ou les termes de l'ontologie qui lui sont proches sémantiquement. Cette proximité est évaluée par une procédure d'appariement telle que les mesures de similarité syntaxique. Si aucun terme n'a été trouvé, nous gardons alors uniquement la valeur d'origine du tableau. Cette primitive est décrite dans l'algorithme 4 présenté ci-dessous.

Algorithm 3 IDENTIFICATION RELATION SCHEMA TABLEAU

Inputs : *tabSch* : le schéma du tableau
Onto.schemaRel : le schéma relationnel de l'ontologie

Output : *Rels* : la liste des relations identifiées dans le schéma.

Foreach ((R $\in$ *Onto.schemaRel*) **Do**
  If (InTab(R, TabSch)) **Then**
    AjoutRelation(R, *Rels*) ;
    If ($RelationCompletementIdentifiee(R)$) **Then**
      AjoutRelation(R, *Rels*) ;
    Else
      $const \leftarrow RechercheConstantesAttrRelation(R, XTab.tabTitle)$;
      If ($const \neq \emptyset$) **Then**
        $R' \leftarrow ConstructionRelationPartielleAttributsConstants(R, const)$;
        AjoutRelation(R', *Rels*) ;
      Else
        $R' \leftarrow ConstructionRelationPartielleAttributsNull(R)$;
        AjoutRelation(R', *Rels*) ;
      End If
    End If
  End If
End For
Return *Rels* ;

Algorithm 4 INSTANTIATION D'UNE RELATION DANS LE TABLEAU

Inputs : R_i : une relation à instancier
XTab.valeurs : l'ensemble des valeurs du tableau (la matrice (lignes, colonnes))
TabSch : le schéma du tableau pré-calculé
th_v : un seuil à utiliser pendant l'instantiation des relations.

Output : *relInst* : relation instanciée par les valeurs du tableau.

For (i=0 ; i < R_i.nbAttrs ; i++) **Do**
  valCol $\leftarrow$ valeurs(TabSch[R_i])
  For (j=0 ; j < valCol.nbVals ; j++) **Do**
    ontoVal $\leftarrow \emptyset$;
    origine$\leftarrow$valCol[j] ;
    ensTerme $\leftarrow$ RechercheEnsembleTermes(valCol[j],R_i, Onto.taxonomie, th_v) ;
    If (ensTerme = $\emptyset$) **Then**
      valsRelInst $\leftarrow$ *valsRelInst*$\bigcup$ {originalVal, NULL} ;
    End If
  End For
  relInst $\leftarrow$ *relInst* $\bigcup$ {valsRelInst} ;
End For
Return *relInst* ;

2.3 Génération automatique de la DTD SML à partir de l'ontologie

La généricité du formalisme SML est garantie par la possibilité de générer de manière automatique la DTD SML à partir d'une ontologie de domaine contenant une taxonomie de termes et de concepts et un schéma relationnel.

Dans la Figure 2.10 nous présentons un extrait de schéma relationnel d'une ontologie du risque alimentaire. En Figure 2.11 la représentation de la DTD SML générée automatiquement pour représenter les documents SML traitant de ce domaine d'application. La DTD est représentée ici par un graphe orienté où les nœuds représentent les éléments de la DTD et les arcs représentent la relation de composition entre les éléments du document SML. Dans cet exemple, nous n'avons pas représenté les attributs XML jouant le rôle d'indicateurs de traitement et de contexte.

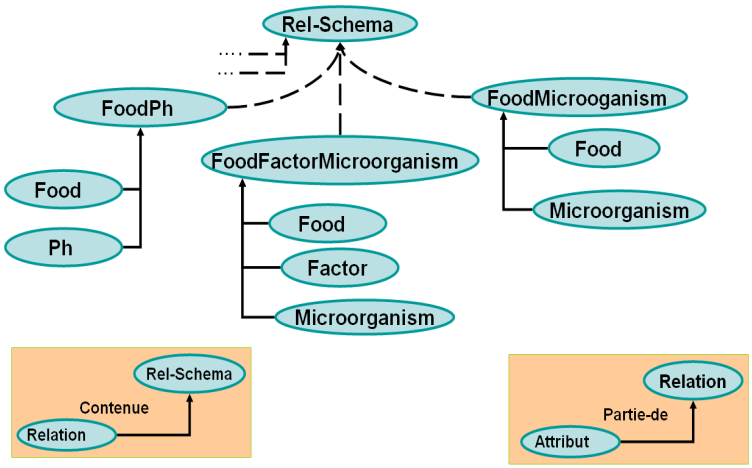

FIG. 2.10 – Extrait du schéma relationnel de l'ontologie du risque alimentaire (Sym'Previus)

Dans la DTD SML, nous distinguons trois types d'éléments :

⋆ *Structurels*, il s'agit des éléments représentant des informations structurelles du tableau d'origine conservées dans le document SML, comme par exemple les éléments *table*, *title*, *content* et *rowRel*. Il s'agit également des éléments que nous avons définis pour distinguer entre les valeurs d'origine du tableau (le tag $originalVal$) et les termes de l'ontologie qui leur sont associés (le tag $ontoVal$).

⋆ *Sémantiques*, il s'agit des éléments représentant des informations sémantiques, contenues dans l'ontologie et que la méthode d'enrichissement sémantique a permis d'identifier dans le contenu des tableaux, comme, par exemple, le nom d'une relation et la liste de ces attributs.

⋆ *Génériques*, il s'agit des éléments que nous avons définis et que nous ajoutons au document SML pour représenter les informations non identifiées telles que les attributs supplémentaires (tag *attribute*) et la relation générique (tag *relation*).

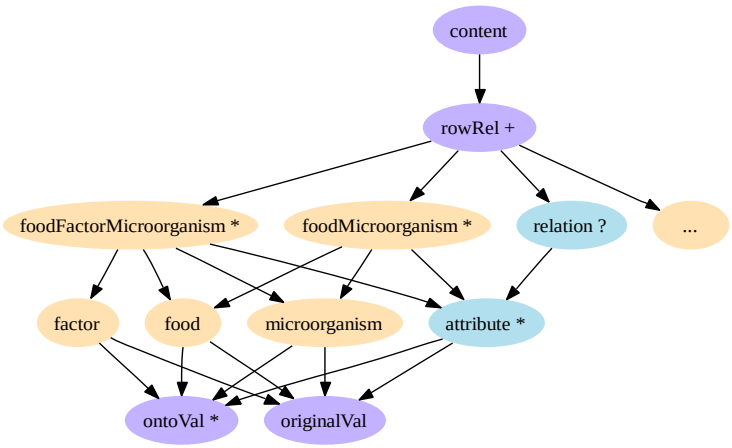

FIG. 2.11 – Extrait de la DTD SML (risque alimentaire)

La génération automatique de la DTD SML peut être effectuée par l'application d'un ensemble de règles de transformation appliquées sur le schéma relationnel de l'ontologie. Dans le cas où l'ontologie est représentée dans un langage appartenant à la galaxie XML (e.g. XML, RDFS, OWL), les règles de transformation peuvent être exprimées dans le language XSLT (*XML Style Language Transform*) dédié aux règles de transformation appliquées sur les documents de format de type XML.

2.4 Découverte de relations candidates à l'alimentation d'un entrepôt thématique

Le caractère automatique de notre approche d'enrichissement sémantique nous a conduit à utiliser un format flexible permettant de représenter les données (termes ou relations) dont l'identification est partielle, incertaine ou multiple.

Il est en effet possible de reconnaître des relations même si tous leurs attributs ne sont pas identifiés dans le tableau ; tel est le cas si, par exemple, l'un des attributs est une constante présente dans le texte qui environne le tableau. Il s'agit alors de relations dites partielles. Il nous semble utile de pouvoir enrichir l'entrepôt avec des instances de relations existantes complétées d'informations imparfaitement identifiées ou même de garder, en cas d'échec lors de l'identification des relations représentées dans le tableau, une relation dite générique qui permettra de conserver l'existence d'un lien (même s'il n'est pas identifié) entre certaines données. Ces relations sont stockées et peuvent être exploitées lors de l'interrogation. Elles peuvent également servir de base à un enrichissement de l'ontologie si un expert les identifie par la suite.

Il s'agit maintenant de savoir ce que le système doit faire des informations contenues dans les tableaux et qu'il n'a pas pu représenter dans une instance de relation de l'ontologie. Nous distinguons deux cas : celui où nous utilisons les informations non reconnues pour compléter une relation identifiée par des attributs supplémentaires et celui où aucune relation n'a pu être identifiée dans le tableau.

2.4.1 Capture de l'information non reconnue dans des attributs supplémentaires

A l'étape de l'identification des colonnes du tableau, il arrive souvent que certaines colonnes ne soient associées à aucun attribut ou qu'elles soient identifiées mais ne soient associées à aucune relation reconnue dans le tableau. Nous considérons alors que ces attributs sont des informations supplémentaires qui peuvent venir préciser ou modifier l'interprétation d'une relation reconnue. Les informations de ces colonnes sont alors représentées dans le document SML par des attributs supplémentaires à l'aide du tag générique $< attribute >$. Ce dernier est ajouté à toute relation reconnue dans le tableau. Ces informations additionnelles peuvent jouer plusieurs rôles :

⋆ recueillir un attribut non identifié dans une relation partielle (ex : attribut numérique) ;
⋆ recueillir un argument supplémentaire de la relation qui n'est pas défini dans l'ontologie (ex : forme ou quantité du produit) afin de préciser l'interprétation de la relation ;
⋆ recueillir une relation supplémentaire qui partage des attributs avec la relation reconnue (ex : le tableau comportant les colonnes *food*, *lipid*, *ph*, *microorganism*).

Le tableau de la figure 2.12 représente les résultats d'une étude de l'influence de trois facteurs expérimentaux sur la croissance de la *listeria* dans les *aliments fumés*. Dans cet exemple de tableau, le document SML correspondant comporte deux attributs supplémentaires permettant de représenter deux attributs *Factor* apparaissant dans les 4ème et 5ème colonnes. Le système n'a en effet pas reconnu que les colonnes ayant pour titre "*growth rate*" et "*egrc at 5◦c (log10 cfu/day)*" représentaient des attributs *Factor*. Ces informations sont alors conservées dans le document SML et représentées par des attributs supplémentaires.

microorganism	**food**	**temp.**	**growth rate**	**egrc at 5◦c (log10 cfu/day)**
listeria	coppa (smoked pork)	4◦c	2.1 logs in 28 days	0.107
listeria	cold-smoked salmon	8◦c	5.4 logs in 21 days	0.116

FIG. 2.12 – *Microorganismes et aliments fumés*

Dans ce second exemple de tableau de la Figure 2.13, l'attribut supplémentaire, que l'on peut voir dans le document SML de la figure 2.14, permet de compléter la relation reconnue par une information qui n'a pas été prévue dans l'ontologie. Ainsi, dans le domaine du risque alimentaire, on trouve souvent des tableaux qui précisent des références d'articles pour chaque résultat présenté dans les lignes du tableau. Ce paramètre n'a pas été prévu dans la signature de la relation mais il est recueilli dans un attribut supplémentaire et peut être proposé par la suite à l'utilisateur mais également pour un enrichissement de l'ontologie.

microbe	**vehicle/source**	**reference**
Campylobacter spp.	poultry	Friedman et al 2000 [1]
Clostridium perfringens	meat	Todd 1997 ; Olsen, S. J. et al. 2000 [1]

FIG. 2.13 – *Aliments et Microorganismes*

Nous allons voir en Section 2.5.2 comment, lors de l'interrogation, ces attributs supplémentaires permettent de préciser, de modifier ou de compléter l'interprétation des réponses aux

```
<table>
   <title>
       </table-title >
            <column-title> microbe </column-title>
            <column-title>vehicle/source</column-title>
            <column-title>reference</column-title>
   </title>
   <column-nb> 3 </column-nb>
   <content>
       <rowRel additionalAttr="yes">
        ...
            <foodMicroorganism relType ="completeRel">
                <food> <ontoVal>... </ontoVal> </food>
                <microorganism>
                    <ontoVal>campylobacter</ontoVal>
                    <originalVal>campylobacter spp. </originalVal>
                </microorganism>
                <attribute attrMatch="notFound">
                    <originalVal> Friedman et al 2000 </originalVal>
                </attribute>
            <foodMicroorganism>
            ...
       </rowRel>
       ...
   </content>
</table>
```

FIG. 2.14 – *Représentation SML du tableau de la fig.2.13 - Attribut supplémentaire*

requêtes de l'utilisateur.

2.4.2 Représentation de tableaux par une relation générique

Nous allons maintenant montrer le rôle des relations génériques dans la représentation des informations des tableaux. Une relation générique permet de représenter les données d'un tableau dans lequel aucune relation de l'ontologie n'a été reconnue. L'échec de la reconnaissance des relations sémantiques est dû à plusieurs facteurs :

⋆ les relations sémantiques représentées dans le tableau ne sont pas décrites dans l'ontologie du domaine ;

⋆ des tableaux peuvent représenter des informations concernant deux thématiques à la fois. Par exemple, certains tableaux traitant de l'épidémiologie représentent des informations concernant aussi le risque alimentaire. Il s'agit en fait d'une thématique connexe ;

⋆ des attributs de relations sémantiques non reconnus dans le tableau empêchent la reconnaissance des relations. C'est en particulier le cas pour des attributs qui représentent des valeurs numériques et pour lesquels le processus de reconnaissance se fonde entièrement sur le titre des colonnes. Dans ce cas, la relation générique permet de représenter les attributs reconnus et non reconnus.

Les relations génériques permettent de conserver ces informations. La Figure 2.15 représente, par exemple, la dose à laquelle on peut dire qu'un germe a infecté un support. Il s'agit d'une relation $factorMicroorganism$ mais la colonne numérique qui a pour titre *infective dose* et qui correspond au facteur n'a pas pu être reconnue. Ce tableau a pour représentation SML

le document présenté en Figure 2.16, document dans lequel les valeurs apparaissant dans les colonnes sont conservées et liées par une relation générique.

Pathogen	Infective dose	reference
Campylobacter spp.	500-800	Robinson 1981 ; Black et al 1988
Clostridium perfringens	10^7	Bermith 1988

FIG. 2.15 – *Doses infectieuses*

```
<table>
    <title>
       </table-title >
            <column-title> pathogen </column-title>
            <column-title>infective dose</column-title>
            <column-title>reference</column-title>
    </title>
    <column-nb> 3 </column-nb>
  <content>
   <rowRel additionalAttr="yes">
    ...
   <relation relType ="generic">
        <attribute attrMatch="microorganism">
            <ontoVal>campylobacter</ontoVal>
            <originalVal>campylobacter spp. </originalVal>
        </attribute>
        <attribute attrMatch="notFound">
            <originalVal>500-800</originalVal>
        </attribute>
        <attribute attrMatch="notFound">
            <originalVal> Robinson 1981; Black et al 1988 </originalVal>
        </attribute>
   </relation>
    ...
   </rowRel> ...
 </content>
</table>
```

FIG. 2.16 – *Représentation SML du tableau de la fig.2.15 - Relation générique*

2.5 Interrogation des documents SML

L'interrogation des documents SML est effectuée en utilisant des requêtes exprimées dans le langage XQuery (*XML Query*). Ces requêtes sont formulées en utilisant le vocabulaire de l'ontologie du domaine par laquelle l'enrichissement des tableaux est guidé.

Dans le cadre du projet e.dot, le moteur d'interrogation MIEL++ est capable d'interroger les deux bases existantes, relationnelle et graphe conceptuel, en combinaison avec l'interrogation de la base SML. Il est donc en mesure de traduire une requête MIEL en une requête sur la base des documents SML.

Le vocabulaire, utilisé pour exprimer les requêtes et les relations sémantiques à interroger, est représenté dans l'ontologie Sym'Previus. Ainsi, l'interrogation de l'entrepôt SML et des bases pré-existantes se fait de manière uniforme et transparente pour l'utilisateur par le biais d'une même interface graphique. Une extension de ce moteur d'interrogation utilisant XQuery peut

permettre d'interroger les données supplémentaires non identifiées ou imparfaitement identifiées qui apparaissent en tant qu'attributs supplémentaires et relations génériques.

2.5.1 Exemple d'interrogation

L'interrogation des documents SML est effectuée par des requêtes XQuery qui sont exprimées en utilisant les termes et les relations déclarées dans l'ontologie du domaine mais aussi en utilisant les éléments structurels déclarés dans la DTD SML tels que les tags *rowRel*, *ontoVal*, etc.

Nous allons maintenant décrire un exemple de requête où l'utilisateur cherche à obtenir la quantité de lipides contenus dans 100 g de crabe. Cette requête est exécutée sur le document présenté en Figure 2.9 et le résultat de cette requête est donné en Figure 2.17. L'évaluation de cette requête consiste à chercher dans le document SML des sous-arbres – fragments SML – tels que le la racine est un élément *foodLipid* et tels que il contient un élément *ontoVal* qui a comme valeur "crabe". Comme l'indicateur *additionalAttr* a comme valeur "yes", le moteur de requêtes affiche à l'utilisateur l'information additionnelle *150g* qui est représentée par l'attribut supplémentaire ce qui améliore la précision de la réponse à la requête de l'utilisateur. De plus, la valeur d'origine *ground crab* est également montrée à l'utilisateur.

```
<table>
    <table-title>Nutritional composition of some food products  </table-title >
    <food> ground crab</food>
    <lipid>11.25 g</lipid>
    <additionalattr>150 g</additionalattr>
    <category> unknown</category>
</table>
```

FIG. 2.17 – *Une présentation possible de la réponse*

2.5.2 Exploitation des attributs supplémentaires

Les attributs supplémentaires peuvent être exploités pour toute requête dont les réponses proviennent de relations avec attributs supplémentaires. Pour permettre à l'utilisateur d'avoir plus d'informations concernant le contexte des réponses à sa requête, le contenu et les titres des colonnes correspondant aux attributs supplémentaires peuvent être affichés. Voici un exemple de requête Q1 où l'utilisateur cherche les aliments pouvant véhiculer le microorganisme "*Campylobacter spp.*". La Figure 2.18 représente le résultat de l'évaluation de la requête Q1 sur l'entrepôt de données SML contenant des documents représentant des informations sur "*Campylobacter spp.*". Parmi ces documents se trouve le document SML de la figure 2.14.

Microorganisme	**Aliment**	**Attributs Supp.**
campylobacter	poultry	Reference : Friedman et al 2000
campylobacter	turkey	

FIG. 2.18 – *La réponse à la requête Q1*

Cet exemple montre l'intérêt de présenter le contenu des attributs supplémentaires. Ici, la complétion de la réponse par la référence bibliographique intéressera vraisemblablement l'utilisateur. La présentation du titre de la colonne *Reference* devant la valeur de cet attribut facilitera l'interprétation de cette nouvelle information.

2.5.3 Exploitation des relations génériques

L'exploitation des relations génériques par l'utilisateur consiste entre autres en l'interrogation de la base de données SML par des requêtes par mots clés, comme la requête Q2 qui cherche toute l'information au sujet du microorganisme *"Campylobacter"*. Son évaluation contiendra toutes les relations identifiées et toutes les relations génériques dont une des valeurs du tag *ontoVal* est "*Campylobacter*" ou une spécialisation de "*Campylobacter*".

Attribut1	Attribut2	Attribut3
Pathogen : campylobacter	Infective dose : 500-800	reference : Robinson 1981 ; Black et al 1988
Organism : campylobacter	Temp : 5∘C	

FIG. 2.19 – *Réponse à la requête Q2*

Dans la figure 2.19 est présenté un extrait de la réponse à la requête Q2 qui correspond à une ligne de la relation générique représentant le tableau de la figure 2.15. Si nous n'avions pas interrogé les relations génériques, nous aurions raté cette réponse et nous n'aurions peut-être même obtenu aucune réponse dans l'hypothèse où aucune relation de l'ontologie ne contient la valeur *"campylobacter"*.

2.5.4 Interrogation par les vues pré-définies dans MIEL++

Dans le projet e.dot, le moteur d'interrogation flexible MIEL++ joue le rôle d'un médiateur entre trois sources de données : la base relationnelle, la base de graphes conceptuels et l'entrepôt de données SML.

Les documents SML sont physiquement stockés dans un entrepôt de données XML. Chaque document SML représente un seul tableau enrichi sémantiquement. L'ensemble des documents SML sont stockés dans trois groupes physiquement séparés. Chaque groupe de documents correspond à une vue logicielle du système MIEL++. Trois vues principales ont été prédéfinies dans le système et il s'agit de : la vue *effet/facteur*, la vue *charge microbienne* et la vue *divers*.

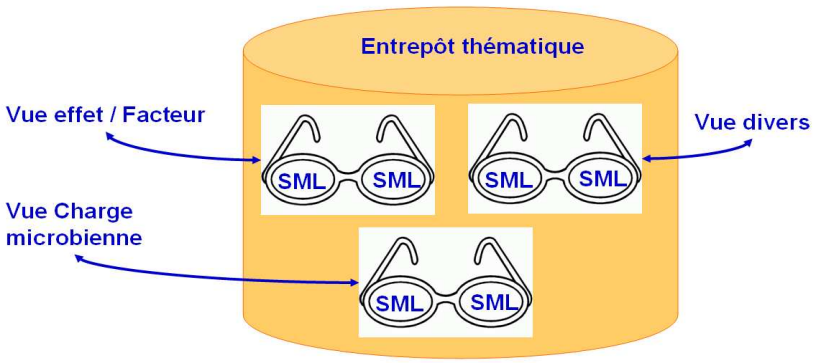

FIG. 2.20 – Une représentation des trois vues de MIEL++

Les différentes vues sur l'entrepôt de données SML

Une vue est caractérisée par l'ensemble des attributs (de sélection ou de projection) auxquels l'utilisateur a accès pour effectuer ses requêtes et voir les résultats. La vue correspond physiquement à la requête XML instanciée par la valeur de ses attributs.

Actuellement dans MIEL++, deux vues sont accessibles à l'utilisateur pour interroger les différentes bases de données : la vue "*aliment/effet du facteur analysé*" et la vue "*charge microbienne*". La première permet d'étudier l'évolution d'un microorganisme dans un aliment sous l'effet d'un ou plusieurs facteurs expérimentaux. La seconde vue représente la contamination d'un aliment par un microorganisme soit par un niveau (ufc/g), soit par un taux (comptage de fréquence de contamination) ou une distribution de fréquences.

Vue Aliment/effet du facteur analysé. L'interrogation des documents SML par cette vue est effectuée à travers une des relations du schéma relationnel dont la signature contient au moins un attribut de type *facteur*. Il s'agit par exemple de :

- ⋆ foodPh (food, ph, attribute*)
- ⋆ foodAw (food, aw, attribute*)
- ⋆ foodState (food, state, attribute*)
- ⋆ foodStructure (food, structure,attribute*)
- ⋆ foodTemperatureMicroorganism (food, Microorganism, temperatureValue, attribute*)
- ⋆ ... autant de relations que de facteurs possibles de sélection contenus dans l'ontologie.

Un document SML doit être stocké dans cette vue s'il possède une relation sémantique de type ***"foodFactorNameMicroorganism"*** où *FactorName* peut prendre comme valeur tous les termes subsumés par le terme *Factor* dans l'ontologie.

Une requête *XQuery* a été définie pour l'interrogation des relations *foodPh* et *foodTemperatureMicroorganism*. La spécification des valeurs des attributs est la suivante : {food="cabbage', ph=5, factor=temperature}.

La requête XQuery est donnée en *Annexe E.1*

Vue charge microbienne. L'interrogation des documents SML, par cette vue, est effectuée à travers les cinq relations sémantiques décrites dans le schéma relationnel de l'ontologie suivantes :

- ⋆ foodPh (food, ph, attribute*)
- ⋆ foodAw (food, aw, attribute*)
- ⋆ foodState (food, state, attribute*)
- ⋆ foodStructure (food, structure,attribute*)
- ⋆ foodMicroorganismConcentration(food, Microorganism, concentration, attribute*)

Un document SML doit être stocké dans cette vue s'il possède la relation sémantique intitulée ***"foodMicroorganismConcentration"***.

Une requête *XQuery* a été définie pour interroger les relations *foodPh* et *foodMicroorganismConcentration*. La spécification des valeurs des attributs est la suivante : {food ="cabbage", ph =5, microorganism="Listeria M."}.

Les indicateurs de traitement *additionnalAttr* et *attrMatch* sont pris en compte pour interroger d'autres relations qui auraient des attributs supplémentaires (*additionnalAttr =yes*) dont la catégorie est *aliment* (*attrMatch=food*) et *microorganism* (*attrMatch=microorganism*).

La requête est donnée en *Annexe E.2.*

Vue divers. Dans un second temps, une troisième vue plus flexible pour l'utilisateur a été définie. Dans cette vue l'utilisateur pourra effectuer une interrogation sur l'ensemble des attributs qui ne sont pas présents dans une des deux autres vues, par exemple, *lipid*, *calorie*.

Tous les documents SML ne répondant pas aux critères des deux autres vues sont stockés dans cette vue *divers*.

2.6 Conclusion

La méthode que nous avons présentée dans ce chapitre permet d'enrichir sémantiquement des documents collectés à partir du Web ou fournis par des fournisseurs de contenus. Ces documents ont la spécificité de contenir des données présentées dans une structure tabulaire. La tâche d'enrichissement sémantique est complètement automatique et guidée par une ontologie du domaine. Par conséquent, nous ne pouvons pas garantir l'exactitude et la complétude du résultat de l'enrichissement sémantique. La représentation XML que nous proposons conserve toutes les interprétations possibles afin de permettre à l'utilisateur de les exploiter à l'étape d'interrogation. Il sera, par exemple, possible d'effectuer une requête utilisant des attributs ou des relations génériques et d'afficher des informations complémentaires pour aider l'utilisateur à mieux interpréter les réponses à ses requêtes. Ces informations complémentaires peuvent être choisies parmi, le titre de colonnes, les attributs supplémentaires ou les valeurs d'origine du tableau. De plus, dans le cas d'une ambiguïté, il est possible d'associer plusieurs termes de l'ontologie et plusieurs relations sémantiques au même ensemble de colonnes. Afin de permettre au moteur d'interrogation d'adapter les réponses aux requêtes ou d'évaluer leur pertinence, nous gardons trace des traitements effectués lors de l'enrichissement sémantique en utilisant un ensemble d'indicateurs. Nous avons également présenté comment les informations imparfaitement identifiées peuvent être exploitées pour découvrir des relations candidates à l'alimentation d'un entrepôt et à l'enrichissement d'une ontologie. La généricité de notre approche est garantie par le fait que la DTD SML peut être générée automatiquement à partir de l'ontologie.

L'algorithme $XTab2SML$ que nous avons présenté dans ce chapitre a été implémenté dans un service Web dans le projet e.dot. Nous avons réalisé des expérimentations, sur des documents traitant du domaine du risque alimentaire, pour la méthode d'enrichissement sémantique ainsi que pour la découverte de relations candidates à l'alimentation de l'entrepôt de données. Le protocole d'expérimentation et les résultats obtenus sont présentés dans le chapitre 3.

Chapitre 3

Développement et expérimentations

Sommaire

3.1 Développement 57
3.2 Protocole d'évaluation des résultats de la méthode XTab2SML 59
3.2.1 Chaîne de traitements des documents évalués 59
3.2.2 Méthodologie d'évaluation 59
3.3 Présentation des résultats 61
3.3.1 Résultats de l'enrichissement sémantique 62
3.3.2 Résultats de la découverte de relations candidates 63
3.4 Conclusion 65

Nous présentons dans ce chapitre les expérimentations que nous avons effectuées pour évaluer notre méthode d'enrichissement sémantique XTab2SML. L'expérimentation a été effectuée sur des données réelles représentées dans des documents XTab qui sont extraits à partir de documents collectés à partir du Web dans le cadre du projet e.dot. Le domaine d'application est le risque microbiologique dans les aliments représenté dans l'ontologie Sym'Previus.

En plus de l'évaluation de la qualité des résultats de notre méthode XTab2SML, nous avons également analysé l'ensemble de documents produits afin d'évaluer la découverte de relations candidates.

Ce chapitre est organisé de la manière suivante : en Section 3.1, nous décrivons succinctement le prototype et le service Web que nous avons implémentés pour la méthode XTab2SML. En Section 3.2, nous présentons le protocole que nous avons suivi pour l'évaluation des résultats. Ensuite, en Section 3.3, nous donnons les résultats que nous avons obtenus pour la méthode d'enrichissement sémantique puis ceux concernant la découverte de relations candidates. Enfin, nous concluons ce chapitre en Section 3.4.

3.1 Développement

Comme le schéma de la Figure 3.1 le montre, l'application XTab2SML prend en entrée, d'une part l'ontologie du domaine en format XML, d'autre part le document XTab à transformer. Elle

renvoie un document SML résultant de l'enrichissement du document XTab par la sémantique du domaine représentée dans l'ontologie.

Pour mettre en œuvre l'algorithme d'enrichissement sémantique que nous avons proposé dans 2, nous avons implémenté un prototype en utilisant le langage *Java*. Ce dernier s'est révélé particulièrement adapté à cette réalisation. Un des principaux atouts du langage Java est sa portabilité qui assure l'exécution de tout programme développé en Java sur n'importe quelle plate-forme (*MS-Windows, UNIX, MAC-OS,* etc). De plus, de nombreuses APIs (*Application Programming Interface*) sont disponibles dans le langage *Java* pour traiter des documents XML, mais également pour concevoir des services Web.

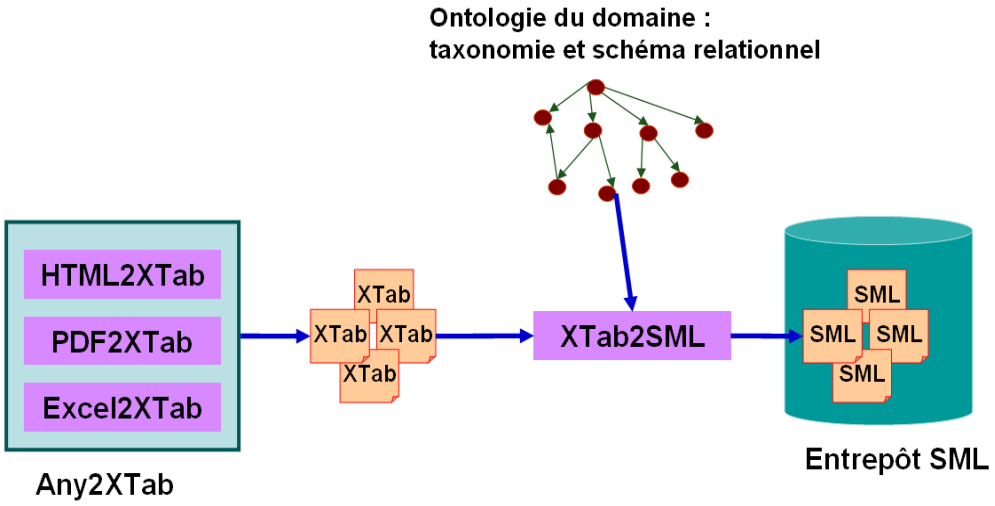

FIG. 3.1 – Schéma représentant les entrées et la sortie du service Web XTab2SML

Le modèle DOM[7] (*Document Object Model*) a été utilisé pour avoir un accès plus aisé aux documents XML via un ensemble d'objets classés en arborescence. Cette arborescence est une représentation équivalente au fichier XML, mais sa manipulation via le langage *Java* en est grandement simplifiée. Ainsi les fichiers XML représentant l'ontologie et le tableau XTab sont convertis en arbre DOM, en utilisant l'API de la librairie JAXP (*Java API for XML Parsing*). Cette dernière nous a également permis de générer le document SML résultat. Après avoir représenté l'ontologie et le document XTab sous une forme d'arbres DOM, qui sont ensuite compilés dans une représentation interne sous la forme d'une *"Table de Hachage"*. Ce choix est fait pour optimiser la recherche des données à manipuler car l'utilisation de la structure de données *"Table de Hachage"* permet d'avoir un accès direct aux données alors qu'un accès dans une représentation arborescente nécessite un parcours complet de l'arbre, dans le pire des cas.

Nous avons publié la méthode principale de notre application qui est la méthode nommée *launchXTabToSML* qui prend en entrée un document XTab et l'ontologie du domaine et qui renvoie le document SML. Nous donnons en Annexe F, la description du service Web sous la forme d'un document WSDL [8] [WSD01] (*Web Service Description Language*)

[7] La spécification de DOM [DOM] est gérée par le consortium *W3C (World Wide Web Consortium)*

[8] Une recommandation du W3C pour la description des services Web.

3.2 Protocole d'évaluation des résultats de la méthode XTab2SML

Parmi deux cents documents XTab extraits à partir de documents réels pertinents pour le domaine d'application et collectés à partir du Web, nous avons manuellement sélectionné 70 documents XTab. Un tableau XTab est retenu pour l'expérimentation si et seulement si, au moins pour une de ses valeurs on peut trouver un terme de l'ontologie qui lui est similaire, par exemple, un tableau contenant la valeur "*Orange jelly*" et pour laquelle on trouve dans l'ontologie le terme "*Orange*", est sélectionné

Dans les deux sections suivantes, nous décrivons brièvement l'outil AQWEB développé dans l'équipe MIA/INRA partenaire du projet e.dot. Puis nous présentons la méthodologie d'évaluation qui a été utilisée.

3.2.1 Chaîne de traitements des documents évalués

L'application AQWEB développée, dans le cadre du projet e.dot par l'équipe MIA/INRA, dans le but de faciliter la tâche de validation par un expert des différents modules développés dans le projet. Cette application permet de relier les différentes tâches intervenant dans la chaîne de traitements des documents, de leur collecte jusqu'à leur enrichissement sémantique.

Le service permettant de rechercher et de collecter des documents PDF [9] potentiels à partir du Web considère en entrée un tuple (nom d'un aliment, nom d'un microorganisme, expressions exactes, mots exclus, langue d'un document, table, références) et récupère des documents vérifiant la requête exprimée dans ce tuple. Les fichiers PDF sont convertis automatiquement en format Word avec l'outil Omnipage [omn05]. Cette tâche est asynchrone. Les fichiers convertis en Word doivent être validés car il arrive parfois que les documents Word générés contiennent des erreurs, notamment dans les polices de caractères. La présentation des formats de tableaux peut être aussi simplifiée pour se ramener à un format de type XTAB (une seule ligne d'entête, sans groupement de cellules et plusieurs lignes de détail).

L'ensemble des tableaux de données se trouvant dans un fichier Word validé est converti automatiquement en un ensemble de documents XML au format XTAB. Ces documents XTab sont enfin enrichis sémantiquement par le module XTab2SML que nous avons développé.

3.2.2 Méthodologie d'évaluation

Dans l'évaluation de notre méthode XTab2SML nous montrons la capacité de notre méthode à identifier les relations sémantiques décrites dans l'ontologie du domaine, dans les tableaux XTab. Notre objectif est de comparer les résultats concernant l'identification des relations obtenus par notre méthode automatique avec ceux obtenus manuellement par un expert du domaine.

Afin de mesurer la qualité de nos résultats par rapport aux résultats attendus, nous calculons le rappel, la précision et la F-Mesure, mesures très utilisées dans le domaine de la recherche d'informations (IR). Pour calculer les valeurs de ces mesures nous avons besoin de calculer préalablement le nombre de relations trouvées par notre méthode, le nombre de relations jugées incorrectes par l'expert du domaine et le nombre de relations jugées comme étant oubliées par notre méthode.

Nous exécutons tout d'abord notre algorithme sur les 70 documents XTab sélectionnés. Puis, l'expert du domaine passe en revue tous les résultats obtenus afin d'évaluer la pertinence et la

[9] la majorité des publications scientifiques diffusées sur le Web sont dans le format PDF

correction des relations sémantiques obtenues par notre méthode.

Pour identifier les relations sémantiques représentées dans un tableau au format XTab, l'expert dispose de l'accès à la totalité du document d'origine – *HTML ou Pdf* . Cependant, l'expert ne considère dans le document que les informations contenues dans le tableau i.e. le titre du tableau, les titres des colonnes et le contenu du tableau. L'expert considère qu'une relation sémantique est *correcte* si la relation est représentée dans le tableau et si tous ses attributs ont été correctement identifiés. S'il reconnaît dans le document XTab une relation sémantique qui n'a pas été identifiée par notre méthode, il considère alors que la relation a été *oubliée*. De cette manière, nous pouvons déterminer quelles sont les relations sémantiques fournies par notre système et qui sont soit incorrectes soit oubliées.

L'expert du domaine considère qu'une relation est représentée dans le tableau pour les trois cas suivants :

1. *Relations complètement représentées* : si chacun des attributs est représenté par une colonne du tableau ;

2. *Relations partiellement représentées avec constantes* : s'il s'agit d'une relation pour laquelle au moins deux attributs ont été représentés dans les colonnes du tableau et dont au moins un attribut a été représenté par une constante dans le titre du tableau ;

3. *Relations partiellement représentées avec attributs Null* : s'il s'agit d'une relation pour laquelle au moins deux attributs ont été représentés dans les colonnes du tableau et dont un ou plusieurs attributs n'ont pas été représentés dans le tableau.

Nous présentons ci-dessous des exemples de tableaux qui nous permettront d'illustrer le cas où les relations sémantique reconnues sont incorrectes ou oubliées. Nous illustrerons également les cas où les attributs supplémentaires sont pertinents.

Relations complètement identifiées. Une relation complètement identifiée est dite correcte si les valeurs des colonnes associées à ses attributs représentent bien les valeurs de ces attributs. Dans la table 3.1 nous présentons un exemple de tableau où la relation $foodMicoorganism$ complètement identifiée dans le tableau est incorrecte. En effet, la première colonne du tableau est associée au A-Term *microoganism* alors que les valeurs de cette colonne sont de type numérique. Il s'agit ici d'un cas où l'exploitation du titre de la colonne nous a conduit à une relation incorrectement identifiée.

Listeria Qty	**Food**	**Temp.**
13 %	Cabbage	10
25 %	Milk	4
7 %	Meat	4

TAB. 3.1 – Exemple de tableau où la relation reconnue est incorrecte

Une relation complètement identifiée peut être oubliée par notre méthode, si l'ensemble des ses attributs n'a pas été identifié dans les colonnes du tableau, comme, par exemple, dans le tableau 3.1 la relation $foodFactor$ a été oubliée par notre méthode car l'attribut $Factor$ qui est représenté dans la troisième colonne n'a pas été reconnu.

Relations partiellement identifiées. Une relation partielle avec constante est dite correcte, si la constante identifiée dans le titre représente bien l'attribut de la relation auquel elle a été affectée. Une erreur dans l'identification d'une relation partielle avec constante peut être due aux mêmes raisons que pour les relations complètement identifiées ainsi qu'au cas où l'association d'une constante à un attribut est incohérente. Par exemple, dans le tableau 3.2, la relation $foodFactorMicroorganism$ partiellement identifiée, on a associé à l'attribut *Factor* la constante ***State*** apparaissant dans le titre du tableau. La constante *State* ne représente cependant pas le terme *State* qui est une spécialisation de l'attribut *Factor* dans l'ontologie. Cette relation est donc considérée comme incorrectement identifiée.

TITRE : Experiments realized in United **States** *of America (USA)*

Microorganism	**Food**
Listeria	Cabbage
E. coli	Milk
Spores	Meat

TAB. 3.2 – Exemple de tableau où la constante reconnue est incorrecte

Une relation partielle avec constante est dite oubliée pour les mêmes raisons que pour les relations totales et dans le cas où il existe dans le titre du tableau une constante qui est une instance d'un attribut de relation et que la méthode n'a pas pu l'identifier.

Pertinence des attributs supplémentaires. un attribut générique est dit pertinent si la présence des informations qu'il représente ajoute un degré de précision à l'interprétation des relations sémantiques représentées dans le tableau.

Microorganism	**Abbrev.**	**Food**	**F. analyses**
Attaching and effacing E. coli	A/EEC	Cabbage	Ph
Enteroinvasive E.coli	EIEC	Milk	Temp.
Diffuse adherent E. coli	DAEC	Meat	Chlor.

TAB. 3.3 – Exemple de tableau : pertinence des attributs supplémentaires

Dans ce tableau deux A-Terms ont été reconnus correspondants à la première et à la troisième colonne. La deuxième et la quatrième colonne ont été représentées par deux attributs génériques ajoutés à la relation $foodMicroorganims$. Dans cet exemple le premier attribut générique (2ème colonne) n'ajoute aucune précision à l'interprétation de la relation ; comme nous disposons des noms complets des microorganismes les abréviations sont donc inutiles. Par contre, la présence du deuxième attribut générique (4ème colonne) est très importante car il indique le nom du facteur analysé dans chaque expérimentation.

3.3 Présentation des résultats

Dans cette section nous présentons les résultats que nous avons obtenus lors de l'expérimentation de notre méthode d'enrichissement sémantique XTab2SML. Le domaine d'application choisi est le risque alimentaire qui est représenté dans l'ontologie Sym'Previus.

3.3.1 Résultats de l'enrichissement sémantique

Pour évaluer notre approche, nous avons considéré les résultats trouvés pour les trois types d'identification de relations sémantiques dans les tableaux : (i) les relations Complètement Représentées (CR), (ii) les relations Partiellement Représentées et où certains attributs ont été identifiés par des constantes dans le titre du tableau (PRC) et (iii) les relations Partiellement Représentées et où les attributs non identifiés dans le tableau sont représentés par des attributs Null (PRN). On note que les relations PRC peuvent être identifiées uniquement dans les tableaux ayant un titre [10].

Dans la Figure 3.2, nous montrons les résultats obtenus par notre système pour chaque type de relation (CR, PRC et PRN). Nous donnons le nombre de relations sémantiques ayant été trouvées par notre méthode. Parmi celles-ci les relations ayant été jugées incorrectes par l'expert du domaine et les relations sémantiques ayant été oubliées par notre méthode. On note que les résultats que nous présentons ici ont été obtenus pour un seuil th fixé de manière empirique à 0,3.

	Relations trouvées	**Relations incorrectes**	**Relations oubliées**
CR	115	37	30
PRC	6	4	5
PRN	112	53	3

FIG. 3.2 – *Les résultats obtenus par l'expert après la vérification des relations sémantiques.*

Sur ces résultats, nous avons calculé les mesures du rappel, de la précision et de la F-Mesure. Soient T et T' deux variables qui représentent les types des relations sémantiques considérées dans le calcul des trois mesures. Les variables T et T' prennent des valeurs dans : {CR, (CR et PRN), (CR, PRN et PRC)}. Soit $Rels_Correctes(T)$ le nombre de relations sémantiques de type T correctement trouvées par notre méthode.

$$Rels_Correctes(T) = Rels_Trouvees(T) \; - \; Rels_Incorrectes(T)$$

Rappel. Il s'agit du pourcentage des relations (de tout type) réellement représentées dans les tableaux et ayant été trouvées par notre méthode. Supposons que T'=$\{CR,\ PRN\ et\ PRC\}$.

$$Rappel = \frac{Rels_Correctes(T)}{(Rels_Correctes(T')) + \; Rels_Oubliees(T'))}$$

Précision. Il s'agit du pourcentage de relations trouvées dans les tableaux par notre méthode et ayant été correctement identifiées.

$$Precision = \frac{Rels_Correctes(T)}{Rels_Trouvees(T)}$$

F-Mesure. Cette mesure permet d'évaluer notre méthode sur sa capacité à assurer un compromis entre la valeur du rappel et de la précision.

$$F - Mesure = \frac{2 \; * \; Rappel \; * \; Precision}{Rappel \; + \; Precision}$$

[10] Dans certains documents d'origine, il est difficile pour le module Any2Xtab d'extraire le titre du tableau soit parce qu'il est simplement absent soit parce qu'il est enfoui dans le texte environnant le tableau.

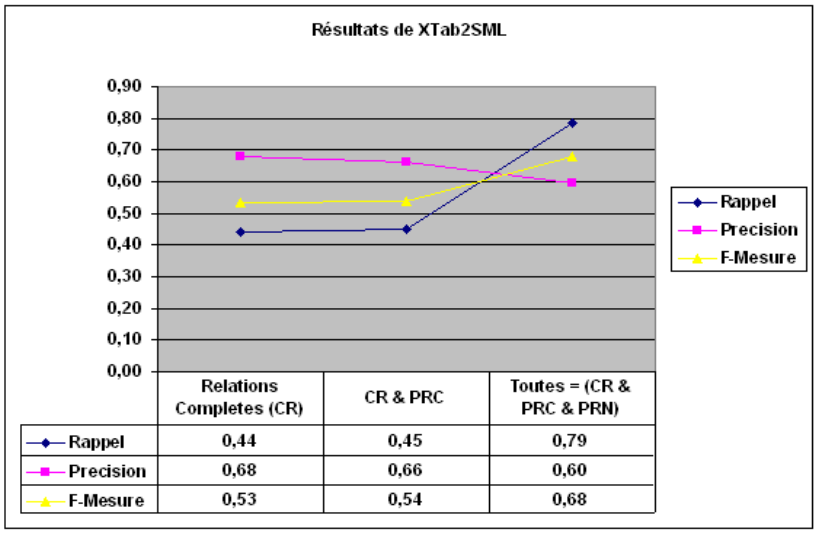

FIG. 3.3 – Rappel, Précision et F-Mesure pour un seuil $th = 0,3$

Validation des résultats

Le diagramme présenté en Figure 3.3 montre les résultats en termes de précision, de rappel et de F-Mesure de notre méthode d'enrichissement sémantique. La première observation importante est que, dans le cas où nous nous contentons des relations complètement identifiées nous obtenons alors seulement une valeur de 0,44 pour le rappel alors que dans le cas où nous considérons tout type de relation identifiée, qu'elle soit complètement ou partiellement identifiée, nous obtenons alors une valeur 0,78 pour le rappel. Ce résultat montre l'intérêt de la conservation des relations partiellement identifiées et donc de la flexibilité du format SML dans l'augmentation du rappel.

Notre principal objectif est précisément d'obtenir un rappel satisfaisant pour notre méthode qui est complètement automatique. En effet, nous avons choisi de garder toute information identifiée ainsi que les informations qui ne sont pas complètement identifiées comme les relations partielles et les attributs supplémentaires. Un autre résultat important est l'augmentation de la F-Mesure grâce au fait que la valeur du rappel augmente de 34 % entre le cas CR et le cas CR & PRC & PRN alors que la précision ne chute que de 8 %. Notre méthode XTab2SML assure donc le compromis entre le rappel et la précision.

3.3.2 Résultats de la découverte de relations candidates

Les résultats de l'évaluation de l'identification des relations sémantiques de l'ontologie dans les documents XTab ont confirmé l'intérêt de cette première flexibilité dans la représentation des documents en SML.

Nous montrons dans la suite les résultats de l'évaluation concernant la découverte de relations sémantiques candidates à l'enrichissement d'un entrepôt thématique ou de l'ontologie du domaine. Ces relations candidates à l'enrichissement peuvent être représentées par une relation comportant un ensemble d'attributs supplémentaires ou par une relation générique.

Relations comportant des attributs supplémentaires

Ces relations comportent les attributs reconnus et intégrés dans des relations sémantiques de l'ontologie mais elle contiennent aussi des attributs supplémentaires. Nous avons pu voir que 40% (soit 28/70) des tableaux testés contiennent des colonnes ne pouvant être associées à aucune relation sémantique identifiée ; ces colonnes sont donc représentées par des attributs supplémentaires. Ces derniers jouent essentiellement deux rôles : le premier consiste en la complétion d'une relation sémantique reconnue ce qui permet à l'utilisateur d'avoir une interprétation plus précise de la relation ; le second consiste en la représentation des données d'attributs non reconnus dans une relation déjà reconnue. Environ 67 % des attributs supplémentaires viennent compléter des relations sémantiques identifiées ou, par distribution des attributs, permettent de représenter de nouvelles relations. En effet, une distribution des attributs reconnus et des attributs supplémentaires peut représenter des relations sémantiques existantes ou nouvelles et candidates à l'enrichissement de l'entrepôt. Les résultats montrent qu'environ 33% (16/49) des attributs représentent des attributs oubliés dans des relations reconnues.

	Nombre d'attributs
Attributs supplémentaires	49
Capture des attributs oubliés	16
Complétion de relations	33

FIG. 3.4 – *Résultat de l'évaluation de l'identification des attributs*

Une relation générique

Elle permet de représenter les liens sémantiques existant entre les données d'un tableau, dans le cas où aucune des relations sémantiques de l'ontologie n'est reconnue. Une relation générique contient soit des attributs reconnus mais qui ne peuvent pas être combinés afin d'instancier une relation existante, soit des attributs représentant des colonnes non reconnues. La Table 3.4 présente les statistiques concernant l'identification des relations sémantiques de l'ontologie dans les documents XTab testés. Dans ces tableaux, 34 relations génériques ont été générées, 139 relations sémantiques ont été reconnues correctes et 38 relations sémantiques de l'ontologie ont été oubliées.

Parmi les relations génériques candidates à l'enrichissement d'un entrepôt thématique ou à l'enrichissement de l'ontologie du domaine, nous distinguons trois cas :

- ⋆ des relations déjà existantes dans l'ontologie pour lesquelles nous n'avons pas pu reconnaître les attributs nécessaires à leur instantiation (au moins deux attributs). Il s'agit dans ce cas de la découverte d'instances de relations sémantiques existantes mais oubliées ;

- ⋆ des relations représentant des informations du domaine de l'ontologie (le risque microbiologique dans les aliments) et pour lesquelles la combinaison des attributs ne peut pas correspondre à une relation de l'ontologie. Il s'agit dans ce cas d'une relation du domaine, candidate à l'enrichissement non seulement de l'entrepôt mais également de l'ontologie du domaine ;

- ⋆ des relations génériques représentant des données d'un domaine connexe. Dans ce cas, nous

pouvons envisager de proposer aux experts du domaine ces relations découvertes, en vue d'étendre l'ontologie existante en une ontologie plus générale, par exemple, en transformant l'ontologie du risque microbiologique dans les aliments en une ontologie traitant également du domaine du risque chimique dans les aliments. En effet, dans notre ensemble de tests, nous avons traité des tableaux qui concernent le domaine du risque chimique et des tableaux qui concernent le domaine de l'épidémiologie.

	Nombre de relations
Relations trouvées	233
Relations trouvées correctes	139
Relations oubliées	38
Relations incorrectes	94
Relations génériques	34

TAB. 3.4 – Résultat de l'évaluation de la découverte de relations sémantiques

Les résultats présentés dans la Figure 3.5 montrent que 20% (7/34) des relations génériques permettent de représenter des données de relations oubliées (parmi 38 relations oubliées). Ces résultats montrent également que 55% (19/34)des relations génériques représentent des relations pertinentes pour l'enrichissement de l'entrepôt et donc intéressantes, par la suite, pour l'interrogation par les utilisateurs. Le faible nombre de relations génériques 23% (8/34) qui ne présentent pas d'intérêt pour le domaine peut s'expliquer par la précision de la directive appliquée lors de la collecte des documents sur le Web. En effet, si les critères de sélection du document sont assez précis, les tableaux contenus dans le document ont de fortes chances d'être pertinents pour le domaine et donc de donner lieu à une relation générique intéressante.

	Nombre de relations
Relations non pertinentes	8
Relations de l'ontologie oubliées	7
Relations pertinentes	19

TAB. 3.5 – Intérêt des relations génériques

3.4 Conclusion

Dans ce chapitre, nous avons présenté les résultats de l'expérimentation de notre méthode d'enrichissement sémantique. Cette expérimentation a été effectuée sur des données réelles, traitant du domaine du risque alimentaire, collectées à partir du Web. À travers cette expérimentation nous avons pu montrer l'intérêt de représenter les tableaux enrichis dans un format flexible. En effet, cela a permis de pouvoir conserver à la fois des informations non complètement identifiées et des informations partiellement ou non identifiées. Les résultats obtenus pour le rappel de la méthode ont pu confirmer l'intérêt de conserver ces informations imparfaitement identifiées. En effet, cela permet d'obtenir une augmentation très significative du rappel tout en assurant une précision satisfaisante. Nous sommes conscients que les résultats concernant le rappel et la précision de la méthode peuvent être améliorés en utilisant des procédures de mise en correspon-

dance moins basiques que les nôtres, telle qu'une mesure utilisant à la fois des critères syntaxiques mais aussi des mesures de similarité utilisant les structures arborescentes de la taxonomie [RS07].

À travers cette expérimentation nous avons également montré que les informations imparfaitement identifiées peuvent être exploitées pour enrichir un entrepôt de données par des informations qui peuvent être interrogées pour compléter des réponses aux requêtes de l'utilisateur ou pour obtenir des réponses approchées en cas d'absence de réponses. Nous avons vu que parmi les relations découvertes dans les informations imparfaitement identifiées, certaines peuvent être proposées à l'expert du domaine pour un enrichissement de l'ontologie par de nouvelles relations ou même pour une extension de l'ontologie à d'autres domaines connexes. L'approche pourrait également être utilisée pour découvrir des valeurs du tableau qui peuvent être également proposées à l'expert du domaine pour enrichir la taxonomie de termes du domaine.

Deuxième partie

Réconciliation de données conformes à un même schéma

Introduction

Le problème de réconciliation de données est un problème majeur pour l'intégration ou la fusion de données provenant de plusieurs sources. Il consiste à décider si deux descriptions provenant de sources distinctes réfèrent ou non la même entité du monde réel (e.g. la même personne, le même article, le même gène). Il est très difficile d'attaquer ce problème dans toute sa généralité car les causes d'hétérogénéité dans la description de données provenant de différentes sources sont variées et peuvent être de nature très différente. L'hétérogénéité des schémas est une des causes premières de la disparité de description des données entre sources. De nombreux travaux, dont on peut trouver une synthèse dans [RB01, SE05, Noy04, EV04], ont proposé des solutions pour réconcilier des schémas ou des ontologies par des mappings. Ces mappings peuvent ensuite être utilisés pour traduire des requêtes de l'interface de requêtes d'une source vers l'interface de requêtes d'une autre source.

L'homogénéité ou la réconciliation de schémas n'empêchent cependant pas les variations entre les descriptions des instances elles-mêmes. En effet, lors de l'intégration de données provenant de différentes sources des représentations différentes peuvent référer la même entité du monde réel car, des vocabulaires et des référentiels différents sont utilisés pour décrire les données.

Toute donnée peut être définie par un identifiant – une *référence* – et par une description. Dans les systèmes d'intégration, les données proviennent de différentes sources, ces sources sont hétérogènes et elles ont été créées de manière autonome. Dans un tel contexte hétérogène, nous ne pouvons pas faire l'hypothèse d'identifiant unique, autrement dit, deux identifiants différents peuvent référer la même entité du monde réel. Nous préférons donc utiliser le terme référence au lieu d'identifiant. Ainsi dans la suite du manuscrit, nous mentionnerons "*le problème de réconciliation de données*" par "*le problème de réconciliation de références*".

Une méthode de réconciliation de références doit pouvoir faire face à une multitude de difficultés dues à l'hétérogénéité des données. Nous allons illuster sur un exemple, présenté en Figure 5, les différents types de difficultés auxquelles les méthodes de réconciliation de références sont confrontées :

- ▷ *Différentes conventions et différents vocabulaires.* Lorsque les différentes sources de données sont conçues de manière autonome, les conventions et les vocabulaires utilisés pour la représentation et le stockage des données peuvent être différents d'une source à une autre mais également au sein d'une même source. Dans les deux sources S1 et S2 de la Figure 5, des vocabulaires différents sont utilisés pour décrire les musées, par exemple, dans l'attribut *Adresse* dans la source S1, les valeurs *rue* et *place* sont représentées par leur abréviation *r.* et *pl.* Les conventions suivies, si elles existent, diffèrent d'une source à une autre, par exemple, pour l'attribut *Contact*, dans la source S1 sa valeur est représentée par un ensemble de numéros de téléphone sur dix chiffres alors que dans la source S2 la valeur de *Contact*, est soit représentée par une adresse e-mail, soit par un numéro de téléphone

contenant en plus l'indicatif international. De plus, les références peuvent être décrites par des informations différentes associées aux mêmes attributs, par exemple, l'adresse du musée du louvre est formulée de manière complètement différente dans les deux sources et les contacts donnés pour les musées sont différents.

▷ *Informations incomplètes.* Dans le contexte de données collectées à partir du Web ou obtenues par la fusion de différentes sources, il est très probable que des informations soient non renseignées. Par exemple, aucun contact n'est donné pour le musée d'*ORSAY* dans la source S1 et le nombre d'œuvres exposées au musée de *Cluny* n'est pas renseigné.

▷ *Présence de données erronées.* Des erreurs de saisie peuvent apparaître dans les descriptions des références. Par exemple, le nom du musée du Louvre contient une erreur : il manque une lettre *u*.

▷ *Fraîcheur des données.* Comme les données sont créées à des moments différents et peuvent être mises à jour à des périodes différentes selon les sources, alors il peut arriver que des informations soient différentes à cause de la non fraîcheur des données. Par exemple, le nombre d'œuvres exposées dans le musée du Louvre de la source S1 est différent de celui donné dans la source S2. En effet, cette valeur évolue dans le temps. De plus, cette différence peut être aussi due à la différence de la sémantique associée à cet attribut : dans la source S1, il s'agit du nombre d'œuvres inventoriées (nombre d'œuvres exposées + le nombre d'œuvres entreposées) et dans la source S2, il s'agit du nombre d'œuvres effectivement exposées dans le musée. Il s'agit donc d'un cas où la résolution de schémas pour l'attribut *Nb-œuvres* n'a pas été traité de manière très fine ce qui a été répercuté au niveau de l'hétérogénéité des données.

Exemple – Données à réconcilier. Soient S_1 et S_2 deux sources de données différentes. Elles contiennent des listes de musées décrits par leur *nom*, leur *adresse*, par leur(s) *contact* et par le nombre d'œuvres qui y sont présentées.

S1 :

Musées	**Adresse**	**Contact**	**Nb-œuvres**
Musée du Lovre	Palais Royal, 75001 Paris	0140205050 | 0140205317	140 000
Musée de Cluny	6, pl. Paul Pain-levé, 75005 Paris	0153737816	3600
Musée d'ORSAY	r. de la Légion d'Honneur, 75007 Paris		4000

S2 :

Musées	**Adresse**	**Contact**	**Nb-œuvres**
Louvre	99, rue Rivoli, Paris	info@louvre.fr	30 000
Cluny	6, place Paul Painlevé	33 (0) 1 53 73 78 00	
Branly	37, quai Branly, Paris	33 (0) 1 56 61 70 00	4000

FIG. 5 – *Exemple de descriptions de données à réconcilier*

Problème de réconciliation de références. Considérons les deux sources de données S_1 et S_2 qui sont conformes au même schéma. Soient I_1 et I_2 les deux ensembles de références correspondant respectivement aux données des deux sources S_1 et S_2. Le problème de réconciliation de références entre S_1 et S_2 consiste à partitionner l'ensemble $I_1 \times I_2$ des paires de références en deux

sous-ensembles REC et $NREC$ regroupant respectivement les paires de références représentant une même entité du monde réel et les paires de références représentant deux entités différentes du monde réel.

Compte tenu du contexte hétérogène des données, l'objectif d'une méthode de réconciliation de références est de fournir des décisions de réconciliation ou de non réconciliation pour les paires de références. Une méthode de réconciliation est *totale* si elle fournit une décision de réconciliation ou une décision de non réconciliation pour toute paire de références $(i_1, i_2) \in I_1 \times I_2$.

Évaluation des méthodes de réconciliation de références. Pour évaluer la qualité des résultats obtenus par une méthode de réconciliation de références, des mesures très utilisées dans le domaine de la Recherche d'Informations (IR) peuvent être calculées. Il s'agit principalement des mesures suivantes :

- *Rappel* : est la proportion, parmi tous les couples possibles de $I_1 \times I_2$, de ceux pour lesquels la méthode a produit un résultat correct.

- *Précision* : est la proportion, parmi les couples pour lesquels la méthode a produit un résultat (de réconciliation ou de non réconciliation), de ceux pour lesquels le résultat est correct.

- *F-Mesure* : elle est calculée pour combiner la valeur du rappel avec celle de la précision. Il est facile à une méthode de réconciliation de références d'obtenir un très bon rappel et une très mauvaise précision par la réconciliation de toutes les paires ou inversement obtenir une très bonne précision et un très mauvais rappel par la non réconciliation de toutes les paires. En revanche, trouver un bon compromis entre le rappel et la précision est un objectif plus difficile à atteindre, c'est pour cela que l'on peut évaluer les résultats en calculant la combinaison des deux mesures. Ce calcul est effectué par la mesure F-mesure qui correspond à : $F-Mesure = (2 * Rappel * Precision) \div (Rappel + Precision)$.

Compte tenu des difficultés que nous venons de mentionner, il n'existe pas de méthode de réconciliation totale et correcte (i.e. F-mesure =1). Pour surmonter en partie ces difficultés, une méthode peut exploiter des connaissances complémentaires comme des connaissances acquises sur un échantillon de données étiquetées manuellement comme réconciliées ou non réconciliées ou des connaissances du domaine fournies par un expert.

Dans cette thèse nous proposons deux méthodes de réconciliation de références : L2R – une méthode Logique pour la Réconciliation de Références (Chapitre 6) et N2R – une méthode Numérique pour la Réconciliation de Références (Chapitre 7). Ces deux méthodes sont guidées par les connaissances du domaine déclarées dans l'ontologie. La première méthode est partielle mais correcte. La seconde, peut être utilisée indépendamment ou pour compléter les résultats de la première en détectant des réconciliations possibles.

Chapitre 4

État de l'art

Sommaire

4.1 Introduction de l'état de l'art 74
4.2 Mesure de similarité pour la réconciliation de références 76
4.2.1 Mesures fondées sur les caractères 76
4.2.2 Mesures fondées sur les chaînes atomiques 77
4.2.3 Mesures hybrides 79
4.2.4 Réconciliation de références fondée sur la description non structurée des références 80
4.2.5 Discussion 81
4.3 Réconciliation de références et exploitation des relations entre données 83
4.3.1 Méthodes locales de réconciliation de références 83
4.3.2 Méthodes globales de réconciliation de références 90
4.3.3 Discussion 102
4.4 Réconciliation de références et exploitation de connaissances pour la réconciliation 103
4.4.1 Méthodes supervisées de réconciliation de références 103
4.4.2 Méthodes déclaratives de réconciliation de références 107
4.4.3 Discussion 111
4.5 Réduction du nombre de comparaisons de références 112
4.6 Synthèse 114

Dans ce chapitre nous dressons un état de l'art des différentes approches de réconciliation de références. Nous avons classé les approches abordées dans la littérature selon deux critères : (i) l'exploitation des relations entre données, et (ii) l'exploitation de connaissances pour la réconciliation de références.

4.1 Introduction de l'état de l'art

Dès lors que les applications informatiques ont commencé à être utilisées pour automatiser de plus en plus les tâches administratives, les recensements démographiques, les expérimentations médicales et épidémiologiques, on s'est intéressé aux données qui pourraient être obtenues grâce à la fusion de différentes sources. Ces sources pouvaient déjà être qualifiées d'hétérogènes puisqu'elles étaient créées par différentes organisations, par différentes personnes et mises à jour à différents moments.

Historiquement, la réconciliation de références a été mentionnée pour la première fois sous la terminologie du "*Record Linkage*" par Halbert Dunn [Dun46]. En effet, dans les années cinquante, les données étaient représentées par des enregistrements (record) dans des fichiers, ce qui justifie l'emploi du terme "*Record Linkage*" pour nommer la tâche de réconciliation de références dans un système d'intégration de données.

La première méthode de réconciliation de références assistée par un ordinateur a été, à l'origine, proposée par le généticien Howard Newcombe [NKAJ59]. Il a introduit l'utilisation des fréquences de valeurs dans les descriptions des références mais également l'utilisation de règles de décision de réconciliation et de non réconciliation des références (enregistrements). Sa méthode avait pour objectif de maintenir des fichiers décrivant des informations cliniques sur des individus. Fellegi et Sunter [FS69] ont fourni un modèle mathématique formel pour les idées introduites par Newcombe [NKAJ59, NK62]. Ils ont fourni différentes méthodes pour estimer les différents paramètres concernant les probabilités de réconciliation. De plus leur méthode a été expérimentée sur des applications réelles caractérisées par des grandes quantités de données et par la présence plus importante de données erronées. Il s'agissait de données médicales et de données concernant le recensement de populations. Face à des données nombreuses, il est crucial de disposer de méthodes de réconciliation efficaces permettant de limiter au maximum les tâches manuelles et permettant d'assurer un taux d'erreur minimal dans les décisions de réconciliation et de non réconciliation des références.

Depuis, le problème de réconciliation de références a été étudié dans une multitude de domaines et sous différentes terminologies. Dans le domaine des *Bases de Données*, lors de la fusion de plusieurs bases ou de leur nettoyage, on cherche à détecter les enregistrements qui se réfèrent à la même entité du monde réel ; on parle alors d'appariement de données, de merge/purge et de détection de doublons. Dans le domaine du *Traitement Automatique du Langage Naturel*, lorsqu'on souhaite trouver quels sont les groupes nominaux qui se réfèrent à la même entité, on parle de co-référence et de résolution des anaphores. En *Traitement d'Images*, la recherche de correspondances d'objets permet de détecter que deux régions dans deux images différentes représentent le même objet. Afin d'automatiser et de rendre efficace la réconciliation de références, un grand nombre de méthodes ont recours à des techniques *d'Intelligence Artificielle* et *d'Apprentissage Artificiel* qui utilisent les termes d'appariement de références, de réconciliation de références, de résolution d'entités et de record linkage.

Dans les approches traditionnelles, pour décider de la réconciliation de paires de références, on mesure leur similarité en se fondant uniquement sur leur description syntaxique structurée (ou non) en attributs. Dans ce cas, on parlera alors d'*approches locales.*

Des travaux plus récents ont souligné le fait que la plupart des décisions de réconciliation et de non réconciliation doivent être effectuées de manière dépendantes les unes des autres. En effet, des informations relationnelles sont souvent représentées dans les données. Par exemple, la peinture "*La Joconde*" est présentée au musée du "*Louvre*" et elle a été réalisée par le peintre

"*Leonardo Da Vinci*". Afin d'améliorer la qualité des résultats de la réconciliation de références, ces informations relationnelles peuvent être exploitées et ainsi compléter les informations prises en compte pour les décisions de réconciliation des références. Il s'agit alors d'approches de réconciliation de références dites *globales* où l'on exploite à la fois la similarité des valeurs des attributs mais aussi celles des références reliées par des relations. Par exemple, pour réconcilier deux peintures, on exploite la similarité des musées où elles sont présentées et la similarité de leur peintre ; pour réconcilier deux musées, on exploite la similarité des peintures qui y sont présentées, de leur villes de localisation, et ainsi de suite. Cette interdépendance des décisions rend la tâche de réconciliation plus complexe. D'une part, les tâches élémentaires (e.g. calcul de similarité) deviennent plus coûteuses et d'autre part, elles doivent être itérées pour prendre en compte l'interdépendance entre réconciliations. Des approches telles que Ananthakrishna et al. [ACG02], Dong et al. [DHM05] et Bhattacharya et al. [BG06] ont montré l'intérêt d'exploiter les informations relationnelles dans la tâche de réconciliation de références.

Le développement de nouvelles techniques de classification en apprentissage automatique et en statistique a conduit à l'apparition de nouvelles techniques de réconciliation de références fondées sur l'apprentissage automatique supervisé ou semi-supervisé. Ces techniques exploitent des échantillons de données sous la forme d'un ensemble de paires de références étiquetées manuellement par un expert comme réconciliées ou non réconciliées. Un grand nombre d'approches supervisées se sont appuyées sur le modèle probabiliste proposé par Fellegi et Sunter [FS69] où la réconciliation de références est vue comme un problème de classification : à partir d'un vecteur de scores de similarité des valeurs des attributs d'une paire de références il s'agit de classifier cette paire comme réconciliée ou non réconciliée. Winkler [Win06] a proposé une synthèse bibliographique des travaux les plus importants relevant de cette approche classique de réconciliation de références. Certains travaux se sont intéressés à l'utilisation des techniques de l'apprentissage actif pour minimiser le besoin de données étiquetées manuellement (par exemple [BM03b]).

Une des limitations les plus importantes des approches supervisées est le besoin de données étiquetées pour apprendre différents paramètres (e.g. probabilités, pondérations, similarités) ou règles de décision. Cette étape d'apprentissage doit être renouvelée dans le cas d'un changement de domaine mais aussi dans le cas où les caractéristiques des données évoluent. Pour éviter cette contrainte, des approches telles que [ME96, Coh00, GKMS04] ont proposé l'utilisation d'algorithmes de calcul de similarité entre valeurs. En utilisant ces algorithmes et des seuils de (non) réconciliation il est possible de prendre des décisions de (non) réconciliation sans l'exploitation de données d'apprentissage.

Une autre façon de limiter le recours à des humains pour l'étiquetage manuel des données (nécessaires dans les méthodes de réconciliation supervisées) consiste à représenter et exploiter des connaissances du domaine. Ainsi, des travaux se sont intéressés à l'exploitation des connaissances du domaine, d'une part pour réduire le problème de la variation syntaxique des données, par exemple les abréviations (e.g. *bd vs boulevard*), la fréquence des noms dans un pays (e.g. *Dupont* en France), et d'autre part pour renforcer le mécanisme d'inférence des réconciliations de références. Par exemple, les contraintes de clé-primaire, de clé-étrangère et les contraintes d'exclusion en bases de données relationnelles peuvent être exploitées. Ces connaissances peuvent être soit codées directement dans les méthodes de manière ad-hoc soit exprimées déclarativement sous forme de règles de réconciliation et/ou de non réconciliation.

Avant de présenter ces différentes méthodes de réconciliation de références, nous allons exposer, en Section 4.2, les principales techniques de mesure de similarité entre valeurs utilisées par les méthodes de réconciliation de références.

4.2 Mesure de similarité pour la réconciliation de références

Une des difficultés les plus importantes auxquelles sont confrontées les méthodes de réconciliation est la variation syntaxique du vocabulaire utilisé pour décrire les données. Ces variations peuvent être dues aux variations typographiques, aux abréviations (e.g. bd. pour boulevard), à la présence d'acronymes (e.g. JSK pour Jeunesse Sportive de Kabylie), à des différences de codage (e.g. (Y|1) pour oui, (75 | Paris) pour la ville de Paris) et aux synonymes. Dans un tel contexte, la plupart des méthodes de réconciliation de références s'appuient sur des techniques de mesure de similarité entre valeurs.

Ces mesures doivent être conformes à la définition suivante :

Définition – Mesure de similarité. Une mesure de similarité normalisée est une fonction : $Sim : E \times E \rightarrow [0..1]$ appliquée à une paire d'éléments (e.g. chaînes de caractères, nombres) et qui renvoie un nombre réel dans [0..1] exprimant la similarité entre ces deux éléments telle que :

1. $Sim(a, a) = 1$ (maximalité)
2. $\forall a, b \in E, Sim(a, b) = Sim(b, a)$ (symétrie)

On appelle *score de similarité*, la valeur réelle calculée par une mesure de similarité pour une paire d'éléments.

Une multitude de mesures de similarité entre chaînes de caractères ont été développées, chacune est efficace pour certains types particuliers de variations syntaxiques. Dans cette sous-section, nous passons en revue les mesures de similarité les plus importantes qui n'utilisent pas de ressources externes telles que WordNet [Mil95] et nous fournissons quelques exemples pour montrer leurs similitudes et leurs différences.

4.2.1 Mesures fondées sur les caractères

Mesures d'édition. Il s'agit de mesures conçues pour traiter les erreurs typographiques apparaissant dans les chaînes de caractères. Le score de similarité entre deux chaînes de caractères est le coût minimum d'une conversion d'une des deux chaînes en l'autre par une séquence d'*insertions*, de *suppressions* et de *remplacements* de caractères. Á chacune de ces opérations est assigné un coût. La mesure dite de Levenshtein [Lev66] est une mesure d'édition où le coût de toutes les opérations est identique et égal à 1. Par exemple, la mesure de Levenshtein appliquée aux chaînes *Marie* et *Marria* est de 2. En effet, *Marie* est obtenue par la suppression d'un caractère *r* et par le remplacement du caractère *a* par le caractère *e* de la chaîne *Marria*.

Mesures par affinement de gaps. Les mesures d'édition sont inefficaces dans le cas où les chaînes de caractères sont abrégées ou raccourcies (e.g., "*MCF. en informatique* " *vs* "*Maître de conférence en informatique*"). La mesure par affinement de gaps proposée par Smith et Waterman [WSB76] fournit une solution à ce problème en introduisant deux opérations d'édition : *ouvrir un gap* et *étendre un gap*. Le coût de l'extension d'un gap est en général plus faible que le coût d'ouverture d'un gap, ainsi le coût des pénalités du non appariement des gaps est plus faible que celui considéré par les distances d'éditions classiques. Par exemple les deux chaînes σ_1="Bd de Paris" et σ_2="Boulevard de Paris" peuvent être alignées comme présenté dans la figure 4.1.

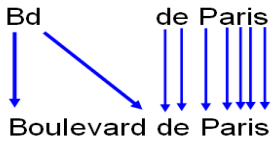

FIG. 4.1 – Smith et Waterman sur les deux chaînes σ_1="Bd de Paris" et σ_2="Boulevard de Paris"

Plusieurs variantes de cette mesure ont été proposées et exploitées. Par exemple, l'algorithme BLAST [AGM+90] utilise une de ces variantes dans des applications d'alignement de séquences d'ADN en biologie.

Mesure de Jaro. Jaro [Jar76] a introduit un algorithme de mesure de similarité entre chaînes de caractères qui est très utilisé pour la comparaison de noms et prénoms de personnes. L'algorithme basique pour calculer le score de similarité par la mesure de Jaro pour deux chaînes de caractères σ_1 et σ_2 s'effectue en trois étapes :

1. Calculer la longueur des chaînes σ_1 et σ_2 ;
2. Calculer le nombre des "caractères communs" dans les deux chaînes ; les *caractères communs* sont tous les caractères $\sigma_1[i]$ et $\sigma_2[j]$ pour lesquels $\sigma_1[i] = \sigma_2[j]$ et $|i - j| \leq (1/2 * \min\{|\sigma_1|, |\sigma_2|\})$;
3. Calculer le nombre de transpositions t ; un *caractère transposé* est un caractère commun qui apparaît dans différentes positions. Par exemple, les chaînes $Marie$ et $Maires$ ont cinq caractères communs dont deux transpositions (les caractères r et i).

Soit N_c (resp. N_t) le nombre de caractères communs (resp. le nombre de transpositions) entre les deux chaînes σ_1 et σ_2. La mesure de Jaro est définie comme suit :

$$Jaro(\sigma_1, \sigma_2) = \frac{1}{3}\left(\frac{N_c}{|\sigma_1|} + \frac{N_c}{|\sigma_2|} + \frac{N_c - N_t/2}{N_c}\right).$$

Par exemple, $Jaro(Marie, Maires) = \frac{1}{3}\left(\frac{5}{5} + \frac{5}{6} + \frac{5-2/2}{5}\right) = 0.87$

D'autres variantes de l'algorithme de Jaro ont été proposées, par exemple la mesure Jaro-Winkler proposée par Winkler et Thibaudeau [WEW91]. Jaro-Winkler associe un poids plus important aux appariement de préfixes. En effet, les appariements au niveau des préfixes sont généralement plus importants que ceux effectués sur le reste de la chaîne lors de l'appariement de noms de personnes.

$$Jaro - Winkler(\sigma_1, \sigma_2) = Jaro(\sigma_1, \sigma_2) + \frac{P'}{10}.(1 - Jaro(\sigma_1, \sigma_2))$$

où $P' = \max(P, 4)$ et où P représente la longueur du plus long préfixe commun entre σ_1 et σ_2.

4.2.2 Mesures fondées sur les chaînes atomiques

Les mesures fondées sur les caractères sont efficaces pour le traitement des erreurs typographiques. Cependant, il arrive souvent que les différences typographiques entre les chaînes de caractères soient dues à l'ordre d'apparition des mots (e.g. "*Prof. Marie-Christine Rousset*" *vs*

"*Rousset Marie-Christine Prof.*"). Les mesures fondées sur les caractères sont incapables de capturer la similarité de telles chaînes de caractères. Afin de tenir compte de ce type de problèmes, des mesures appliquées sur des ensembles de chaînes atomiques ont été proposées.

Une *chaîne atomique* (ou token) est une séquence de caractères alphanumériques délimitée par un caractère de ponctuation. Il s'agit en général des mots formant la chaîne de caractères.

Mesure de Jaccard. Jaccard [Jac01] a proposé une mesure de similarité appliquée sur les mots qui est simplement le rapport entre le cardinal de l'ensemble des mots communs entre les deux chaînes sur le cardinal de l'union des mots des deux chaînes : $\frac{|\sigma_1 \cap \sigma_2|}{|\sigma_1 \cup \sigma_2|}$.
Par exemple, Jaccard("bd de Paris", "boulevard de Paris") = 2/4.

Mesure de Monge et Elkan. Monge et Elkan ont proposé dans [ME96] un algorithme pour la comparaison de champs textuels fondé sur les *chaînes atomiques*. Deux chaînes atomiques s'apparient si elles sont égales, ou si l'une est le préfixe de l'autre. Ainsi, la similarité de deux champs textuels est le nombre de *chaînes atomiques* qui s'apparient divisé par le nombre des *chaînes atomiques*.

Cosinus. La mesure cosinus s'applique à des vecteurs modélisant les chaînes de caractères. La représentation vectorielle la plus utilisée en Recherche d'Information se fonde sur la pondération TF–IDF [Aiz03] (acronyme de Token Frequency–Inverse Document Frequency). L'idée principale est d'associer un poids fort aux chaînes atomiques apparaissant fréquemment dans un document (poids TF) et d'associer un poids plus faible à celles apparaissant fréquemment dans l'ensemble des documents considérés (poids IDF). Cohen [Coh98a] a adapté ce principe de pondération pour la mesure de similarité lors de la réconciliation de références. Il considère la description d'une référence (enregistrement) comme un document et l'ensemble des descriptions de toutes les références comme l'ensemble des documents.

Cohen sépare chaque chaîne de caractères σ en mots et à chaque mot w est associé un poids :

$$v_\sigma = \log(tf_w + 1).\log(idf_w)$$

où tf_w est la fréquence de w dans la description d'une référence et idf_w est égal à $\frac{|D|}{n_w}$, où n_w est le nombre de références dans la base de données D contenant w. Le poids TF/IDF pour un mot w dans une description d'une référence est élevé si w est fréquent dans la description (un grand tf_w) et w est suffisamment "rare" dans la base de données (un grand idf_w). Par exemple, pour une liste de laboratoires de recherche les mots "LRI" ou "LIP6" auront un poids *idf* plus élevé que les mots fréquents tels que "Univ.", "Laboratoire". La mesure de similarité *cosinus* de deux chaînes de caractères σ_1 et σ_2 est alors définie comme le cosinus de l'angle des vecteurs correspondant :

$$sim(\sigma_1, \sigma_2) = \frac{\sum_{j=1}^{|\sigma_1 \cap \sigma_2|} v_{\sigma 1}(j).v_{\sigma 2}(j)}{\parallel v_{\sigma 1} \parallel . \parallel v_{\sigma 2} \parallel}$$

La mesure cosinus est pertinente pour des données volumineuses car, plus le nombre de données à traiter est grand, meilleure est l'estimation des poids TF/IDF. L'introduction des poids pour les mots fréquents n'affecte que sensiblement le score de similarité des chaînes de caractères car les poids *idf* qui leur sont associés sont faibles. Par exemple, "*Prof. Marie-Christine Rousset*" et "*Marie-Christine Rousset*" auront un score très proche de un. En revanche cette mesure ne capture pas les erreurs typographiques. Par exemple, les chaînes de caractères "*Laboratoire de*

Recheche en Informatique" et "*Laboratore de Recherche en Informtique*" auront un score proche de zéro par cette mesure.

4.2.3 Mesures hybrides

SoftTFIDF. Afin de contourner la faiblesse de la mesure cosinus présentée ci-dessus, Bilenko et al. [BMC+03] ont proposé la mesure *SoftTFIDF* pour traiter les erreurs d'orthographe. Dans SoftTFIDF, les paires de chaînes de caractères "similaires" (non nécessairement identiques) sont considérées dans le calcul de similarité par cosinus. Dans la mesure cosinus, seules les chaînes atomiques identiques sont prises en compte dans le calcul, c'est–à–dire celles appartenant à $\sigma_1 \cap \sigma_2$. Dans la mesure SoftTFIDF on relâche cette contrainte de l'égalité. Une mesure de similarité sim' est utilisée pour calculer les scores de similarité des différents tokens. Soit $CLOSE(\theta, \sigma_1, \sigma_2)$ l'ensemble des tokens $w \in \sigma_1$ tel qu'il existe un token $v \in \sigma 2$ tel que $sim'(w, v) > \theta$. Pour tout $w \in CLOSE(\theta, \sigma_1, \sigma_2)$, on a $S(w, \sigma_2) = \max_{v \in \sigma_2} sim'(w, v)$. La mesure SoftTFIDF est définie comme suit :

$$SoftTFIDF(\sigma_1, \sigma_2) = \frac{\sum_{j=1}^{|CLOSE(\theta,\sigma_1,\sigma_2)|} v_{\sigma 1}(j).v_{\sigma 2}(j).S(w, \sigma_2)}{\parallel v_{\sigma 1} \parallel . \parallel v_{\sigma 2} \parallel}$$

N-grammes. La fonction de comparaison *n-grammes* proposée par [Ull77, SD04] exploite l'ensemble de toutes les sous-chaînes de caractères de taille n de chaque chaîne de caractères.
Par exemple pour $n = 3$ et $\sigma = PERNELLE$, l'ensemble X des sous chaînes obtenu est :
$X = \{PER, ERN, RNE, NEL, ELL, LLE\}$

La mesure de similarité entre deux chaînes σ_1, σ_2 est définie comme suit :

$$sim(\sigma_1, \sigma_2) = \sqrt{\sum_{\forall x \in X} |f_{\sigma 1}(x) - f_{\sigma 2}(x)|}$$

où $f_{\sigma 1}(x)$ et $f_{\sigma 2}(x)$ sont le nombre d'occurrences des sous-chaînes x dans les chaînes σ_1 et σ_2 respectivement. Les n-grammes d'une chaîne de caractères σ sont obtenus en glissant une fenêtre de taille n sur la chaîne σ. La distance *bi-grams* (n=2) est largement utilisée, car elle est très efficace pour les erreurs typographiques mineures.

Code *Soundex*. Le code phonétique *Soundex* inventé par Russell et Odell [Rus18], pour la langue anglaise, est utilisé pour regrouper les mots ayant des sons similaires. Par exemple, le code *Soundex* de *Dupond* et de *Dupont* est le même. Un code *Soundex* contient quatre caractères. Les règles de codage Soundex sont :

1. Garder la première lettre du mot comme préfixe et ignorer toute occurrence de W et de H dans les autres positions du mot ;
2. Assigner itérativement les codes suivants pour les lettres restantes : (B, F, P, V → 1) ; (C, G, J, K, Q, S, X, Z → 2) ; (D, T → 3) ; (L → 4) ; (M, N → 5) ; (R → 6)
3. A, E, I, O, U et Y ne sont pas codées.

À titre d'exemple, le code Soundex de *Dupond* et *Dupont* est D153. Dans le cas où la limite des quatre caractères est dépassée, tous les caractères restant sont ignorés. D'autres codages phonétiques des mots ont été proposés tel que NYSIIS [Taf70] qui considère dans le codage la position des voyelles en convertissant toutes les voyelles en lettre A. Dans [Taf70] une comparaison

a été effectuée entre Soundex et NYSIIS sur une base de données de noms de personnes à New York et il conclut que NYSIIS est plus précis que Soundex (98,72 % *vs* 95,99 %)

4.2.4 Réconciliation de références fondée sur la description non structurée des références

Certaines approches de réconciliation de références considèrent la description de chaque référence comme une seule valeur d'attribut (i.e. une seule longue chaîne de caractères). Afin de déterminer les paires de références similaires (ou candidates à la réconciliation), une mesure de similarité est appliquée et ainsi on obtient un score de similarité qui va servir pour l'étape de décision de réconciliation ou de non réconciliation. Monge et Elkan [ME96, ME97] ont utilisé leur mesure de similarité, définie ci-dessus, pour comparer des descriptions de références non structurées en attributs.

William Cohen dans son système WHIRL [Coh00] a proposé une méthode de réconciliation de références dans une base de données relationnelle représentant des informations collectées à partir du Web. Il a suggéré l'utilisation des pondérations TF/IDF et la mesure *cosinus* pour calculer la similarité syntaxique et, ainsi, identifier les paires de références candidates à la réconciliation. Toute valeur d'attribut peut être décomposée en un ensemble de chaînes atomiques. Chacun de ces ensembles de chaînes atomiques est représenté par un vecteur de poids TF/IDF noté v_r.

Dans WHIRL, cette mesure de similarité est utilisée pour définir la *jointure textuelle* entre relations en exploitant les attributs textuels. Soient deux relations R_1 et R_2 à intégrer (ou à fusionner), leur *jointure textuelle* est définie comme suit :

$$R_1 \bowtie_\phi R_2 = \{(r_1, r_2) | r_1 \in R_1,\ r_2 \in R_2 \ et\ sim(v_{r1}, v_{r2}) \geq \phi\}$$

avec ϕ un seuil de réconciliation fixé par l'utilisateur.

Exemple : soient R_1 et R_2 deux relations provenant de deux sources différentes. Elles contiennent des listes de musées décrits par leur nom et leur adresse respective, l'adresse physique pour l'une et l'adresse Web pour l'autre.

Musées	**Adresse**
Musée du Louvre	99, rue Rivoli, 75001 Paris
Musée de Cluny	6, place Paul Painlevé, 75005 Paris
Musée d'ORSAY	1, rue de la Légion d'Honneur 75007 Paris

Musées	**Adresse**
Louvre	http ://www.louvre.fr
Cluny	http ://www.musee-moyenage.fr
Branly	http ://www.quaibranly.fr

FIG. 4.2 – *Exemple de relations à intégrer*

Appliquons la jointure textuelle sur les relations R_1 et R_2 de la Figure 4.2 selon la méthode proposée dans le système WHIRL. Soient r_{11} et r_{21} les premières références des relations R_1 et R_2 que l'on représente par l'ensemble des chaînes élémentaires (i.e. mots) qu'elles contiennent :

r_{11} ={Musée, du, Louvre, 99, rue, Rivoli, 75001, Paris} $\sim$ r_{21} ={ Louvre, http, www, fr }

D ={Musée, du, Louvre, 99, rue, Rivoli, 75001, Paris,Cluny, 6, place, Paul, Painlevé, 75005, d, ORSAY, la, Légion, Honneur, 75007, http, www, fr, moyenage, Branly, quaibranly }

Supposons que les vecteurs de poids TF/IDF des références r_{11} et r_{21} soient représentés par les vecteurs v_{r1} et v_{r2} :

$$v_{r11} = (0.08,\ 0.01,\ 0.8,\ 0.1,\ 0.04,\ 0.9,\ 0.09,\ 0.01)$$
$$v_{r21} = (0.6,\ 0.003,\ 0.01,\ 0.007)$$
$$\text{alors on obtient } sim(v_{r11}, v_{r21}) = 0.8 * 0.6 = 0.48$$

Gravano et al. [GIKS03] ont proposé une méthode qui intègre le système de jointure textuelle du système WHIRL dans un système de gestion de base de données. Pour la mesure cosinus, au lieu de considérer les *mots* comme chaînes atomiques, ils ont considéré les *q-grammes* des chaînes de caractères. Une expérimentation a permis de comparer les résultats obtenus dans le cas où on considère les mots et dans le cas où on considère les N-grammes (avec N=2 et N=3). Elle montre que l'utilisation des q-grammes permet d'obtenir de meilleurs résultats en présence d'abréviations dans les valeurs.

Ces méthodes, qui considèrent la description de chaque référence comme une seule longue chaîne de caractères, permettent de traiter des références qui ne sont pas décrites dans le même schéma ou qui ne sont pas structurées en attributs. Ce type de méthodes est parfois utilisé en pré–traitement pour obtenir un premier ensemble de paires de références candidates à la réconciliation. Par exemple, dans le système DUMAS [BN05], une mesure de ce type est utilisée pour obtenir un ensemble de paires de références candidates qui sont ensuite exploitées pour la réconciliation de schémas XML. Toutefois, ces méthodes ignorent certaines informations qui, si elles existent, peuvent être importantes pour la réconciliation de références : les différents impacts des attributs sur le calcul de similarité ou sur les décisions de réconciliation et la possibilité de choisir les algorithmes de comparaison des valeurs des attributs en fonction de l'attribut. Un autre type d'approche consiste à comparer les valeurs des différents attributs individuellement et à utiliser une mesure de similarité appropriée à chaque attribut et, enfin, grâce à une fonction d'agrégation (e.g. la Moyenne, le Maximum) on obtient un score de similarité pour les paires de références.

4.2.5 Discussion

Le nombre de mesures de similarité des chaînes de caractères reflète le nombre d'erreurs ou de variations syntaxiques qui peuvent surgir dans les données réelles. Bien que des erreurs puissent aussi apparaître dans des valeurs numériques, les travaux de recherche sont encore à l'état embryonnaire tel que Agrawal et Srikant [AS02]. Certains travaux nous guident dans les choix des mesures de similarité en fonction du contexte, des types d'erreurs et des variations syntaxiques rencontrées. Yancey [Yan05] et Cohen et al. [CRF] ont montré que la mesure Jaro-Winkler est la plus efficace pour le calcul de similarité entre noms de personnes dans les données d'un recensement démographique aux États-Unis. Parmi les mesures appliquées aux chaînes atomiques (tokens), la mesure cosinus TF/IDF est celle qui obtient les meilleurs résultats pour les longues chaînes de caractères. Un autre travail important de comparaison des différentes mesures est celui

de Bilenko et al. [BMC+03], où l'on compare l'efficacité des mesures appliquées aux caractères avec celles appliquées aux chaînes atomiques. Ils ont montré que la mesure de Monge-Elkan, fondée sur les chaînes atomiques, obtient les meilleurs résultats parmi toutes les autres mesures expérimentées. Bilenko et al [BMC+03] et Cohen et al. [CRF] ont souligné le fait qu'aucune mesure non hybride n'est appropriée pour tous les jeux de données. Les meilleures mesures pour certains jeux de données peuvent être les plus mauvaises pour d'autres jeux de données. La mesure hybride SoftTFIDF a permis d'obtenir globalement les meilleurs résultats dans les expérimentations de [BMC+03, CRF]. Ils concluent que des mesures plus flexibles qui combinent plusieurs mesures de similarité, telles que SoftTFIDF, sont donc nécessaires. Plus généralement, le choix de ces mesures doit pouvoir être paramétré dans un système de réconciliation de références pour pouvoir s'adapter aux données.

Dans notre méthode de réconciliation de références nous avons considéré que le choix des mesures de similarité peut être paramétré en fonction des différents attributs. Comme les travaux de [BMC+03, CRF] suggèrent, nous avons choisi la mesure de Jaro-Winkler pour la comparaison des valeurs des attributs prenant des valeurs courtes (e.g. noms de personnes). En revanche, nous n'avons pas suivi le choix de la mesure TF/IDF pour les attributs prenant des valeurs de taille longue, car l'étape de calcul de fréquences des mots dans l'ensemble des valeurs des attributs ralentit l'exécution de la tâche de réconciliation.

4.3 Réconciliation de références et exploitation des relations entre données

Dans cette section nous allons présenter différentes approches de réconciliation de références. Nous présentons tout d'abord celles qui exploitent la description syntaxique des tuples de valeurs de différents attributs, il s'agit des approches dites *locales*. Ensuite, nous présentons les approches qui exploitent, en plus des valeurs des attributs, les relations liant les références entre elles, comme par exemple, la relation *Réalisée* entre une référence d'une *Peinture* et une référence d'un *Artiste*. Il s'agit dans ce second cas d'approches dites *globales*.

4.3.1 Méthodes locales de réconciliation de références

Les méthodes locales de réconciliation de références sont des méthodes qui exploitent les descriptions syntaxiques des références. Ces méthodes considèrent les descriptions des références représentées en terme d'ensemble de couples (attribut, valeur). La description d'une référence r est notée par $desc(r)$. Par exemple, les descriptions de deux paires de références de musées $m1$ et $m2$ sont représentées par :

$desc(m1) = \{(Nom,$"*Arts Premiers*"$), (Adresse,$"*27, quai Branly 75007 Paris*"$), (AnnéeIn,$"*2006*"$)\}$
$desc(m2) = \{(Nom,$"*Musée Quai Branly*"$), (Adresse,$"*27, quai Branly 7ème arr. Paris*"$), (AnnéeIn\ ,$"*2006*"$)\}$

Pour pouvoir décider de la (non) réconciliation d'une paire de références, les méthodes locales de réconciliation comparent les valeurs des attributs deux à deux afin de capturer la similitude ou la différence des valeurs des attributs décrivant la paire de références. Par exemple, pour réconcilier la paire de références $m1$ et $m2$, on compare les paires de valeurs de leurs attributs : ("*Arts Premiers*", "*Musée Quai Branly*"), ("*27, quai Branly 75007 Paris*", "*27, quai Branly 7ème arr. Paris*"), etc. Les méthodes s'appuient généralement sur des mesures de similarité appliquées aux attributs ce qui donne un vecteur de comparaison représentant le score de similarité ou la probabilité d'accordance des attributs deux à deux. Différentes méthodes peuvent être utilisées pour décider de la réconciliation ou de la non réconciliation des paires de références. Certaines méthodes s'appuient sur un modèle probabiliste pour estimer les probabilités de réconciliation ou de non réconciliation et sur des règles de décision appliquées sur ces probabilités. D'autres méthodes exploitent des fonctions d'agrégation de ces similarités (e.g. moyenne, moyenne pondérée, calcul de rangs).

Méthodes probabilistes : théorie de Fellegi et Sunter et ses extensions.

Fellegi et Sunter [FS69] ont fourni, pour le problème de réconciliation de références un modèle mathématique pour les idées introduites par Newcombe [NKAJ59, NK62].
Tout d'abord nous avons besoin d'introduire quelques notations. Soient deux sources A et B représentant deux ensembles de références à réconcilier. Considérons le produit cartésien $A \times B = \{(a,b)|a \in A\ et\ b \in B\}$. Deux ensembles disjoints M et U peuvent être définis sur $A \times B$, soient : $M = \{(a,b)|a \equiv b,\ a \in A\ et\ b \in B\}$ et $U = \{(a,b)|a\ !\equiv b,\ a \in A\ et\ b \in B\}$, où le symbole $\equiv$ exprime le fait que les références a et b représentent la même entité du monde réel et le symbole $!\equiv$ exprime le fait que les références a et b ne représentent pas la même entité du monde réel. L'ensemble M est appelé l'ensemble des paires de références *Réconciliées* (*Matched*) et U est appelé l'ensemble des paires de références *non-Réconciliées* (*Unmatched*). La tâche de réconciliation de références consiste à classifier toute paire de références de l'ensemble

$A \times B$ comme appartenant soit à M soit à U. Un troisième ensemble P peut aussi être introduit comme l'ensemble représentant les paires de références dont la réconciliation est *Possible*.

Un vecteur de comparaison γ représente le niveau d'accordance entre les valeurs des attributs d'une paire de références a et b. Si les paires de références sont comparées sur k attributs alors le vecteur γ contient k composantes : $\gamma = (\gamma^1(desc(a), desc(b)), \ldots, \gamma^k(desc(a), desc(b)))$, où γ^i, $1 \leq i \leq k$, représente le degré d'accordance (ou similarité) des valeurs du i-ème attribut. L'ensemble de tous les vecteurs de comparaison forme l'espace de comparaison Γ.

Pour tout vecteur de comparaison γ appartenant à Γ, $m(\gamma)$ représente la probabilité conditionnelle d'observer γ sachant que ce vecteur correspond à une paire de références réconciliée et que cette réconciliation est correcte. De manière similaire, $u(\gamma)$ est définie pour représenter la probabilité conditionnelle d'observer γ sachant que ce vecteur correspond à une paire de références non réconciliée et que cette non réconciliation est correcte. On appelle ces probabilités les *m-probabilités* et les *u-probabilités*. Elles sont définies comme suit :

- $m(\gamma) = Pr(\gamma|M)$ et
- $u(\gamma) = Pr(\gamma|U)$

Fellegi et Sunter ont introduit le ratio de vraisemblance[11] R qui est fonction de γ et qui est utilisé pour prendre les décisions de réconciliations et de non réconciliations :

$$R(\gamma) = m(\gamma)/u(\gamma), avec\ \gamma \in \Gamma$$

Ce ratio $R(\gamma)$, ou son logarithme, est appelé le *poids de la réconciliation*.

Fellegi et Sunter ont défini la règle de *décision* suivante, où T_μ et T_λ sont deux valeurs réelles représentant deux seuils :

- si $(R(\gamma) > T_\mu)$, alors désigner la paire de références correspondant au vecteur de comparaison γ comme réconciliée,
- si $(T_\lambda \leq R(\gamma) \leq T_\mu)$, alors désigner la paire de références correspondant au vecteur de comparaison γ comme candidate à la réconciliation,
- si $(R(\gamma) < T_\lambda)$, alors désigner la paire de références correspondant au vecteur de comparaison γ comme non-réconciliée.

La zone $T_\lambda \leq R(\gamma) \leq T_\mu$ partitionne l'ensemble des $\gamma \in \Gamma$ et les paires de références correspondantes en trois zones nommées, A_1, incluant les paires de références déclarées comme *réconciliées*, A_2, incluant les paires de références déclarées comme *candidates à la réconciliation* et A_3, incluant les paires de références déclarées comme *non réconciliées* .

La figure 4.3 montre les trois zones dans lesquelles les paires de références sont ordonnées de manière décroissante relativement au *poids de réconciliation* $R(\gamma)$. La figure 4.3 montre aussi que le nombre de paires de références désignées comme *réconciliées* est habituellement inférieur à celui des paires désignées comme *non réconciliées*.

L'affectation de paires (a, b), ayant comme vecteur de comparaison γ, à l'ensemble des réconciliations M ou à l'ensemble des non réconciliation U, en fonction de $R(\gamma)$, peut conduire à des fausses classifications. Les *fausses réconciliations* ou les *fausses non réconciliations* sont les deux

[11] Le ratio de vraisemblance permet de refléter dans quelle proportion une hypothèse, ici M, est plus probable qu'une autre, ici U.

types d'erreurs possibles dans le modèle, et μ et λ représentent les taux d'erreur correspondants. Des valeurs élevées de $R(\gamma)$ (voir la zone A_1 dans la Figure 4.3) correspondent à des probabilités faibles pour des *fausses réconciliations*. De manière analogue, des valeurs faibles de $R(\gamma)$ (voir la zone A_3 dans la Figure 4.3) correspondent à des probabilités faibles pour des *fausses non réconciliations*.

Dans la Figure 4.3 la ligne qui traverse toutes les zones représente une tendance possible des probabilités des *fausses réconciliations* et des *fausses non réconciliations*. Les trois zones sont identifiées par des valeurs spécifiques de T_μ et T_λ et, les régions A_1, A_2 et A_3 sont à leur tour divisées en deux sous régions vraies/fausses réconciliations et vraies/fausses non réconciliations, respectivement.

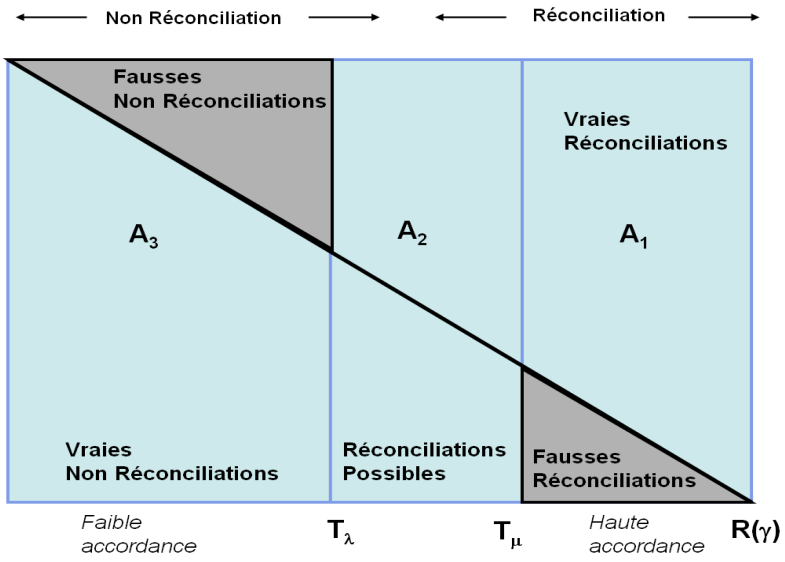

FIG. 4.3 – Les régions du modèle de Fellegi et Sunter

Le problème est tout d'abord de fixer les deux seuils T_μ et T_λ. Pour ce faire, les auteurs proposent de décider quel taux d'erreur on peut accepter dans la règle de décision présentée précédemment ; de tels taux d'erreur correspondent aux deux régions grisées dans la Figure 4.3. Une fois que les taux d'erreur sont donnés a priori, deux seuils peuvent alors être fixés.

Ensuite, la capacité d'estimer les m-probabilités et u-probabilités est cruciale pour l'établissement des décisions de réconciliation ou de non réconciliation. Fellegi et Sunter [FS69] ont proposé des pistes pour estimer ces probabilités sous certaines conditions. Parmi ces pistes, l'une des approches possibles est fondée sur le *Bayes Naïf*, où on calcule les distributions de $Pr(\gamma|M)$ et $Pr(\gamma|U)$ sous l'hypothèse d'indépendance conditionnelle, c'est-à-dire, où les probabilités $Pr(\gamma^i|M)$ (resp. $Pr(\gamma^i|U)$) et $Pr(\gamma^j|M)$ (resp. $Pr(\gamma^j|U)$) sont indépendantes si $i \neq j$. Dans ce cas, les probabilités sont exprimées comme suit :

$$Pr(\gamma|M) = \prod_{i=1}^{k} Pr(\gamma^i|M)$$

$$Pr(\gamma|U) = \prod_{i=1}^{k} Pr(\gamma^i|U)$$

où k représente le nombre d'attributs décrivant les références. Les valeurs de $Pr(\gamma^i|M)$ et $Pr(\gamma^i|U)$ peuvent être calculées en utilisant un échantillon de paires de références étiqueté manuellement. Cependant, ce modèle probabiliste peut être utilisé sans l'exploitation de données étiquetées. Jaro [Jar89] a, par exemple, utilisé un modèle binaire pour les valeurs γ^i (i.e. si les valeurs du i-ème attribut s'apparient alors $\gamma^i = 1$ sinon $\gamma^i = 0$). Il a suggéré l'utilisation de l'algorithme d'apprentissage non supervisé EM–Espérance et Maximisation [DLR77] pour calculer les probabilités $Pr(\gamma^i|M)$.

Quand l'hypothèse d'indépendance conditionnelle ne peut plus être vérifiée, Winkler [WEW93] a suggéré l'utilisation d'une variante de l'algorithme EM pour estimer $Pr(\gamma|M)$ et $Pr(\gamma|U)$.

Winkler [Win02] a constaté, qu'en général, l'algorithme non-supervisé EM n'est efficace que lorsque certaines conditions sont vérifiées, telles que :

1. Les données contiennent un pourcentage de réconciliation relativement grand (plus de 5%),
2. Les valeurs des ratios correspondant aux paires de références réconciliées sont "bien séparées" des deux autres classes U et P,
3. Le taux des erreurs typographiques est faible,
4. Les données contiennent un nombre suffisant de références redondantes pour pouvoir appréhender les erreurs dans les valeurs des attributs des références.

Belin et Rubin [TRB95] ont introduit une méthode automatique pour l'estimation du taux d'erreur pour des seuils T_μ et T_λ donnés. Leur méthode utilise des données d'entraînement et elle n'est efficace que lorsque les valeurs du ratio pour les paires réconciliées et les paires non réconciliées sont bien séparées.

Dans [NMTM00], un réseau Bayesien est utilisé pour modèliser les règles de décision. Les auteurs ont observé que l'utilisation des réseaux Bayesiens donne la possibilité de combiner l'exploitation de données *étiquetées* et de données *non étiquetées* pendant la procédure d'apprentissage, mais, dans le cas où on n'utilise que des données *non-étiquetées*, les règles de décision sont très pauvres.

Méthodes probabilistes fondées sur le coût. Dans cette section nous décrivons une méthode probabiliste de réconciliation de références [VME03] qui cherche à minimiser le coût associé aux erreurs de classification. Ces erreurs correspondent aux *fausses réconciliations* et aux *fausses non réconciliations* dans la Figure 4.3.

Comme nous l'avons présenté précédemment, Fellegi et Sunter ont prouvé dans leur modèle que la règle de décision proposée est optimale si elle respecte la minimisation de la taille de la zone des réconciliations possibles (voir la zone $A2$ dans la Figure 4.3) qui nécessite une vérification manuelle et si elle permet de minimiser le nombre des *fausses réconciliations* et des *fausses non réconciliations*.

À la différence de Fellegi et Sunter où les décisions sont fondées sur le taux d'erreur, l'approche proposée par [VME03] est fondée sur la minimisation du coût nécessaire pour effectuer les décisions de réconciliation et de non réconciliation. Le coût des décisions représente : (i) le coût du processus de décision, incluant, par exemple, le nombre de comparaisons nécessaires pour la décision et (ii) le coût de l'impact de certaines décisions.

Un coût est associé à chaque type de décision possible. Ces coûts doivent être définis en fonction du domaine et du modèle probabiliste utilisé. De plus, les probabilités de réconciliation des références comparées sont aussi considérées comme connues a priori. Étant données ces entrées, le modèle produit comme résultat la règle de décision sur l'appartenance des paires de références à M ou à U.

La Figure 4.4 montre que dans le modèle, les coûts c_{ij} considérés expriment les coûts de la prise d'une décision d'assigner une paire à A_i[12] quand les paires de références comparées ont un statut réel de réconciliation M ou U.

Coût	Décision	Statut réel
c_{10}	A_1	M
c_{11}	A_1	U
c_{20}	A_2	M
c_{21}	A_2	U
c_{30}	A_3	M
c_{31}	A_3	U

FIG. 4.4 – *Les coûts correspondant aux différentes décisions*

La classification des paires de références de l'espace de décision composé de l'union de A_1, A_2 et A_3 est effectuée de manière à ce que le coût des décisions c_m soit minimisé. Le coût c_m à minimiser par la méthode est exprimé comme suit :

$$c_m = c_{10} * Pr(d = A_1, r = M) + c_{11} * Pr(d = A_1, r = U) +$$

$$c_{20} * Pr(d = A_2, r = M) + c_{21} * Pr(d = A_2, r = U) +$$

$$c_{30} * Pr(d = A_3, r = M) + c_{31} * Pr(d = A_3, r = U)$$

où d représente la classe prédite pour une paire de références et r le statut réel de réconciliation de la paire de références. Un ensemble d'équations est obtenu après quelques transformations de c_m en appliquant, par exemple, le théorème de Bayes (pour plus de détails, voir [VME03]).

Dans les systèmes d'intégration de données, la notion de coût peut être considérée de telle sorte que le coût des fausses réconciliations soit plus important que celui des fausses non réconciliations. En effet, l'impact de la fusion de données qui ne réfèrent pas à la même entité du monde réel pourrait être plus pénalisant pour le système que celui de la non détection de redondances dans les données.

Méthodes fondées sur des combinaisons de scores de similarité.

Pour le problème de réconciliation de références certaines méthodes s'appuient sur des modèles probabilistes telles que Fellegi et Sunter et ses extensions, et d'autres méthodes s'appuient sur des fonctions de combinaison de scores de similarité. En effet, en utilisant des mesures de

[12] On rappelle que les trois zones A_1, A_2 et A_3 représentent respectivement les paires réconciliées M, les paires candidates à la réconciliation P et les paires non réconciliées U.

similarité et des seuils appropriés on peut calculer un score de similarité pour les paires de références et ainsi décider de la (non) réconciliation des paires de références.

Debabrata et al. [DSD98a] ont proposé une méthode de réconciliation de références, dans une base de données relationnelle, fondée sur les mesures de similarité des valeurs des attributs. Ils proposent de calculer la similarité de chaque paire de références par une *moyenne pondérée* des différents scores de similarité des valeurs des *attributs communs*. De manière plus formelle, soient D_1 et D_2 deux bases de données, et soient $R_1 \in D_1$ et $R_2 \in D_2$ deux relations représentant des données du même type du monde réel (e.g. personne, musée, ville).

Posons A l'ensemble des *attributs communs* entre les relations R_1 et R_2. On considère l'ensemble d'attributs $\{Y_1, Y_2, \ldots, Y_K\} \subset A$ tels que : $(k \neq k') \Rightarrow \neg(Y_k \rightarrow Y'_k)$, $\forall k, \forall k' \in \{1, 2, \ldots, K\}$. En d'autre termes, si un attribut commun $Y_{k'}$ (e.g. *NomVille*) est déterminé fonctionnellement par un autre attribut commun Y_k (e.g. *CodePostal*) alors les valeurs de cet attribut commun $Y_{k'}$ ne sont pas considérées dans le calcul de similarité. Soit sim_{ijk} le score de similarité entre les valeurs y_{aik} et y_{bjk} du k-ème attribut commun entre les références de R_1 et R_2, calculé par une fonction de similarité normalisée. Le score de similarité SIM_{ij} d'une paire de références a_i et b_j est défini en fonction des scores des valeurs de leurs attributs communs :

$$SIM_{ij} = \sum_{k=1}^{K}(w_k * sim_{ijk}), \quad avec \quad (\sum_{k=1}^{K} w_k) = 1$$

Pour chaque type de valeurs d'attributs (e.g. numérique, booléennes, dates, textuelles), une fonction appropriée est donnée en paramètre. Par exemple, pour les valeurs textuelles, des mesures de similarité telles que celles présentées en Section 4.2 sont appliquées et, pour les valeurs numériques, le score de similarité peut être calculé par la différence absolue des valeurs, normalisé sur une échelle de [0..1], i.e. $sim(v_1, v_2) = \frac{|v_1 - v_2|}{\max(v_1, v_2)}$, avec v_1 et v_2 deux valeurs numériques.

La pondération w_k représente l'importance relative de l'attribut Y_k dans le calcul de similarité. Ces pondérations sont obtenues grâce à des préférences saisies par l'utilisateur sous forme de comparaisons par paires d'attributs.

Le problème réside dans l'estimation des pondérations qui est effectuée, dans ce type de méthodes, à partir d'informations saisies par l'utilisateur. Cette difficulté est similaire à celle des méthodes probabilistes qui sont dépendantes des moyens d'estimer les paramètres de leur modèle.

Guha et al [GKMS04], au lieu d'utiliser une fonction de combinaison des différents scores de similarité, proposent de créer une mesure fondée sur le principe de fusion de *listes ordonnées (ranked)*. Ils se posent le problème de trouver, dans une base de données, l'ensemble de références les plus similaires à une *référence requête*[13] donnée. Leur idée principale est que lorsqu'on compare les références par rapport à un seul attribut, l'algorithme peut facilement trouver les meilleurs appariements et les ordonner selon leur score de similarité. Toutefois, le même principe ne peut pas être trivialement généralisé à une comparaison des références sur plusieurs attributs. En effet, pour une comparaison sur m attributs, on obtient m listes ordonnées de références pour chaque référence.

Soit R une relation de taille n, ayant m attributs et $r_1, \ldots, r_n$ représentent les références de la relation R. Une *liste ordonnée* de R est une liste $\tau = \{x_1 \preceq x_2 \preceq \ldots \preceq x_n\}$, où $x_i \in R$ et $\preceq$ est une relation d'ordre sur R. Un tel rangement peut être obtenu en appliquant une fonction

[13] Cette *référence requête* peut être représentée dans une autre base de données.

de similarité appropriée entre une valeur d'un attribut d'une référence dite *référence requête* et les valeurs de cet attribut des références dans R. La position d'une référence r_i dans une liste ordonnée τ est dénotée par $\tau(r_i)$ (la première référence de la liste a la position la plus faible). Posons r_q une référence requête avec m attributs et $\tau_1, \ldots, \tau_m$ les différentes listes ordonnées obtenues après l'application de fonctions de mesure de similarité entre chaque valeur des attributs de r_q et les valeurs de l'attribut correspondant de la relation R. Les scores de similarité obtenus pour chaque référence, en considérant un attribut à la fois, sont ensuite ordonnés par ordre décroissant, i.e. du plus similaire au moins similaire. En répétant ce principe sur l'ensemble des attributs, on obtient, pour chaque référence, m *listes ordonnées* de références, i.e. une liste pour chaque attribut $\tau_1, \ldots, \tau_m$.

Relation R :

IdRéférence	**NomMusee**	**Adresse**
r_1	Arts Premiers	206, rue de l'Université 75007 Paris
r_2	Louvre	99, rue Rivoli, 75001 Paris
r_3	Cluny	6, place Paul Painlevé, 75005 Paris

Requête Q :

$Q_{NomMusee}$	$Q_{Adresse}$
Quai Branly	206, rue de l'Université 75007 Paris

Listes ordonnées de R :

τ_1
r_3 (0.44)
r_2 (0.40)
r_1 (0.34)

τ_2
r_1 (1.00)
r_2 (0.58)
r_3 (0.29)

FIG. 4.5 – *Exemple de listes ordonnées d'une relation R par rapport à une requête Q donnée.*

Par exemple, dans la figure 4.5, pour l'ensemble des références r_1, r_2 et r_3 de la relation R et la référence requête Q, on obtient deux listes ordonnées τ_1, pour une comparaison sur l'attribut $NomMusee$, et τ_2, pour une comparaison sur l'attribut $Adresse$. Chacune de ces deux listes est obtenue en ordonnant les scores de similarité calculés[14] entre la valeur de $Q_{NomMusee}$ (resp. $Q_{Adresse}$) et les valeurs de l'attribut $NomMusee$ (resp. Adresse) de la relation R. On remarque que les listes ordonnées obtenues représentent des ordonnancements différents selon que l'on considère l'attribut *NomMusee* ou l'attribut *Adresse*. Par exemple, la référence r_1 apparaît en premier dans la liste τ_2 alors qu'elle est en dernier dans la liste τ_1. Pour obtenir un ordre des références le plus représentatif possible, on a besoin d'utiliser une fonction de combinaison des positions des références qui considère toutes les listes ordonnées. Guha et al [GKMS04] utilisent la distance *Footrule* proposée dans [PD77] pour synthétiser de nouveaux ordres des références de la relation R. La distance *Footrule* entre deux listes ordonnées correspond à la somme des différences absolues des positions des différentes références de R dans les deux *listes ordonnées*. Plus formellement, soient σ et τ deux *listes ordonnées* de R. La distance *Footrule* entre ces deux listes est définie comme suit :

$$F(\sigma, \tau) = \sum_{i=1}^{n} |\sigma(r_i) - \tau(r_i)|.$$

[14] En utilisant l'algorithme de calcul de similarité Monge-Elkan (voir en Section 4.2)

Par exemple, $F(\tau_1, \tau_2) = (2 + 0 + 2) = 4$

La mesure peut être généralisée pour calculer la distance entre une liste ordonnée σ et un ensemble de listes ordonnées $\tau_1, \ldots, \tau_m$:

$$F(\sigma, \tau_1, \ldots, \tau_m) = \sum_{i=1}^{m} F(\sigma, \tau_i).$$

Le but final de cette méthode est de créer une liste ordonnée σ des références de R représentant les *k-premières* (k est un paramètre fixé par l'utilisateur) références les plus similaires à une référence requête r_q donnée. Il s'agit des *k–premières* références d'une liste ordonnée σ de R qui rend $F(\sigma, \tau_1, \ldots, \tau_m)$ minimale. Chaque référence r_i dans σ est annotée par un coût, appelé *coût de rangement*, qui est obtenu comme résultat de cette minimisation. Une référence r_i à la position j, $1 \leq j \leq k$, dans σ a le coût suivant : $\sigma^{rg}(r_i, j) = \sum_{l=1}^{m} |\tau_l(r_i) - j|$. Par exemple, dans la Figure 4.4, si on considère que σ suit l'ordre de la liste τ_1 alors $\sigma^{rg}(r_3, 1) = |3 - 1| = 2$.

La solution du problème de sélection des *k–premières* références est un ensemble de taille k de triplets $(r_i, j, \sigma^{rg}(r_i, j))$, où j est la position de la référence r_i dans le liste résultat σ, telle que pour toutes les positions j' de la référence r_i, la somme $\sum_{j'=1}^{k} \sigma^{rg}(r_i, j')$ des coûts de rangement de la référence r_i est minimisée.

Pour résoudre ce problème de minimisation, Guha et al. ont adapté une méthode fondée sur l'*algorithme Hongrois* [AMO93], un algorithme s'appuyant sur la théorie des graphes.

4.3.2 Méthodes globales de réconciliation de références

Les approches locales présentées ci-dessus considèrent chaque référence comme une entité complètement indépendante des autres entités et ignorent le fait que les données sont souvent représentées dans des schémas qui permettent d'exprimer des relations entre les données. Par exemple, dans une base de données relationnelle, les données sont stockées dans différentes tables liées par des relations (clé-étrangères). De même des données décrites en RDF forment un graphe où les ressources (URIs) sont reliées entre elles. Des approches récentes de réconciliation de références [ACG02, BG04, BG06, DHM05, KMC05] exploitent la description des références en terme d'attributs, mais également les relations liant les différentes références afin de décider de leur réconciliation ou de leur non réconciliation. Par exemple, lorsqu'on cherche à réconcilier "*Le Musée du Louvre*" avec "*Le Louvre*", on est encore plus sûr de leur réconciliation s'ils contiennent tous les deux la peinture "*La Joconde*" et si celle-ci a été réalisée par le même peintre "*Leonardo Da Vinci*". L'exploitation des relations entre références permet aux méthodes de réconciliation de réduire à la fois le nombre de fausses réconciliations (Faux Positifs) mais aussi le nombre de fausses non réconciliations (Faux Négatifs).

Parmi ces méthodes globales nous en distinguons deux types :

1. **Méthodes globales non itératives de réconciliation de références.** Il s'agit des approches de réconciliation de références [ACG02, KM06] qui exploitent les relations entre les références et où le calcul est fait une seule fois pour une paire de références. En d'autres termes, aucune mise à jour n'est effectuée sur les scores de similarité (ou probabilité de (non) réconciliation) ni sur les décisions de (non) réconciliation une fois que ce calcul est réalisé.

2. **Méthodes globales itératives de réconciliation de références.**

Il s'agit des approches de réconciliation de références (e.g. [BG04, DHM05]) qui exploitent les relations entre les références de manière itérative, en parcourant plusieurs fois les références. Toute modification concernant les scores de similarité (ou les probabilités) et concernant les décisions de (non) réconciliation, est propagée à d'autres références. Ceci est itéré jusqu'à ce qu'un point fixe soit atteint : il n'y a alors plus de mise à jour possible des scores de similarité et/ou plus aucune décision de (non) réconciliation ne peut être inférée.

Méthodes globales non itératives de réconciliation de références

Des travaux se sont intéressés à l'exploitation des relations entre les données de manière non-itérative pour la réconciliation de références de différents types, comme par exemple, des pays, des villes, des personnes.

Nous présentons maintenant plus en détail la méthode nommée *Delphi* proposée par Ananthakrishna et al. [ACG02]. Ils ont introduit une méthode relationnelle de réconciliation de références dans des entrepôts de données conçus selon un modèle de données hiérarchique, i.e. en étoile, où chaque référence est décrite par un attribut. Pour la réconciliation de références, un score de similarité est calculé en combinant la similarité calculée en terme d'attributs textuels avec la similarité entre les références des relations qui leur sont reliées par les clés étrangères.

Les hiérarchies dimensionnelles sont particulièrement utilisées dans les entrepôts de données dont le schéma est en étoile. Dans ce type de schéma, les hiérarchies dimensionnelles forment une chaîne de relations composée d'une relation représentant la table de faits, et d'une ou plusieurs relations représentant les dimensions d'intérêt pour les analyses multi-dimensionnelles. Dans la suite, nous utiliserons le terme *hiérarchie relationnelle* pour se référer à la *hiérarchie dimensionnelle*.

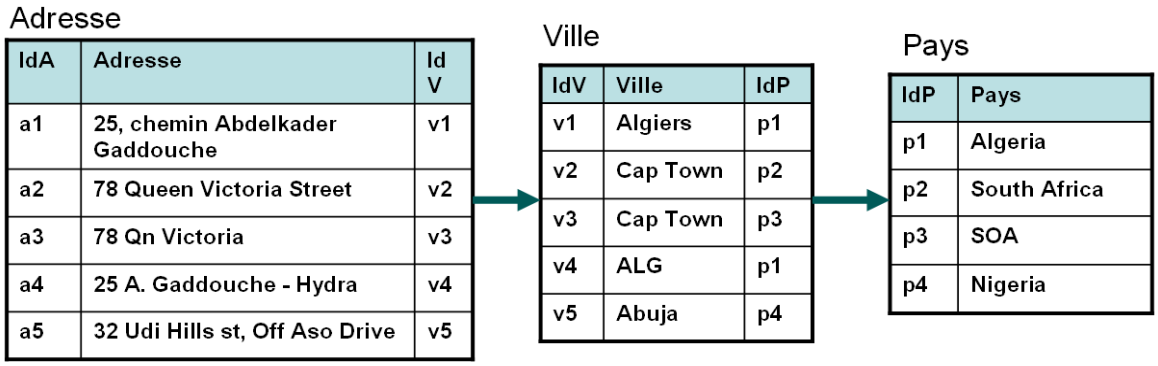

FIG. 4.6 – Exemple de trois relations formant une hiérarchie relationnelle.

Un exemple de hiérarchie relationnelle est montré en Figure 4.6 où une table de faits représentant des références redondantes d'ambassades de France à l'étranger est reliée à une hiérarchie relationnelle représentant leur localisation : la relation *Adresse*, leur *Ville* associée et le *Pays*. Étant donnée une paire de relations adjacentes dans la hiérarchie, on qualifie de *parente* la relation contenant la clé étrangère, et de *enfant* la relation contenant la clé primaire. La relation *Pays* est parente de la relation *Ville* qui est elle-même parente de la relation *Adresse*.

Dans une hiérarchie relationnelle, à une référence d'une relation parente on peut associer

plusieurs références d'une relation enfant. Par exemple, à un pays on peut associer plusieurs villes et à une ville on peut associer plusieurs adresses.

L'idée principale de Ananthakrishna et al. est d'exploiter la structure hiérarchique des relations, en s'appuyant à la fois sur la similarité des descriptions locales des références, i.e. les valeurs d'attributs, et sur une similarité de *co-occurrence* correspondant à une similarité plus globale. À titre d'exemple, considérons les références de la relation *Pays* de la Figure 4.6. Si on se limite uniquement à la similarité locale des références, en utilisant par exemple une mesure d'édition entre les noms des pays, on peut conclure que (*Algeria*, *Nigeria*) sont des références de pays redondantes et que (*SOA*, *South Africa*) sont des références disjointes. La similarité de co-occurrence entre deux références est mesurée sur l'ensemble représentant l'intersection des ensembles de références qui leur sont associés dans les relations enfant. Par exemple, pour deux références de pays, plus la taille de l'intersection des ensembles de villes qui leur sont associés est grande plus la similarité de co-occurrence entre ces deux pays est grande. Dans l'exemple de la Figure 4.6, pour la paire (*Algeria*, *Nigeria*) aucune référence enfant n'est commune. Ce type d'information est exploité pour éviter de fausses réconciliations.

Plus formellement la similarité de co-occurrence est calculée de la manière suivante. Soit $G = \{r_1, \ldots, r_n\}$ l'ensemble des références d'une relation R_i.

La similarité de co-occurrence d'une paire de références (r, r') notée Sce est calculée sur les ensembles de références enfant[15] des références r et r'. On note $EE(r)$ l'ensemble des références enfant d'une référence r. Dans la Figure 4.6, la similarité de co-occurrence Sce des références $(p1, p2)$ est calculée sur les ensembles de références enfant $EE(p1) =$ {"Algiers", "ALG"} et $EE(p2) =$ {"Cap Town"}. Soit $B_c(G)$ le multi-ensembles des références enfant de R_i et appartenant à la relation R_{i-1}. Dans l'exemple $B_c(G) =$ {{Algiers, ALG}, {Cap Town}, {Cap Town}, {Abuja}}

La similarité de co-occurrence $Sce(r, r')$ entre deux références r et r' de G par rapport à $B_c(G)$ utilise IDF pour tenir compte de la fréquence des descriptions textuelles des références de $B_c(G)$ [ACG02] :

$$Sce(r, r') = \frac{IDF(EE(r) \cap EE(r'))}{IDF(EE(r))}$$

Par exemple, la similarité $Sce(p2, p3)$ est de 1.0, car les seules références enfant $v2$ et $v3$ des références $p2$ et $p3$ ont exactement la même valeur "Cap Town".

Pour calculer la similarité textuelle, Ananthakrishna et al. s'inspire de la mesure de similarité cosinus (voir en Section 4.2) notée St. Soit $TS(r)$ l'ensemble de chaînes atomiques (tokens) dans une description de référence r et soit $B_t(G)$ le multi-ensembles de tous les tokens apparaissant dans les descriptions des références dans G. La similarité textuelle de deux références r et r' est obtenue par la mesure cosinus appliquée aux ensembles de tokens $TS(r)$ et $TS(r')$ de G et en fonction des pondérations TF (calculées sur G) et IDF (calculées sur $B_t(G)$).

Une fois que les scores de similarité textuelle et de co-occurrence ont été calculés, pour décider de la réconciliation des paires de références, une fonction de combinaison de ces scores de similarité est définie. Cette fonction de combinaison pondérée renvoie une valeur 1 si la paire est réconciliée et -1 sinon. Pour chaque référence r de R_i, les poids w_t et w_c utilisés dans la fonction de combinaison sont les suivants : $w_t = IDF(TS(r))$ et $w_c = IDF(EE(r))$. Soient $seuilSt$ un seuil fixé pour la similarité textuelle et $seuilSce$ un seuil fixé pour la similarité de co-occurrence. Soit $pos : \mathcal{R} \longrightarrow \{1, -1\}$ une fonction définie comme suit : $pos(x) = 1$, si $x > 0$, et -1 sinon. La

[15] Il s'agit des références liées aux références r et r' par une clé-étrangère dans la relation enfant.

fonction de combinaison pondérée des similarités d'une paire de références (r, r') est donnée par l'expression suivante :

$$pos(w_t * pos(St(r, r') - seuilSt) + w_c * pos(Sce(r, r') - seuilSce)).$$

Par exemple, pour la paire de références $(p1, p4)$ de la relation *Pays* de la Figure 4.6, supposons que les pondérations w_t et w_c soient identiques et égales à 0.5 et supposons que les seuils $seuilSt$ et $seuilSce$ soient également identiques et égaux à 0.7. Nous avons donc la combinaison suivante : $pos(0.5 * pos(0 - 0.7) + 0.5 * pos(1 - 0.7)) = pos(-1 + 1) = -1$. Ainsi, nous décidons de la non réconciliation des deux pays $p1$ et $p4$, malgré leur similarité textuelle.

Nous montrons dans la Figure 4.7 les différentes étapes de l'algorithme de réconciliation de références *Delphi*.

1. Traiter la plus haute ($\top$) relation de la hiérarchie ,
2. Grouper en différents clusters les références des relations plus spécifiques,
3. Éliminer, de chaque cluster, les références qui, selon un score de similarité locale, sont potentiellement distinctes ,
4. Comparer les paires de références dans chaque cluster en utilisant les deux fonctions de comparaison :
   - Similarité textuelle entre deux références ;
   - Similarité de co-occurrence sur les ensembles de références des relations enfant.
5. Décider d'une combinaison appropriée des résultats des deux scores selon un seuil ou des seuils donnés,
6. Mise à jour dynamique des seuils,
7. Descendre d'un niveau dans la hiérarchie.

FIG. 4.7 – *Les étapes de l'algorithme Delphi*

Les étapes 2 et 3 de l'algorithme sont des étapes de clustering et de filtrage des paires de références candidates qui permettent d'améliorer l'efficacité du parcours de la hiérarchie en réduisant le nombre de comparaisons. Ainsi, les paires de références qui ne peuvent pas être réconciliées sont éliminées de l'espace de réconciliation. L'étape de mise à jour dynamique des seuils a pour but de les adapter aux caractéristiques structurelles des différents clusters ; le nombre de valeurs distinctes des domaines de définition des références peut varier d'un cluster à un autre ainsi que la taille des valeurs (e.g. les noms des villes d'un pays peuvent être particulièrement longs). Ces différentes caractéristiques influent sur les valeurs des seuils $seuilSt$ et $seuilSce$.

Dans la méthode *Delphi*, pour la réconciliation de références d'une relation R_i, la similarité de co-occurrence est calculée en exploitant uniquement les références de la relation R_{i-1}. Cette méthode pourrait facilement être étendue et exploiter non seulement les relations adjacentes directes, mais toutes les relations enfant d'une relation parente R_j (avec $j < i$). En effet, une telle extension améliorerait encore plus les résultats de la réconciliation de références. Cependant, les auteurs estiment que cela rendrait la méthode inefficace, car, le nombre de références croît de façon importante en fonction du niveau dans la hiérarchie.

Un autre travail qui exploite de manière non itérative les relations pour améliorer la qualité des résultats de la réconciliation de références est celui de Kalashnikov et al. [KM06]. Les auteurs proposent une méthode qui combine des techniques de mesure de similarité des valeurs des attributs avec des techniques qui permettent d'exploiter des relations dans une base de données relationnelle. L'objectif est de réconcilier des références dites ambiguës d'une table avec des références d'entités d'une table qui, elle, ne comporte pas de redondances.

Pour analyser les relations, ils considèrent la base de données comme un graphe *entité–relation*, et en utilisant des techniques de théorie des graphes ils analysent les chemins existant entre les nœuds du graphe ce qui correspond à l'analyse de chaînes de relations entre les entités.

Nous allons illustrer le problème de réconciliation de références considéré par Kalashnikov et al. [KM06] dans le domaine des publications scientifiques. Considérons une base de données d'auteurs et d'articles, où les auteurs sont représentés en utilisant les attributs < *id, nomAuteur, affiliation*> et les articles sont représentés par les attributs <id, *titre*, $EcritPar_1, \ldots, EcritPar_N$>. En Figure 4.8, nous présentons une table d'auteurs sans redondances et une table de publications où se pose le problème de réconciliation des références d'auteurs des articles. Il s'agit par exemple de trouver à quelle entité *auteur* de la table Auteur, la référence "D. White" de l'article $P2$ se réfère.

Table : Auteur (réconciliés)		**Table : Publication (à réconcilier)**
<A1, Dave White, Intel>		< P1, Databases . . . , John Black, Don White>
<A2, Don White, CMU>		< P2, Multimedia . . . , Sue Grey, **D. White** >
<A3, Susan Grey, MIT>		< P3, Title3 . . ., Dave White>
<A4, John Black, MIT>		< P4, Title5 . . ., Don White, Jœ Brown>
<A5, Jœ Brown, null>		< P5, Title6 . . ., Jœ Brown, Liz Pink>
<A6, Liz Pink, null>		< P6, Title7 . . . , Liz Pink, **D. White** >

FIG. 4.8 – *Exemple de tables sans redondances et avec redondances*

Lors d'une première étape, des mesures de similarité sur les valeurs des attributs sont utilisées pour définir, pour chaque référence et relation, un ensemble d'entités candidates à la réconciliation. Par exemple, pour les références d'auteurs de la table publication de la Figure 4.8, on peut comparer les valeurs de l'attribut *EcritPar* des articles avec celles de l'attribut *NomAuteur* de la table *Auteur*. Ceci va permettre de résoudre le problème de réconciliation de références pour certaines références d'auteurs. Néanmoins, les références "$D.\ White$" qui apparaissent dans les descriptions des articles $P2$ et $P6$ restent ambiguës. En effet, ces références peuvent être associées soit à l'auteur "$Dave\ White$" ou à l'auteur"$Don\ White$". Dans un tel cas, des relations entre entités, comme *affiliation* et *co-auteur*, peuvent être analysées et exploitées afin de lever l'ambiguïté sur les entités représentées par ces références. Par exemple, on remarque que, dans l'article $P1$, "$Don\ White$" est co-auteur avec "$John\ Black$" qui est au MIT, alors que l'auteur "$Dave\ White$" n'a été co-auteur d'aucun chercheur du MIT. Comme le co-auteur de "$D.\ White$" dans $P2$ est "$Susan\ Grey$" du MIT alors il est plus probable que l'auteur "$D.\ White$" dans $P2$ soit "$Don\ White$" au lieu de "$Dave\ White$". Cette observation peut être utilisée pour lever l'ambiguïté concernant l'auteur "$D.\ White$" de l'article $P2$.

Pour analyser les relations entre entités, Kalashnikov et al. proposent l'utilisation du "Principe d'Attraction de Contexte (CAP)" qui suppose qu'une référence ambiguë est connectée de façon plus forte à l'entité à laquelle elle correspond vraiment qu'au reste des entités candidates. Ils traduisent ce principe dans un système d'équations non linéaire qui relie toutes les *forces de*

connexion dans le graphe d'entités et celles entre les références et leurs entités possibles. La solution à ce système d'équations donne les *forces de connexion* et les plus élevées déterminent les choix des entités représentées par les références ambiguës.

Définition – Principe d'Attraction du Contexte (CAP). Si une référence r appartenant au contexte d'une entité x se réfère réellement à une entité y_j et si la description de la référence r s'apparie avec plusieurs entités $y_1, \ldots, y_j, \ldots, y_N$, alors l'entité x et y_j sont probablement plus fortement connectées l'une à l'autre via des chaînes de relations que x et y_l, avec $(l = 1, 2, \ldots, N; l \neq j)$.

Par exemple, la description de "D. White" s'apparie à la fois avec "Dave White" et "Don White". Le principe CAP indique alors que si la référence "D. White" appartenant au contexte de l'article $P2$ se réfère réellement à l'entité "Don White" alors $P2$ et "Don White" sont probablement plus fortement connectés que ne le sont $P2$ et "Dave White".

Plus formellement, Kalashnikov et al. considèrent les données comme un graphe G où, les nœuds représentent les entités (articles, auteurs, organisations, etc.) et les arêtes représentent les relations entre les entités découvertes lors de la première étape. Par exemple, l'article $P1$ a été écrit par l'auteur $A2$ "*Don White*". Dans la description d'une entité x_i, une référence vers une entité y_j par la relation r_k est notée $x_i.r_k$. Par exemple, la référence du k-ème auteur de l'article $P1$ est notée $P1.EcritPar_k$. Á toute relation, est associé un poids w dans [0..1] qui représente le degré de certitude dans l'existence de la relation entre deux entités. Par exemple, si on est à 100 % sûr que "*Don White*" a écrit l'article $P1$ alors on associe un poids $w = 1$ à la relation $EcritPar2$ entre l'entité $P1$ et l'entité $A2$ de la table Auteur. Kalashnikov et al. considèrent que chaque référence $x_i.r_k$ se réfère à une seule entité parmi les entités $X = \{x_1, x_2, \ldots, x_N\}$ représentées dans la base de données. L'ensemble des entités possibles $\{y_1, y_2, \ldots, y_N\}$ associé à une référence ambiguë $x_i.r_k$ est noté $CS[x_i.r_k]$. Par exemple, $CS[P2.EcritPar2] = \{A1, A2\}$.

La notion de *force de connexion* entre deux entités x_i et y_j, notée $c(x_i, y_j)$ reflète la force du lien existant entre ces deux entités dans le graphe G. La valeur de la force de connexion pourrait être calculée en fonction de la longueur du plus court chemin entre les deux entités. Cependant, un calcul plus sophistiqué a été proposé par les auteurs afin de prendre en compte les nœuds représentant un ensemble d'entités possibles pour une référence r. Soient deux chemins p_a et p_b entre deux entités x et y dans le graphe G. Si le chemin p_a contient un nœud représentant un ensemble d'entités possibles pour une référence donnée et si p_b ne contient aucun nœud de ce type alors la force de connexion entre les entités x et y via le chemin p_b doit être plus élevée que celle via le chemin p_a.

La méthode de réconciliation de références proposée par Kalashnikov et al. [KM06] suit les quatre étapes suivantes :

1. *Expression du calcul des forces de connexion.* Pour chaque référence $x_i.r_k$, la force de connexion $c(x_i, y_j)$ est exprimée pour chaque $y_j \in CS[x_i.r_k]$ par l'équation : $c(x_i, y_j) = g_{ij}(\bar{w})$. Cette équation est fonction de $\bar{w}$ qui représente l'ensemble des poids des relations appartenant aux chemins reliant l'entité x_i aux entités y_j de $CS[x_i.r_k]$. La fonction g_{ij} est la somme des forces de connexion de tous les chemins reliant l'entité x_i aux entités y_j de longueur inférieure à un paramètre L.
2. *Determiner les équations pour les poids des relations dans les ensembles d'entités candidates.* Les auteurs utilisent les équations de la première étape pour exprimer le calcul des

poids en fonction des poids[16] des autres relations. Par exemple, le poids de la relation $EcritPar$ peut être calculé en fonction des poids des relations $co-Auteur$ et $Affiliation$ des entités auteurs.

3. *Calculer les poids.* L'ensemble des équations résultant de la seconde étape est résolu en utilisant une méthode itérative qui renvoie des solutions approchées. Dans cette méthode itérative on initialise les poids w_j à $\frac{1}{|CS[x_i.r_k]|}$ pour toute référence $x_i.r_k$. Ensuite, une étape d'itération est appliquée sur les pondérations w_j pour lesquelles les valeurs varient entre $\frac{1}{|CS[x_i.r_k]|}$ et 1. Le calcul itératif des poids se termine quand la différence des valeurs w_j d'une itération à une autre est négligeable.
4. *Résoudre la réconciliation de références.* Il s'agit de décider, pour chaque référence $x_i.r_k$, quelle entité parmi les entités $Y = \{y_1, y_2, \ldots, y_N\}$ appartenant à $CS[x_i.r_k]$, représente effectivement cette référence $x_i.r_k$. Les auteurs proposent de choisir l'entité y_j qui possède le plus grand poids w_j parmi tous les poids $w_1, w_2, \ldots, w_N$. Ils proposent une autre alternative qui consiste à combiner les scores de similarité obtenus pour les valeurs des attributs en tenant compte des différents poids w_j par une somme pondérée.

Dans cette méthode, les auteurs ont posé le problème de réconciliation de références en terme de calcul de degré de certitude de l'existence de relations entre deux références, e.g. le degré de certitude dans l'existence de la relation $EcritPar$ entre une publication et un auteur. La similarité entre références découle des poids des relations calculés de manière itérative.

Méthodes globales itératives de réconciliation de références

De récents travaux [DHM05, BG06] se sont intéressés à une exploitation itérative des relations entre les références. Dong et al. [DHM05] ont proposé une méthode itérative de réconciliation de références, de types multiples, par propagation des scores de similarité et des décisions de réconciliation dans un graphe de dépendances. Bhattacharya et Getoor [BG06] ont proposé une méthode de clustering relationnel pour une réconciliation de références itérative dans un graphe de références.

Réconciliation de références dans un graphe de dépendances. La méthode de réconciliation de références proposée par Dong et al. [DHM05] est une méthode qui s'appuie sur un *graphe de dépendances* pour traiter de manière itérative le problème de réconciliation de références. Cette méthode est capable d'appréhender ce problème, dans un modèle de données complexe où, les données sont de différents types (e.g. personne, article, conférence) et où, les informations peuvent être incomplètes, c'est-à-dire où des attributs peuvent ne pas être renseignés. Cette méthode exploite les relations existant entre les références pour prendre les décisions de réconciliation et propager les informations concernant les modifications des scores de similarité et les nouvelles décisions de réconciliation, entre les différentes paires de références. Par exemple, quand une décision de réconciliation de deux articles est prise, alors le score de similarité de leurs auteurs peut être augmenté et cette augmentation permet d'augmenter le score de similarité d'autres articles et ainsi de suite. De plus, afin de pallier le problème du manque d'information disponible[17] lors de la comparaison des références, après toute décision de récon-

[16] La somme des poids des relations impliquant des entités candidates doit être égale à 1.

[17] Dans des applications comme les PIM – les systèmes de gestion d'informations personnelles – peu d'informations sont disponibles pour décrire les références. Par exemple, une référence d'une personne est décrite uniquement par son *nom* et une autre décrite uniquement par son *adresse e-mail*.

ciliation, les auteurs appliquent une procédure d'enrichissement des références qui permet de compléter leurs informations.

Afin de pouvoir propager les informations, l'approche repose sur une structure de graphe appelé *graphe de dépendances*. Ce dernier contient un ensemble de nœuds représentant les paires de références et les paires de valeurs. A chacun des nœuds représentant une paire de valeurs, on associe un score de similarité obtenu par une mesure de similarité entre valeurs (telle que celles présentée en Section 4.2). A chacun des nœuds représentant une paire de références, un score de similarité initialisé à 0 est associé. Les arcs du graphe représentent les dépendances entre les scores de similarité. Certains arcs lient des paires de valeurs à des paires de références et d'autres lient des paires de références à d'autres paires de références.

Le graphe de dépendances est construit de façon à ce que les nœuds des paires de références ne représentent que les paires qui peuvent potentiellement référer à la même entité du monde réel, c'est-à-dire, celles pour lesquelles le score de similarité des valeurs des attributs communs dépasse un seuil donné. En exploitant des connaissances du domaine, qui peuvent être données par un expert ou apprises sur un échantillon de données étiquetées, on distingue deux types d'arcs dans le graphe : des arcs représentant des dépendances à valeurs booléennes et des arcs représentant des dépendances à valeurs réelles. Si la similarité d'un nœud n dépend seulement de la *réconciliation* d'une paire de références (ou de valeurs) représentées dans un nœud m, et non de son score de similarité, alors m est appelé *voisin à valeur booléenne*. Dans le cas où, la similarité de n dépend de la *valeur de similarité* courante de m, alors m est appelé *voisin à valeur réelle*. Par exemple, la similarité de deux références de conférences c_1 et c_2 dépend de la valeur réelle de similarité de leur nom et dépend de la décision de réconciliation des articles a_1 et a_2 qui y sont publiés. Parmi les voisins à valeur booléenne, deux types de voisins sont distingués : (i) si la réconciliation d'une paire de références (ou de valeurs) m correspondant à un voisin à valeur booléenne implique la réconciliation de la paire de références (ou de valeurs) du nœud n alors ce nœud m est dit *voisin à valeur booléenne forte*, et (ii) si en revanche, sa réconciliation n'implique qu'une augmentation de la similarité de la paire de références (ou de valeurs) du nœud n alors ce nœud m est dit *voisin à valeur booléenne faible*.

Une fois que le graphe de dépendances des références est construit, les scores de similarité des paires de références et des valeurs sont calculés itérativement jusqu'à ce qu'un point fixe soit atteint. La similarité d'une paire de références ou de valeurs est calculée en fonction des scores de similarité de leurs paires voisines et renvoie une valeur dans [0..1].

Calcul itératif de la similarité par propagation dans le graphe de dépendances. Le graphe de dépendances permet de capturer toutes les dépendances entre les similarités et les décisions de réconciliation et sert également de guide pour la tâche de re-calcul des différents scores de similarité.

Dans le graphe, les nœuds sont marqués comme *réconcilié*, *actif* ou *inactif*. Un nœud est marqué comme *réconcilié* si son score de similarité dépasse un seuil de réconciliation donné. Un nœud est marqué comme *actif* si on doit recalculer sa similarité. Le reste des nœuds sont marqués comme *inactifs*. L'algorithme est composé des deux étapes suivantes et itère l'étape 2 , jusqu'à ce qu'aucun nœud ne soit marqué comme *actif* :

1. Initialement, tous les nœuds sont marqués comme *actifs*. Les nœuds représentant les similarités des valeurs des attributs sont marqués comme *réconciliés* si leur score de similarité est au dessus du seuil de réconciliation et ils sont marqués comme *inactifs*, sinon.
2. Un nœud *actif* est sélectionné pour recalculer sa similarité. Si le nouveau score de similarité

dépasse le seuil de réconciliation alors on le marque comme *réconcilié*, sinon on le marque comme *inactif*. Ensuite tous ses voisins ayant un score de similarité inférieure à 1, sont marqués comme *actifs*.

Pour calculer la similarité d'une paire de références (ou de valeurs), trois types de similarité sont calculées, selon les trois types de voisins et donc d'arcs définis précédemment :

Similarité des voisins à valeur réelle. Il s'agit de calculer une moyenne pondérée des scores de similarité des différents nœuds voisins. Soit S_{rv} la fonction qui permet de calculer le score de similarité des N voisins à valeurs réelles, pour une paire de références ou de valeurs, qui est la suivante :

$$S_{rv} = \sum_{i=1}^{i=N} \lambda_i x_i$$

où x_i représente le score de similarité du i-ème voisin à valeur réelle et λ_i le poids lui correspondant. λ_i représente le degré d'influence du score de similarité du i-ème voisin sur le score de similarité S_{rv}.

Similarité des voisins à valeur booléenne forte. Il s'agit d'incrémenter la similarité de la paire de références ou de valeurs d'une valeur constante β pour chaque voisin à valeur booléenne forte réconciliée. Ainsi, une décision de réconciliation pourra être inférée excepté dans le cas où le score de similarité des valeurs de leur attributs est très faible. Cela va permettre d'éviter, par exemple, de réconcilier des auteurs d'un article ayant des noms très différents. Pour une paire de références ou de valeurs, le score de similarité des voisins à valeur booléenne forte S_{bvF} est défini comme suit :

$$S_{bvF} = \begin{cases} \beta * |N_{bvF}| & Si,\ S_{rv} > seuil_{rv} \\ 0 & Sinon \end{cases}$$

où β est une constante, où $|N_{bvF}|$ est le nombre de voisins réconciliés à valeur booléenne forte et où $seuil_{rv}$ est un seuil de score de similarité indiquant que deux références peuvent représenter la même entité du monde réel.

Similarité des voisins à valeur booléenne faible. Il s'agit, cette fois d'incrémenter la similarité de la paire de références ou de valeurs d'une valeur constante γ plus faible. Par exemple, si on arrive à réconcilier deux auteurs qui sont en contact avec deux autres auteurs d'une paire de références, alors on augmente légèrement la similarité de cette paire. Pour une paire de références ou de valeurs, le score de similarité des voisins à valeur booléenne faible S_{bvf} est défini comme suit :

$$S_{bvf} = \begin{cases} \gamma * |N_{bvf}| & Si,\ S_{rv} > seuil_{rv} \\ 0 & Sinon \end{cases}$$

où γ est une constante (inférieure à β), où $|N_{bvf}|$ est le nombre de voisins réconciliés à valeur booléenne faible et où $seuil_{rv}$ est un seuil de score de similarité indiquant que deux références peuvent représenter la même entité du monde réel.

La terminaison du calcul itératif de la similarité est garantie sous certaines conditions (voir [DHM05]). Une fois que le point-fixe est atteint, on calcule une fermeture transitive qui permet d'inférer d'autres réconciliations par transitivité : à partir de $reconcile(a, b)$ et $reconcile(b, c)$ on infère $reconcile(a, c)$.

Les auteurs ont envisagé d'étendre leur approche afin d'éviter l'inférence de certaines fausses décisions de réconciliation grâce à l'exploitation de certaines connaissances du domaine. Par exemple, une connaissance peut spécifier que deux auteurs du même article sont distincts. Cette contrainte permet d'inférer des non réconciliations qui pourraient servir à filtrer et éventuellement à initilialiser la similarité d'autres paires de références.

Clustering relationnel. Bhattacharya et Getoor [BG04, BG06] ont formulé le problème de réconciliation de références comme un problème de clustering, où chaque cluster représente des références qui réfèrent à la même entité du monde réel. La similarité entre deux clusters combine la similarité des valeurs des attributs des références avec la similarité calculée en exploitant les relations entre données. Les auteurs s'intéressent dans ce travail à la réconciliation de références d'un type déterminé. Par exemple, dans une base de données de publications scientifiques, ils se sont intéressés à la réconciliation de références d'auteurs.

Voici un exemple de quatre articles, qui va nous permettre d'illustrer la méthode proposée par Bhattacharya et Getoor pour réconcilier les auteurs :

- P1 : W. Wang, C. Chen, A. Ansari, "A mouse immunity model"
- P2 : W. Wang, A. Ansari, "A better mouse immunity model"
- P3 : L. Li, C. Chen, W. Wang, "Measuring protein-bound fluxetine"
- P4 : W. W. Wang, A. Ansari, "Autoimmunity in biliary cirrhosis"

Dans cette approche, la *base de données de références* est la base permettant de stocker les informations concernant les références et les relations entre références. Elle contient une table de références $\mathcal{R} = \{r_i\}$ où, chaque référence a un identifiant $\mathcal{R}.id$ et un ensemble d'attributs $\{\mathcal{R}.A_1, \ldots, \mathcal{R}.A_j\}$. Chacun des articles P1, P2, P3 et P4 de l'exemple ci-dessus est décrit par un ensemble de noms d'auteurs et par un titre. Chaque auteur est représenté, dans la *base de données de références*, par une référence r_i et les attributs $r_i.Nom$, $r_i.Email$, etc. représentent la description en terme de valeurs d'attributs de chaque auteur r_i. Dans cette approche, une référence n'est pas considérée individuellement mais comme une référence appartenant à un ensemble de relations entre références que l'on modélise ici par un ensemble d'hyper-arêtes $\mathcal{H} = \{h_i\}$. Par exemple, les références d'auteurs d'un article sont considérés comme membres d'une hyper-arête représentant les relations co-auteurs entre ces références. L'hyper-arête à laquelle appartient une référence r_i est donnée par $r_i.\mathcal{H}$. On note que des attributs $\{\mathcal{H}.A_1, \ldots, \mathcal{H}.A_k\}$ sont associés à chaque hyper-arête, par exemple, le titre de l'article, le nom de la conférence et l'année de publication. Ces attributs vont être considérés lors du calcul de similarité fondé sur les valeurs des attributs des références d'auteurs.

La base de données de références contient, pour les quatre articles de l'exemple précédent, dix références d'auteurs $\mathcal{R} = \{r_1, r_2, \ldots, r_{10}\}$, correspondant dans l'ordre aux auteurs représentés par leur noms {"W. Wang", "C. Chen", ... ,"W. W. Wang", "A. Ansari" }. Dans cette base de données de références, une référence r_1 est décrite par ($r_1.id$=rid1, $r_1.Name$ ='W. Wang'). Dans la table $\mathcal{H}$, on représente l'ensemble des relations co-auteur d'un article par une hyper-arête. Il y'a donc autant d'hyper-arêtes que d'articles : $\mathcal{H} = \{h_1, \ldots, h_4\}$. Dans cette table $\mathcal{H}$, par

exemple, l'hyper-arête $h1$ est décrite par ($h_1.id$ = hid1, $h_1.Titre$ ="A mouse immunity model") ; Le fait que la référence r_1 appartienne à l'hyper-arête h_1 correspondant au premier article P1, est représenté dans la table $\mathcal{H}$ par $r_1.\mathcal{H} = h_1$.

La réconciliation de références consiste à partitionner les références selon les entités qu'elles représentent, en groupant dans un même cluster celles qui représentent la même entité du monde réel. Les références représentant la même entité du monde réel sont dites *co-référentes*. Pour illustrer cela, supposons que six entités sont représentées dans les dix références de l'exemple précédent : toutes les références ayant comme nom "A. Ansari" sont co-référentes, de même, pour les références ayant comme nom "L. Li". Par contre, les deux références ayant comme nom "C. Chen" ne sont pas co-référentes et donc représentent deux entités différentes. De même, les quatre références dont le nom est "W. Wang" se réfèrent à deux entités différentes. Pour ces dernières références, les méthodes qui auraient exploité uniquement la similarité des noms auraient trouvé qu'elles se réfèrent à la même entité du monde réel. C'est une des raisons pour laquelle Bhattacharya et Getoor ont proposé l'exploitation des relations entre références en plus des valeurs des attributs pour la réconciliation de références.

Dans [BG04], Bhattacharya et Getoor ont proposé un algorithme de clustering relationnel pour la réconciliation de références. Cet algorithme effectue des opérations de fusion de manière itérative sur un ensemble de clusters en utilisant une mesure de similarité entre clusters. Pour calculer cette similarité entre clusters de références, Bhattacharya et Getoor combinent deux types de similarité : une mesure de similarité entre deux clusters de références sim_A qui exploite les valeurs des attributs et une mesure de similarité sim_R dite relationnelle qui exploite les relations auxquelles les références des deux clusters participent. La similarité de deux clusters c_i et c_j est définie comme suit :

$$sim(c_i, c_j) = (1 - \alpha) * sim_A(c_i, c_j) + \alpha * sim_R(c_i, c_j), \quad avec \quad 0 \leq \alpha \leq 1$$

Clustering relationnel itératif. Initialement, chaque référence appartient à un cluster de références distinct. Tout d'abord, pour construire les premiers clusters de références, les clusters de références ayant une similarité sim_A, fondée sur les attributs, élevée sont fusionnés. Ensuite, l'algorithme de clustering relationnel itère sur les étapes suivantes :

1. Identifier les paires (c_i, c_j) de clusters les plus similaires c'est-à-dire ceux ayant une similarité dépassant un seuil donné ;
2. Fusionner les clusters identifiés comme similaires et créer un cluster c_{ij} contenant les références des deux clusters c_i et c_j ;
3. Mettre à jour les similarités des paires de clusters courants.

L'algorithme s'arrête quand la similarité de la paire de clusters les plus similaires est inférieure à un seuil donné, i.e. la distance inter-clusters est maximale.

Calcul de similarité entre les clusters de références. Nous présentons ci-après le calcul de similarité entre clusters contenant des références. Il s'agit de la similarité entre deux clusters de références en terme de valeurs d'attributs et de la similarité en terme de relations entre les références des clusters.

Similarité fondée sur les attributs. Pour calculer la similarité de deux clusters c_i et c_j de références en fonction des valeurs des attributs, l'algorithme proposé par Bhattacharya et Getoor, calcule tout d'abord, la similarité des valeurs des attributs des paires de références, appartenant au produit cartésien des références des deux clusters, en utilisant des mesures de similarité existantes de chaînes de caractères (e.g. celles présentées en Section 4.2). Ensuite, une moyenne pondérée des scores de similarité des valeurs des attributs est calculée pour associer un score de similarité aux paires de références appartenant respectivement aux clusters c_i et c_j. De plus, les auteurs considèrent la similarité des attributs décrivant les hyper-arêtes, dans l'exemple, il s'agit du titre de l'article écrit par les auteurs. Enfin, puisqu'on considère que chaque cluster représente une seule entité du monde réel alors, par transitivité, toutes les références d'un même cluster sont similaires. Ainsi, le score de similarité $sim_A(c_i, c_j)$ des deux clusters correspond au score de la paire de références ayant le score de similarité le plus élevé (i.e. le maximum).

Similarité relationnelle. La similarité relationnelle entre deux clusters de références est calculée en considérant l'ensemble de leurs clusters voisins [18]. Un cluster c_i est dit voisin d'un cluster c_k s'il existe une hyper-arête qui lie deux références appartenant respectivement à c_i et à c_k. Par exemple, un cluster c_1 d'auteurs est dit voisin d'un autre cluster c_2 s'il existe un auteur de c_1 qui est co-auteur d'un auteur de c_2.

Les hyper-arêtes qui sont pertinentes pour un cluster donné, sont celles qui concernent les références contenues dans ce cluster. L'ensemble des hyper-arêtes $c.\mathcal{H}$ pour un cluster c de références est défini comme suit :

$$c.\mathcal{H} = \{h | r.\mathcal{H} = h, r \in c\}$$

Par exemple, pour le cluster représentant l'unique référence correspondant à l'auteur "L. Li", l'ensemble $c.\mathcal{H}$ représente l'hyper-arête $h3$ correspondant au troisième article.

Soit $r.C$ l'étiquette associée à la référence r et qui représente le cluster auquel la référence r appartient. Soit $label(h_i)$ l'ensemble des étiquettes $r.C$ associées à toutes les références r appartenant à l'hyper-arête h_i. Par exemple, lorsqu'on a une seule référence par cluster, $label(h_1) = \{c_1, c_2, c_3\}$ avec c_1, c_2 et c_3 représentant les étiquettes des clusters auxquels les références d'auteurs r_1, r_2, et r_3 de l'article P1, appartiennent. Ainsi, le multi-ensemble $c_i.\mathcal{V}$ de voisins de c_i est défini comme suit :

$$c_i.\mathcal{V} = \bigcup_m label(h_i),\ h_i \in c.\mathcal{H}$$

avec $\bigcup_m$ l'opérateur d'union de multi-ensembles (i.e. conserve les redondances). Enfin, la mesure de similarité relationnelle sim_R entre deux clusters c_i et c_j consiste à appliquer la mesure de similarité Jaccard (pour plus de détails, voir en Section 4.2) sur les ensembles des étiquettes des clusters voisins $c_i.\mathcal{V}$ et $c_j.\mathcal{V}$:

$$sim_R(c_i, c_j) = Jaccard(c_i.\mathcal{V}, c_j.\mathcal{V})$$

Pour le clustering de références d'auteurs, l'ensemble de clusters voisins d'un cluster de références d'un auteur correspond à l'ensemble d'auteurs collaborateurs.

[18] La notion de voisin correspond aux clusters des références liées par des relations, par exemple, la relation co-auteur.

4.3.3 Discussion

Dans cette section de l'état de l'art, nous avons présenté les méthodes de réconciliation de références en les distinguant par leur capacité à exploiter les relations entre les données : *méthodes locales* vs *méthodes globales*. Parmi les méthodes locales de réconciliation de références certaines utilisent un modèle probabiliste pour comparer et décider de la réconciliation ou de la non réconciliation de références. Fellegi et Sunter [FS69], Winkler [Win89] et Verykios et al [VME03] sont des exemples représentatifs de ce type d'approches. Toutefois, ces méthodes sont très dépendantes de l'estimation des différents paramètres du modèle probabiliste : les *u-probabilités*, les *m-probabilités*, les seuils ou encore les coûts des fausses classifications. Cette dépendance peut être un obstacle pour certaines applications telles que les portails d'information en ligne. En effet, ces paramètres peuvent difficilement être estimés parce qu'on ne peut pas disposer de données étiquetées et leur estimation rendrait l'exécution de la méthode inefficace. En revanche, les méthodes locales fondées sur des mesures de similarité qui ont été proposées, par exemple, dans le domaine des bases de données, telles que Cohen et al. [Coh98b], Debabrata et al. [DSD98b] et Guha et al. [GKMS04] ne se confrontent pas à ce problème.

Les méthodes globales ont été proposées pour améliorer la qualité des résultats des méthodes de réconciliation en exploitant les relations entre les données. Ces relations peuvent être exprimées explicitement dans les données telles que les clés-étrangères dans les bases de données relationnelles ou les relations sémantiques dans d'autres modèles de données, comme dans le cas de [ACG02, DHM05, BG06] ou, peuvent être découvertes et utilisées dans l'algorithme de réconciliation de références de [KM06]. Parmi ces méthodes globales, certaines méthodes, telle que Dong et al. [DHM05], considèrent le problème de réconciliation de références dans un modèle de données où l'on s'intéresse à des réconciliations simultanées de références qui sont de plusieurs types, par exemple, des *personnes*, des *articles* et des *conférences*. Il s'agit dans ce cas de méthodes de réconciliation de références plus flexibles prenant en compte un contexte d'information plus riche. Afin de capturer les dépendances entre les paires de références induites par les relations liant ces références, les méthodes globales peuvent s'appuyer sur une structure de graphe qui représente les paires de références et leurs attributs associés dans un graphe de dépendances [DHM05]. D'autres approches modélisent le calcul par un système d'équations. Par exemple, [KM06] traduit le Principe Attraction de Contexte par un système d'équations permettant de calculer les forces de connexion entre les entités à travers les relations.

Les relations, lorsqu'elles existent, permettent de prendre en compte une quantité d'informations plus grande lors de la comparaison de descriptions de références. Ainsi, les méthodes améliorent la qualité des résultats obtenus par l'exploitation de la description locale des références. Nous avons vu que certaines approches utilisent cette information uniquement pour réduire le nombre de fausses réconciliations. C'est le cas par exemple du travail de [KM06] qui utilise un système de vote où, les résultats obtenus sur la description locale et globale doivent être consensuels. D'autres approches s'intéressent à l'augmentation du nombre de réconciliations et donc du rappel [ACG02, BG06] et d'autres encore s'intéressent aux deux aspects, c'est à dire à la réduction du nombre de fausses réconciliations et du nombre de fausses non réconciliations. De manière générale, parmi ces approches, peu se sont intéressées aux décisions de non réconciliation.

4.4 Réconciliation de références et exploitation de connaissances pour la réconciliation

Dans la section 4.3 précédente nous avons présenté les approches de réconciliation de références en distinguant celles qui exploitent uniquement les attributs des références (méthodes locales) et celles qui exploitent, en plus des attributs, les relations entre références (méthodes globales). Dans cette section, nous allons présenter les approches de réconciliation de références en fonction des connaissances additionnelles qu'elles exploitent et qui sont souvent nécessaires pour concevoir des méthodes efficaces et performantes.

Pour la réconciliation de références différents types de connaissances plus ou moins liées à la tâche de réconciliation peuvent être exploitées : choix et paramètres des mesures de similarité, importance des différents attributs dans la décision de réconciliation, connaissances concernant les références et les attributs (e.g. corrélation des attributs) et connaissances sur les valeurs telles que la synonymie. Pour acquérir ces connaissances certaines approches de réconciliation de références ont recours à des algorithmes d'apprentissage pour apprendre ces différentes connaissances à partir de données étiquetées par un expert. Il s'agit dans ce cas de méthodes dites *supervisées*. D'autres méthodes utilisent un langage déclaratif pour permettre à un expert de spécifier les connaissances nécessaires pour la réconciliation, par exemple, sous forme de règles. Il s'agit dans ce deuxième cas d'approches dites *déclaratives*.

4.4.1 Méthodes supervisées de réconciliation de références

Les méthodes supervisées de réconciliation de références supposent l'existence de données sous forme d'un ensemble de paires de références étiquetées comme réconciliées ou non réconciliées. En exploitant cet ensemble d'exemples positifs et négatifs ces méthodes utilisent des algorithmes d'apprentissage supervisé pour apprendre des connaissances ou des paramètres nécessaires pour la prise de décision de (non) réconciliation. Ces connaissances peuvent correspondre soit à des paramètres de mesures de similarité soit à des règles de (non) réconciliation pour les approches s'appuyant sur les mesures de similarité. Pour rendre l'apprentissage supervisé plus efficace, certaines méthodes utilisent des techniques d'*apprentissage actif* [CAL94] permettant de déterminer automatiquement les paires de références à présenter à l'utilisateur afin qu'il les étiquette comme réconciliées ou non réconciliées. Ainsi, ils réduisent la taille de l'ensemble de données étiquetées manuellement en choisissant les paires de références les plus informatives à présenter à l'utilisateur pour être étiquetées comme réconciliées ou non réconciliées.

Nous allons tout d'abord présenter une approche qui permet d'apprendre des connaissances pour adapter les mesures de similarité aux caractéristiques des données. Bilenko et al. [BM03a] ont proposé une approche supervisée qui utilise un classifieur *SVM –Support Vector Machine*. Il s'agit d'une technique de discrimination qui consiste à séparer deux ensemble de points – ici les scores de similarité des différents attributs – par un hyperplan. La méthode proposée par Bilenko et al. apprend sur un échantillon de paires de références étiquetées comme réconciliées ou non réconciliées la fonction qui combine les scores de similarité des valeurs des différents attributs. Il s'agit d'apprendre, dans une application donnée les différents poids des attributs décrivant les références en favorisant, par exemple, les poids des attributs ayant des valeurs de taille courte par rapport à ceux ayant des valeurs plutôt longue. Différentes mesures de similarité sont utilisées pour calculer les scores de similarité des valeurs des attributs des paires de références étiquetées comme réconciliées ou non réconciliées. Le classifieur SVM est entraîné sur l'ensemble des vecteurs de comparaison représentant les différents scores de similarité des

valeurs des attributs des paires de références. Ensuite, le classifieur entraîné est appliqué sur les nouvelles paires de références, non considérées dans l'étape d'apprentissage, pour obtenir les décisions de réconciliation. Dans leur expérimentation Bilenko et al. ont montré que les résultats obtenus par leur classifieur SVM sont meilleurs que ceux obtenus par une moyenne simple des scores de similarité. Bilenko et al. ont également proposé une méthode permettant d'exploiter un échantillon de paires de références pour entraîner une mesure d'édition avec affinement de gaps, comme la mesure Smith et Waterman (voir Section 4.2). Ils ont observé que les poids associés aux différentes opérations d'édition, telles que *suppression*, *remplacement*, *ouverture d'un gap* et *extension d'un gap* utilisées dans cette mesure peuvent être adaptés aux types de variations apparaissant dans les données. Par exemple, l'opération de *remplacement* des caractères *'/'* par *'-'* dans un numéro de téléphone doit avoir un poids faible car, il s'agit dans ce cas d'une variation assez fréquente dans les numéros de téléphone. En revanche, le poids de *remplacement* du caractère *'B'* par *'C'* dans le nom d'une personne "*Bart*" doit être plus fort. De la même façon le poids d'ouverture d'un gap en début de chaîne doit être plus faible que celui d'ouverture d'un gap au milieu de chaîne. Ils ont utilisé le modèle probabiliste *CMC-Chaîne de Markov Cachée* [Rab90] pour représenter la mesure d'édition par affinement de gaps qui produit des alignements de caractères entre deux chaînes de caractères. Pour une paire de chaînes de caractères, le modèle génère un alignement représenté par une séquence de paires de caractères ou de caractères alignés avec un gap correspondant à une insertion ou à une suppression de caractère.

La similarité entre deux chaînes de caractères dans ce modèle correspond à la probabilité de générer l'alignement le plus probable des caractères de ces deux chaînes. Le modèle CMC est entraîné [19] sur un ensemble de paires de références étiquetées comme réconciliées ou non réconciliées. Ensuite, le modèle est utilisé pour estimer la similarité des paires de chaînes de caractères en calculant la probabilité de générer l'alignement des paires de chaînes de caractères. Les chaînes de caractères alignées avec une forte probabilité sont considérées comme similaires et non similaires, sinon. Les résultats obtenus par cette mesure adaptée ont été comparés avec ceux obtenus par la mesure Smith et Waterman non adaptée. Bilenko et al. ont montré que les résultats étaient meilleurs sur sept jeux de données.

Nous allons présenter une approche qui permet d'apprendre à partir d'un ensemble de données étiquetées, des connaissances pour la réconciliation exprimées sous forme de règles. Tejada et al. ont proposé dans [TKM01] une approche supervisée de réconciliation de références qui apprend des connaissances pour la réconciliation en s'aidant d'informations données par l'utilisateur. Les connaissances apprises sont de deux types : (i) des degrés de confiance associés à des règles de transformation non dépendantes du domaine suggérant des correspondances possibles entre deux chaînes de caractères écrites différemment, comme les acronymes, les préfixes ou la lemmatisation, et (ii) des règles de réconciliation exprimées en fonction des attributs et de seuils de similarité donnés.

La méthode est effectuée en deux étapes. Tout d'abord, les scores de similarité de chaque paire de valeurs d'attributs sont calculés en utilisant la mesure de similarité *cosinus*. Cette mesure est appliquée aux vecteurs de poids TF/IDF des mots qui sont communs à une transformation près. Le score de similarité des paires de références est calculé par une somme pondérée des scores des valeurs des différents attributs. Ensuite, une étape d'apprentissage est effectuée pour obtenir les règles de réconciliation en fonction de valeurs de seuils donnés. Plus précisément, les auteurs utilisent les données étiquetées pour apprendre un arbre de décision qui permet de représenter les règles de réconciliation.

[19] Une variante de l'algorithme EM est utilisée pour apprendre les paramètres du modèle CMC.

Exemple : voici deux relations provenant de deux sources. Elles contiennent des listes de musées décrits par leur nom et leur adresse.

Nom	Adresse	CP	Ville	Telephone
Louvre	99, rue Rivoli	75001	Paris	01 40 20 53 17
Musée de Cluny	6, place Paul Painlevé	75005	Paris	01 53 73 78 00

Nom	Adresse	Telephone
Louvre	99, r. Rivoli, Paris 1er	01 40 20 50 50
ORSAY	1, r. de la Légion d'Honneur Paris 7ème	01 40 49 48 14

FIG. 4.9 – *Deux relations à intégrer*

Par exemple, pour les références des relations présentées en Figure 4.9, on peut avoir la règle de réconciliation suivante :

$$Si\ (Nom > seuil1) \wedge (Adresse > seuil2)\ Alors\ Reconciliation.$$

Cette règle exprime le fait que, pour une paire de références, si le score de similarité calculé pour les valeurs de l'attribut Nom est dépasse un $seuil1$ et que le score de similarité calculé pour les valeurs de l'attribut $Adresse$ dépasse un $seuil2$ alors on classe cette paire de références comme réconciliée.

La construction d'un arbre de décision permet de déterminer, étant donné des valeurs de seuils de similarité, les attributs ou les combinaisons d'attributs les plus appropriés pour la réconciliation de références. Une règle de réconciliation est dite *pertinente* si elle permet de partitionner au mieux l'ensemble étiqueté de paires de références en paires de références réconciliées et non réconciliées.

L'algorithme d'apprentissage proposé dans cette méthode construit un arbre de décision tel que présenté en Figure 4.10, pour classer correctement des exemples positifs et négatifs donnés. Chaque nœud de l'arbre représente une condition (e.g. $Nom > 0.90$) à vérifier sur un attribut, et chaque arête est étiquetée par un résultat possible de la condition (e.g. vrai ou faux). Pour classer une paire de références candidate à la réconciliation, on commence par tester les conditions à partir du nœud racine jusqu'à atteindre une feuille dans l'arbre qui représente la décision pour cette paire. Par exemple, considérons les deux références des deux relations en Figure 4.9 concernant le musée du *Louvre*, ayant comme scores de similarité pour les valeurs des attributs : *(Nom : 1.0, Adresse : 0.78, Telephone : 0.83)*. L'arbre de décision de la Figure 4.10 classe comme une paire de références réconciliées, la paire de références correspondant au musée du Louvre des deux relations.

L'algorithme d'apprentissage procède comme suit : à partir d'un ensemble de paires de références étiquetées, des sous-ensembles d'exemples de paires de références sont générés aléatoirement. À partir de ces sous-ensembles d'exemples, un ensemble d'arbres de décision est construit. Ces arbres permettent de classifier de nouvelles paires de références comme réconciliées ou non réconciliées. Pour apprendre de manière efficace des règles de décision pour un domaine particulier, une méthode d'apprentissage supervisée est appliquée sur une combinaison d'arbres de décision. Ces arbres de décision votent pour choisir les exemples les plus informatifs de paires de références à montrer à l'utilisateur afin qu'il les classe comme réconciliées ou non réconciliées. Certaines classifications produites par ces arbres de décision divergent. C'est pour cela que les

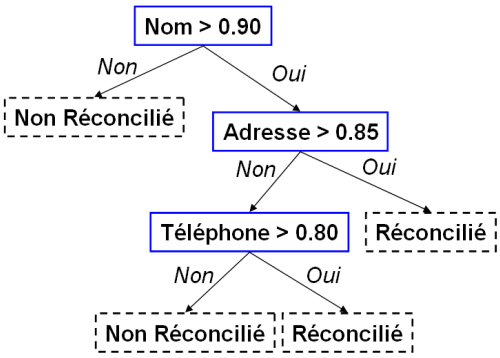

FIG. 4.10 – Exemple d'arbre de décision pour le domaine des Musées.

auteurs ont utilisé une procédure d'apprentissage actif pour choisir les exemples les plus informatifs de paires de références à montrer à l'utilisateur afin d'améliorer l'homogénéité et la qualité des décisions. Plus précisément, les paires, pour lesquelles les arbres de décision donnent des décisions différentes, sont proposées à l'utilisateur pour être étiquetées, ce qui va permettre de mettre à jour les arbres de décision. Le processus s'arrête quand les décisions produites par les différents arbres de décision sont homogènes, i.e. convergence vers un consensus.

Enfin, une fois que les règles de réconciliation, représentées dans les arbres de décision, sont apprises, elles sont appliquées sur l'ensemble des nouvelles paires de références pour déterminer l'ensemble des paires de références réconciliées et celui des paires de références non réconciliées.

Dong et al. [DHM05], dans leur méthode globale itérative de réconciliation de références présentée en Section 4.3.2, calculent la similarité d'une paire de références en fonction de leurs voisins à valeurs réelles en utilisant un algorithme supervisé. Un ensemble de fonctions de similarité sont organisées dans un arbre de décision, où, chaque nœud, qui ne correspond pas à une feuille, représente une certaine condition, par exemple si une valeur de similarité existe (valeurs non nulles des attributs). Dans cet arbre de décision les feuilles de l'arbre représentent les fonctions de similarité qui doivent être appliquées, lorsque les conditions du chemin qui mène vers une de ces feuilles sont satisfaites dans les données traitées. Cette étape d'apprentissage permet de considérer les différentes situations suivantes, qui peuvent apparaître dans les données :

Valeurs nulles dans les attributs. Habituellement, dans le cas où une des valeurs d'une paire d'attributs comparée n'est pas renseignée, un score de similarité de 0 est assigné à la paire de valeurs. Par conséquence, le rappel de la méthode est diminué. Dong et al. proposent d'ignorer l'attribut en adaptant le poids λ_i assigné à cet attribut, en mettant 0.0 par exemple.

Attributs clé. Considérer les attributs qui servent de clé pour l'entité représentée par la référence. Par exemple, si l'attribut adresse mail est clé pour l'entité personne alors lorsque les valeurs de cet attribut sont les mêmes pour deux références de personnes alors on les réconcilie sans considérer les autres attributs.

Paires de valeurs non réconciliées. Considérer dans le calcul de similarité le fait que des décisions de non réconciliation soient associées à certaines paires de valeurs. Cela va permettre d'augmenter la précision de la méthode.

Pour calculer la similarité à valeur réelle pour une paire de références, on parcourt l'arbre

de décision de la racine jusqu'aux feuilles en vérifiant les conditions sur les descriptions des références, la feuille atteinte représente la fonction de similarité qui doit être appliquée. En résumé, cette méthode est utilisée pour ajuster les pondérations associées aux différents attributs qui représentent leur degré d'influence sur le score de similarité associé aux paires de références.

Une autre méthode globale itérative qui s'appuie également sur des techniques d'apprentissage est la méthode du clustering relationnel proposée par Singla et Domingos [SD05]. A la différence de Bhattacharya et Getoor [BG06] qui considère un graphe où les nœuds représentent des références, Singla et Domingos ont observé qu'en utilisant les valeurs d'attributs comme nœuds dans le graphe, il devient possible de propager des informations à travers les nœuds et ainsi améliorer la qualité des résultats de la réconciliation de références, par exemple, si on décide de la réconciliation de ces deux références r1 : ("Équipe Gemo", "Parc Club d'Orsay", "Orsay") et r2 : ("Gemo", "PCO" , "Orsay") alors les valeurs "Parc Club d'Orsay" et "PCO" sont aussi considérées comme réconciliées (égales), et donc cette information pourra être utile pour d'autres comparaisons de paires de références. Ils supposent que les seules différences qui peuvent surgir sont dues aux différentes représentations des données et non pas aux erreurs d'association de valeurs pour les attributs. Par exemple, mettre que l'équipe "Gemo" est localisée à "Paris". Sous cette hypothèse, le modèle probabiliste relationnel (i.e. prend en compte les relations entre les références) qu'ils proposent dans [SD05] est entraîné sur un échantillon de données étiquetées. Ainsi, l'exploitation d'un ensemble de données étiqueté, leur permet également d'apprendre un ensemble de connaissances sur les valeurs.

4.4.2 Méthodes déclaratives de réconciliation de références

Dans ce type d'approche de réconciliation de références, au lieu d'utiliser des algorithmes supervisés pour apprendre les connaissances nécessaires à la réconciliation de références, on considère que ces connaissances peuvent être données par un expert du domaine sous forme de règles de réconciliation plus ou moins complexes.

Quand les connaissances du domaine sont exploitées par les méthodes de réconciliation de références de manière ad-hoc en les codant dans l'algorithme de réconciliation, on parle alors d'approches dites *non déclaratives*. Ces méthodes *non déclaratives* ne peuvent pas assurer la généricité de la réconciliation de références. En d'autres termes, quand on change de domaine d'application (e.g. tourisme, publications scientifiques) ces méthodes de réconciliation doivent être reconçues de manière à ce que les connaissances du nouveau domaine soient prises en compte. Les méthodes *déclaratives* exploitent les connaissances du domaine déclarées par un expert sous forme de règles de réconciliation. L'avantage des approches déclaratives réside dans leur adapatibilité, c'est-à-dire lorsqu'on modifie les connaissances du domaine ou lorsqu'on change de domaine d'application, il est uniquement nécessaire de mettre à jour les axiomes ou les règles de (non) réconciliation données par l'expert du domaine d'application.

Nous présenterons tout d'abord des approches de réconciliation de références exploitant les connaissances du domaine sous forme de règles de réconciliation. Les règles sont des règles de production et elles sont de la forme : *SI <condition> ALORS <action>*. Des prédicats et des fonctions complexes peuvent être contenus dans les parties *condition* et *action* de la règle.

Hernández et Stolfo [HS98] ont proposé une approche qui s'appuient sur des règles de réconciliation spécifiées manuellement par un expert du domaine pour inférer des réconciliation entre références. Par exemple, un expert exprime par une règle de réconciliation la connaissance suivante : si deux musées ont des noms *similaires* et si ces musées ont la même adresse alors il

s'agit du même musée. La règle correspondante est :

```
FORALL (m1, m2) IN CulturalPlace
IF (m1.MuseumName IS–SIMILAR–TO m2.MuseumName)
AND (m1.MuseumAddress = m2.MuseumAddress)

THEN (m1 MATCHES m2).
```

On note que IS–SIMILAR–TO correspond à une mesure de similarité entre chaînes de caractères comme celles présentées en Section 4.2, et MATCHES permet de déclarer que la paire de références est réconciliée.

Une autre méthode déclarative, nommée *Intelliclean*, a été proposée par Wai Lup Low et al [LLL01]. Elle permet d'inférer des réconciliations entre références avec un certain *facteur de certitude (FC)*. Il s'agit d'une méthode qui exploite les connaissances du domaine sous forme d'un ensemble de règles écrites par un expert du domaine dans le langage offert par l'interface du système expert JESS [FH99]. Une fois que ces règles ont été écrites, elles sont injectées dans le moteur d'inférence du système expert. L'inférence est effectuée en trois étapes. Une étape de pré-traitement où des erreurs syntaxiques sont éliminées et une normalisation des valeurs est effectuée en appliquant des règles de transformation telles que les abréviations. À l'étape de traitement les règles de réconciliation, comme celle présentée en Figure 4.11, sont appliquées pour inférer les réconciliations entre références. Enfin, une dernière étape où l'on calcule la fermeture transitive afin d'obtenir encore plus de réconciliations. On note que ces trois étapes sont des étapes qui sont suivies par la majorité des approches de réconciliations de références.

Les deux principaux types de règles utilisés dans la méthode *Intelliclean* sont :

1. **Règles de réconciliation**. Ces règles spécifient les conditions pour qu'une décision de réconciliation soit effectuée pour une paire de références. Ces règles de réconciliation contiennent des fonctions de similarité des valeurs des attributs mais aussi des prédicats plus au moins complexes afin de déterminer la réconciliation des références.

Définir règle *regle-musee1*
Entrée references : $m1$, $m2$

SI $(m1.telephone = m2.telephone)$
ET $(SOUSCHAINES(m1.Id, m2.Id) = VRAI)$
ET $SIMILARITE(m1.adresse, m2.adresse) > 0.8)$
ALORS
$RECONCILIER(m1, m2)$ $CERTITUDE = 0.8$.

FIG. 4.11 – *Intelliclean : un exemple de règle de réconciliation*

Par exemple, la règle de réconciliation *regle-musee1* présentée en Figure 4.11 est appliquée pour la réconciliation de références d'un ensemble de références de musées décrites par les attributs *Id*, *Adresse* et *Telephone*. Pour que la règle *regle-musee1* puisse être déclenchée : (i) les valeurs de l'attribut *Telephone* doivent être égales, (ii) la valeur de l'attribut *Id* d'une des références doit être une chaîne de l'autre et (iii) les valeurs des adresses doivent être très similaires (leur score de similarité supérieur à 0.8). Si ces conditions sont vérifiées

alors on infère la réconciliation de la paire de musées avec un *facteur de certitude* de 0.8. Un *facteur de certitude (FC)* représente le degré de confiance de l'expert du domaine dans les réconciliations inférées par cette une règle, avec $0 \leq FC \leq 1$. Un expert associe un degré de certitude élevé à une règle s'il est sûr que cette règle permet d'identifier des réconciliations correctes. Il associe un degré de certitude plus faible pour des règles moins strictes.

2. **Règles de filtrage**. Il s'agit de règles qui contraignent les descriptions des références qui doivent être comparées : par exemple, "ne garder que les références ayant un nombre négligeable d'attributs non renseignés (valeurs nulles)". Des actions plus complexes peuvent être exprimées dans ces règles de filtrage.

L'étape d'inférence des réconciliations à partir des différentes règles est suivie d'une étape de calcul de la fermeture transitive afin d'améliorer le rappel de la méthode. Cependant, comme les réconciliation inférées sont associées à un facteur de certitude plus ou moins élevé, les auteurs proposent donc un calcul de la fermeture transitive considérant cette incertitude des réconciliations. Nous présentons cette méthode à travers un exemple. Soit l'ensemble de réconciliations suivant : { ((A,B) avec FC = 0.9) ; ((B,C) avec FC = 0.85) ; ((C,D) avec FC = 0.8)}. Supposons que SR = 0.5 est un seuil défini par l'utilisateur et représente la valeur de FC à partir de laquelle on autorise une réconciliation. Les deux paires (A,B) et (B,C) sont considérées en priorité car elles ont les plus grands FCs. Ces paires sont fusionnées pour former un groupe (A, B, C) avec FC = 0.765 (FC est calculé par 0.9 * 0.85 = 0.765). Ensuite, ce groupe de références est fusionné avec la paire (C, D) pour former le groupe (A, B, C, D) avec FC = 0.612 (FC est calculé par 0.765 * 0.8 = 0.612). On remarque que la valeur résultat de FC est supérieure au seuil SR. Si SR = 0.7 alors le groupe (A, B, C) est fusionné et la paire (C, D) restera séparée du groupe, car FC = 0.612 est inférieur à SR.

D'autres approches exploitent des connaissances déclarées sous forme de règles pour pallier le problème de l'absence de combinaisons d'attributs qui peuvent servir de clé pour les références. Wang et Madnick [WM89] ont proposé une approche qui exploite des règles écrites par un expert du domaine afin d'inférer des informations qui complètent les descriptions des références. Ces informations inférées sont utilisées pour trouver des combinaisons d'attributs formant des clés communes aux différentes références. Par exemple ces règles peuvent être données par un expert :

SI age < 22 ALORS $status = non - gradue$ SINON $status = gradue$.
SI distanceDeDomicile > 10 ALORS $transport = voiture$ SINON $transport = velo$.

En appliquant ce type de règles Wang et Madnick souhaitent générer des valeurs de clés uniques afin de pouvoir grouper les références qui se réfèrent à la même entité. Dans le même esprit Lim et al. [LSPR96] proposent aussi une approche qui exploite des règles pour compléter les descriptions des références par l'inférence de nouvelles informations. À la différence de Wang et Madnick où les règles données par l'expert sont fondées sur des heuristiques, Lim et al. ne s'autorisent que des règles certaines pour garantir une bonne précision des réconciliations.

Une méthode originale de réconciliation de références qui exploite les connaissances du domaine est celle proposée par Doan et al. [DLLH03]. L'originalité de cette approche consiste à exploiter des corrélations pouvant exister entre des attributs disjoints entre références à la différence de la plupart des approches où on se focalise uniquement sur les attributs communs. Les auteurs ont observé que souvent des attributs différents sont corrélés. Par exemple, considérons les deux références suivantes : r_1 : (7, "*Dupont Martin*") et r_2 : ("*Dupont Martin*", $150K$). Dans

le cas où l'on détecte la réconciliation de ces deux références r_1 et r_2, nous obtenons alors une référence qui représente la personne "*Dupont Martin*" âgée de 7 ans et ayant comme salaire $150K$. Cependant, il est impossible qu'une personne de 7 ans ait un salaire de $150K$, ce qui montre qu'une corrélation existe entre les attributs a priori disjoints *âge* et *salaire* d'une personne. C'est pour cela que Doan et al. exploitent des *profils* contenant des connaissances sur des concepts du domaine (e.g. personnes, films, lieu).

Les *profils* sont décrits par un expert du domaine et contiennent des connaissances sur une instance (référence) typique d'un concept du domaine. Pour une paire de références donnée et un profil d'un concept donné le *profiler* donne un score de confiance représentant le degré de compatibilité entre les informations décrivant la paire de références et le profil du concept qu'elles représentent. Un type de profil plus strict est aussi considéré dans cette méthode, il s'agit des profils représentant des contraintes "strictes" sur un concept et qui *doivent* être satisfaites par toute référence représentant ce concept. Par exemple, la date de sortie d'un film doit être plus récente que celle de sa production. Les profils plus souples représentent des contraintes que l'on souhaite être satisfaites par les références du concept, avec l'autorisation de quelques exceptions. Par exemple, une contrainte souple sur les films consiste à considérer une forte corrélation entre la valeur de la note donnée par *IMBD* et celle donnée par *AlloCiné* et que l'on autorise une différence d'au maximum deux points.

Les *profilers stricts* renvoient une réponse Vrai/Faux alors que les *profilers souples* renvoient un score représentant le degré de compatibilité entre la paire de références et le profil du concept. Pour combiner les différents résultats envoyés par les différents profilers Doan et al. utilisent une somme pondérée des différents résultats. Les poids des différents profils sont donnés manuellement en s'appuyant sur des résultats expérimentaux.

Ce travail a été étendu par Shen et al [SLD05] pour exploiter d'autres types de connaissances afin d'améliorer la qualité des résultats de la réconciliation de références. Á la différence de [DLLH03], les auteurs proposent dans ce travail une exploitation des connaissances du domaine sous forme de deux types de contraintes : des contraintes fortes (e.g. "*aucun chercheur n'a publié plus de cinq articles à AAAI la même année*") et des contraintes faibles (e.g. "*le chercheur ayant comme nom 'Mayssam Saria' a moins de cinq articles dans DBLP*"). De plus, ils ont montré comment ces contraintes peuvent être exprimées dans un cadre probabiliste. Par exemple, la contrainte "une personne âgée de deux ans ne peut pas avoir un salaire de 150K" peut être modélisée par la probabilité conditionnelle s uivante : $Pr(r = e \mid [salary(r) = 150K] \wedge [\exists r' \; tq. \; (r' = e) \wedge (age(r') = 2)]) = 0$, avec $r = e$ exprime le fait que la référence r réfère à l'entité du monde réel e. L'algorithme EM [DLR77] est utilisé pour estimer les probabilités des différentes contraintes du domaine. On note que les contraintes fortes ont une probabilité égale à 0 ou à 1, alors que les contraintes faibles ont une probabilité dans l'intervalle [0..1].

AJAX [GFSS00, GFS01] est un prototype de réconciliation de références qui fournit un langage déclaratif pour la spécification des opérations de réconciliation de références qui, traditionnellement, sont codées de manière ad-hoc dans les méthodes. Il s'agit des opérations de normalisation de valeurs, d'extraction de groupes de références, de mise en correspondance des références, des opérations de fusion des références détectées comme réconciliées, etc. Le processus de réconciliation de références est exprimé sous forme d'une chaîne de traitement dans laquelle les différents opérateurs sont ordonnés et qui est représentée sous forme de graphe où les nœuds représentent les différents opérateurs et les arcs représentent leurs entrées et sorties. Ce travail a été adapté au traitement de données biologiques et a été implémenté dans le prototype BIO–AJAX [HGP+04].

4.4.3 Discussion

Dans cette section nous avons présenté les approches de réconciliation de références en s'attachant aux moyens d'acquérir et d'exploiter les connaissances. Les méthodes de réconciliation de références qui exploitent les connaissances de manière ad-hoc en les codant directement dans la méthode sont très vulnérables aux changements des sources de données et aux changements de domaine (e.g. médical, démographique, commercial). Par exemple, les données de recensement de population en Europe ou en Asie ont des caractéristiques différentes en ce qui concerne les noms de personnes (e.g. nombre de prénoms), une information non segmentée peut être présentée dans des ordres différents selon la culture du pays, par exemple, au Japon, en Corée et en Chine le nom de famille précède le prénom alors que dans les pays européens la configuration (*Prénom–Nom*) est plus fréquente. De manière plus générale la qualité des sources de données peut être différente (e.g. présence d'erreurs, abréviations). L'exploitation déclarative ou supervisée des connaissances du domaine rend les méthodes capables de s'adapter aux changements des données et des caractéristiques des applications. Pour les méthodes qui acquièrent les connaissances du domaine par apprentissage supervisé telles que Bilenko et al. [BMC+03] et Tejada et al. [TKM01] lorsque les données changent il faut créer un nouvel échantillon et réappliquer la méthode d'apprentissage. En ce qui concerne les méthodes utilisant un langage déclaratif pour spécifier les connaissances telles que Hernández et Stolfo [HS98], Wai Lup Low et al. [LLL01] et Doan et al. [DLLH03] il faut re-spécifier, en tenant compte des changements des données, les connaissance nécessaires telles que, les règles de transformation ou de réconciliation ou les descriptions des profils des concepts. En plus de la généricité des méthodes exploitant les connaissances du domaine, la qualité des résultats de ces méthodes est améliorée en terme du nombre de fausses réconciliations et du nombre de fausses non réconciliations. En revanche, le coût en terme d'effort humain nécessaire pour ces méthodes, soit pour étiqueter les données d'apprentissage soit pour déclarer manuellement les règles, rend ces méthodes difficilement applicables à de grands volumes de données ou à des données qui évoluent. Nous avons donc besoin de méthodes qui garantissent les qualités assurées par l'approche de réconciliation exploitant des connaissances et qui comblent les faiblesses des ces approches en terme d'effort manuel, ceci en rendant l'acquisition des connaissance la plus simple possible.

4.5 Réduction du nombre de comparaisons de références

Pour résoudre le problème de réconciliation de références entre deux sources de données A et B les méthodes doivent considérer un espace de réconciliation de taille égale au produit cartésien $A \times B$ des références contenues dans les deux sources. Néanmoins, certaines comparaisons de paires de références peuvent parfois être évitées, c'est pour cela que des méthodes ont proposé des pistes pour réduire la taille de l'espace de réconciliation. Quatre méthodes différentes ont été proposées : les méthodes dites de blocking, les méthodes à voisins triés et les méthodes de clustering.

Méthodes de blocking. Elles ont été proposées pour la première fois par Newcombe [NK62, New88]. Il s'agit des méthodes qui partitionnent l'ensemble de références en un ensemble de blocks (ou groupes) mutuellement exclusifs tout en limitant le nombre de comparaisons à l'intérieur d'un block. Afin de créer les blocks, une *clé de blocking* est choisie et ainsi sont groupées dans un seul block les références ayant les mêmes valeurs pour la *clé de blocking*. Pour créer une table de hachage des paires de références, on peut utiliser des fonctions de hachage telle que le code Soundex [Rus18] ou NYSIIS [Taf70] sur des attributs fortement discriminants tels que les noms de personnes et comparer ensuite uniquement les références ayant des valeurs d'attributs similaires et non nécessairement identiques. Ces méthodes de blocking améliorent significativement l'efficacité de la réconciliation de références en terme de temps d'exécution. En revanche, elles peuvent conduire à une baisse du rappel dans le cas où des références sont mises dans des blocks de manière erronée car les autres possibilités de réconciliation ne sont plus considérées.

Méthodes à voisins triés. Hernández et Stolfo [HS98] ont proposé la méthode de réconciliation à *voisins triés*. Cette méthodes s'effectue en trois étapes :

1. *Création de la clé.* Étant donnée la liste des références dérivée de l'union des sources à réconcilier, une clé est calculée pour chaque référence en extrayant un sous ensemble d'attributs pertinents.
2. *Tri des références.* La liste de références est triée en utilisant la clé calculée à l'étape précédente.
3. *Fusion.* Une fenêtre de taille fixe est glissée à travers la liste de références triée afin de réduire le nombre de comparaisons des références au nombre de références contenues dans la fenêtre. Si la taille de la fenêtre est de w alors toute nouvelle référence contenue dans la fenêtre est comparée avec les $w - 1$ références précédentes pour trouver des réconciliations de références.

Pour augmenter le nombre de réconciliations, Hernández et Stolfo ont proposé plusieurs stratégies d'implémentation en utilisant différentes clés de tri et en fixant une taille plus petite pour la fenêtre.

Méthodes de clustering ou de *Canopies*. Monge et Elkan [ME97] ont proposé une méthode pour éviter de considérer la totalité de l'espace de réconciliation en supposant que la réconciliation est transitive ; par exemple, si la référence a est réconciliée avec b et b est réconciliée avec c, alors par transitivité a est réconciliée avec c. Sous cette hypothèse, le problème de réconciliation de références revient à déterminer les composantes fortement connexes de références dans un graphe non orienté. Ainsi, ces dernières correspondent à la fermeture transitive des réconciliations de références. Dans [ME97] les auteurs proposent un moyen efficace pour calculer les composantes

fortement connexes. Le principe consiste à considérer une référence non similaire à une référence, qui existe déjà dans la composante fortement connexe comme non similaire avec toutes les autres références de cette même composante fortement connexe.

4.6 Synthèse

Dans cet état de l'art nous avons présenté un ensemble de méthodes représentatives des différents types d'approches pour le problème de réconciliation de références. Les méthodes de réconciliation de références cherchent à décider si deux descriptions de données réfèrent à la même entité du monde réel. Ce problème se produit lorsque différentes présentations sont utilisées pour décrire les mêmes entités du monde réel, lorsque les données contiennent des erreurs et lorsque les informations peuvent être incomplètes.

Nous avons distingué les méthodes de réconciliation de références tout d'abord sur la façon d'exploiter les descriptions des références, puis sur la façon d'acquérir les connaissances.

En ce qui concerne les descriptions des références, nous avons distingué trois types d'approches possibles. La première est l'exploitation des descriptions des références non structurées en attributs. Ces travaux, tels que les travaux de William Cohen dans son système WHIRL [Coh00], de Monge et Elkan [ME97], de Gravano et al. [GIKS03] et de Bike et Naumann [BN05], calculent un score de similarité en exploitant uniquement les valeurs textuelles sous forme d'une seule chaîne de caractères sans que l'on distingue quelle valeur est associée à quel attribut. Ce type d'approche est intéressant lorsqu'on souhaite effectuer un calcul rapide de la similarité [ME97, Coh00, GIKS03], avoir un ensemble de paires candidates à la réconciliation pour ensuite leur appliquer un calcul de similarité plus sophistiqué [BN05] ou encore lorsque l'association (attribut, valeur) n'est pas certaine. C'est le cas, par exemple, des données extraites de manière automatique à partir du Web telles que les références bibliographiques du portail CiteSeer [GBL98] qui sont sujettes à des erreurs dues aux variations dans la structuration des informations dans les pages Web. Le second type d'approche consiste à considérer la description des références structurées en attributs. Une multitude de méthodes ont adopté cette vision en proposant soit des modèles probabilistes, telles que [FS69, Win02, VEH00], qui permettent de prendre des décisions de réconciliation après l'estimation des différents paramètres du modèle probabiliste, soit en calculant un score de similarité pour les paires de références, telles que [DSD98a, GKMS04]. Dans ce dernier cas, il s'agit de calculer les scores de similarité des valeurs des attributs (e.g. par des mesures de similarité comme celles présentées en Section 4.2) et ensuite de les combiner par une fonction d'agrégation en capturant éventuellement les degrés d'influence des différents attributs sur la similarité des paires de références. Un dernier type d'approches consiste à considérer non seulement la description structurée en attributs des références mais aussi les relations qui les lient les unes aux autres, il s'agit des approches globales. Ces approches globales [DHM05] [KM06] visent à améliorer la qualité des résultats en terme du nombre de fausses réconciliations (faux positifs) [ACG02, BG06] ou en terme du nombre de fausses non réconciliations (faux négatifs).

L'autre volet de cet état de l'art concerne l'exploitation de connaissances pour la réconciliation de références. Pour améliorer leur efficacité, certaines méthodes de réconciliation de références exploitent des connaissances apprises par des algorithmes d'apprentissage supervisé ou déclarées explicitement par un expert. Pour ce volet, nous avons distingué deux types d'approches : celles qui exploitent des données étiquetées pour apprendre les différentes connaissances et celles qui ont recours à des experts du domaine pour exprimer ces connaissances par le biais d'un langage déclaratif. L'exploitation des connaissances permet aux méthodes d'être adaptables à l'évolution des données et aux changements de domaines (e.g. tourisme, épidémiologie, démographie). Toutefois, ces méthodes sont dépendantes de l'expertise humaine pour étiqueter les données d'apprentissage ou pour spécifier déclarativement des connaissances sur la tâche de réconciliation.

Dans le cadre de cette thèse nous avons proposé une méthode réconciliation de références dont l'originalité est de combiner une méthode logique (L2R) et un méthode numérique (N2R). La méthode logique permet d'inférer des réconciliations et des non réconciliations certaines. La méthode numérique permet de calculer des scores des similarité pour des paires de références pour lesquelles la méthode logique n'a pas permis d'inférer leur réconciliation ou leur non réconciliation. Notre méthode logique et numérique de réconciliation de références (LN2R) peut être caractérisée de :

- **Globale** : la description des références en terme d'attributs et en terme de relations est exploitée à la fois pour inférer des réconciliations et des non réconciliations par la méthode logique ainsi que pour calculer les scores de similarité des paires de références par la méthode numérique ;

- **Itérative** : le calcul (symbolique ou numérique) de la similarité des paires de références est inféré par itération successive jusqu'à saturation ou atteinte d'un point fixe ;

- **Non supervisée** : aucun étiquetage manuel des paires de références n'est nécessaire ;

- **Déclarative** : exploitation des connaissances sur le schéma et sur les données qui peuvent être exprimées simplement ;

- **Apte à capitaliser son expérience** : plus la méthode de réconciliation est appliquée meilleurs les résultats seront. La méthode logique nous permet d'inférer des synonymies entre valeurs qui sont stockées et utilisées par la méthode logique elle même mais aussi par la méthode numérique.

Nous présentons dans le tableau 4.1 une comparaison des différentes méthodes que nous avons présentées dans cet état de l'art ainsi que la méthode LN2R que nous proposons.

Méthodes	Description globale	Acquisition des connaissances		Particularités
		Apprentissage	Déclaration	
Fellegi et Sunter [FS69], Winkler [Win02] et Jaro [Jar89]	Non	les m-probabilités, les u-probabilités et les seuils T_λ et T_γ (algorithme EM)	Non	Premier modèle formel
Verykios [VEH00] : méthode probabiliste fondée sur les coûts	Non	Non	Les coûts des règles de décisions	Modèle fondé sur les coûts des décisions
Debabrata et al. [DSD98b]	Non	Non	Un ordre sur l'importance des attributs (pour estimer ensuite leur degré d'influence)	Sous ensemble d'attributs communs : non déterminables par dépendances fonctionnelles.
Guha et al. [GKMS04]	Non	Non	Un entier paramètre k, utilisé pour obtenir les k-premières références	Calcul des k-premières références
Ananthakrishna et al. [ACG02] (Delphi)	Hiérarchie relationnelle	Non	seuils pour la similarité textuelle et pour la similarité de cooccurrence entre références	Hiérarchies relationnelles
Kalashnikov et al. [KM06]	Relations entre références	Non	Un entier paramètre L : indique la taille maximale des chaînes de relations	Découverte de relations entre références
Dong et al. [DHM05]	Relations entre références	Apprentissage (par Arbre de décision) des pondérations des attributs et des relations ajustés aux données traitées, e.g. valeurs nulles, paires non réconciliées.	– Seuils de similarité et de réconciliation – Constantes β et γ	– Graphe de dépendances, – Méthode applicable dans un PIM.
Bhattacharya et Getoor [BG06] : clustering relationnel	Relations entre références	Non	Pondération des attributs communs et pondération de la similarité entre clusters : fondée sur les attributs et sur les relations entre références.	Clustering relationnel pour la réconciliation de références
Bilenko et al. [BMC+03]	Non	– Les pondérations des attributs (par SVM) – Adaptation des mesures de similarité aux données (par CMC)	Non	Comparaison adaptative des valeurs
Tejada et al. [TKM01] (Atlas)	Non	– règles de décision de réconciliation (Arbre de décision) – poids des règles de transformation : normalisation des valeurs (Arbre de décision)	seuils de similarité et seuils de (non) réconciliation	Apprentissage de règles de réconciliation
Hernandez et Stolfo [HS98]	Non	Non	règles de réconciliation	Filtrage par la méthode à voisin triés
Wai Lup Low et al. [LLL01] : Intelliclean	Non	Non	– Règles de réconciliation – seuils de réconciliation	Approche fondée sur les règles
Doan et al. [DLLH03]	Non	Non	Profils des références représentatives des entités.	Corrélation entre des attributs différents
LN2R : Saïs et al. [SPR07b]	Relations entre références	Non	– Axiomes du schéma RDFS+, – Axiomes sur les sources de données, – Seuil de réconciliation pour N2R.	– Génération automatique des règles à partir d'axiomes, – Précisions à 100% (pour L2R), – Capitalisation de l'experience.

Tab. 4.1 – Tableau comparatif des différentes méthodes de réconciliation de références Description/Connaissances

Chapitre 5

Modèle de données

Sommaire

5.1 Langages d'ontologies 117
5.1.1 Langages RDF et RDFS 117
5.1.2 Langage OWL 119
5.1.3 Langage SWRL 120
5.2 Le modèle de données RDFS+ 122
5.2.1 Représentation du schéma et des axiomes du schéma 122
5.2.2 Représentation des données et de leurs axiomes 125

Le modèle de données que nous avons choisi pour représenter le schéma et les données est le langage RDF/RDFS. Nous avons étendu ce langage par un fragment du langage OWL-DL et par un ensemble de règles SWRL pour pouvoir exprimer des connaissances telles que la disjonction entre classes et les dépendances fonctionnelles. Avant de présenter plus précisément notre modèle de données que nous avons nommé RDFS+, nous survolons les trois langages RDF/RDFS, OWL-DL et SWRL.

5.1 Langages d'ontologies

Les ontologies jouent un rôle très important dans le Web Sémantique proposé par Tim Berners-Lee en permettant de spécifier de manière formelle des vocabulaires pour la description du contenu du Web. Ainsi, les informations du Web peuvent être accessibles et compréhensibles à la fois par les humains et par les machines. Plusieurs langages ont été proposés pour représenter les ontologies avec une expressivité plus ou moins élevée et une complexité de raisonnement plus ou moins grande. Nous nous intéressons aux trois langages de représentation d'ontologies recommandés ou soumis à recommandation par le W3C (*World Wide Web Consortium*) RDF/RDFS [McB04], OWL [MvH04] et SWRL [HPSB+04].

5.1.1 Langages RDF et RDFS

RDF (*Resource Description Framework*). Il s'agit d'un langage qui permet de décrire la sémantique des données, de manière compréhensible par les machines. Le modèle de données RDF est constitué de trois types de composantes :

- *Sujet* : il représente la *ressource*[20] à décrire. Il est nécessairement identifié par une URI.
- *Prédicat ou Propriété* : il s'agit d'une propriété utilisée pour caractériser et décrire une ressource. Un prédicat est nécessairement identifié par une URI.
- *Objet* : représente une donnée ou une autre ressource (identifiée par une URI).

En utilisant le langage RDF, les informations des ressources sont décrites par un ensemble de déclarations RDF sous forme de triplets :

$$(Sujet, Prédicat, Objet).$$

Cette notation en triplet est appelée N-TRIPLE. Une syntaxe en XML a été définie pour exprimer des déclarations RDF, on appelle cette syntaxe RDF/XML. Voici par exemple une déclaration RDF qui décrit une ressource (un musée) par la propriété *localiseA* et par la propriété *NomMusee* :

```
<rdf:Description rdf:about="http://www.louvre.fr">
    <rdf:type rdf:resource="http://www.paris-art.com/LieuxCulturels#Musee"/>
    <localiseA rdf:resource="http://www.paris.fr"/>
    <nomMusee> Le Louvre</nomMusee>
</rdf:Description>
```

Les mêmes déclarations peuvent être représentées dans un graphe étiqueté, où les noeuds ovales représentent les ressources identifiées par des URIs et les noeuds rectangulaires représentent les valeurs de type de base (e.g. chaînes de caractères, date). Les arcs représentent les propriétés des ressources dont la source correspond au sujet et la cible correspond à l'objet d'une déclaration RDF.

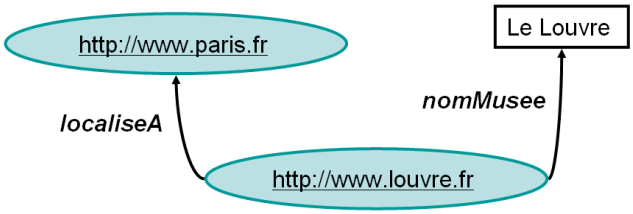

Fig. 5.1 – Exemple de graphe RDF

Une autre notation, qui est très utilisée, est la notation relationnelle : $predicat(objet, sujet)$

On réfère également à une déclaration RDF par le terme *fait RDF*. Par exemple, le fait RDF qui déclare le nom du musée du louvre est $nomMusee(http://www.louvre.fr,$"*Le Louvre*"$)$.

[20] [*WikiPedia*] Toute chose ou entité susceptible d'être identifiée, nommée, manipulée à travers ses représentations, par quelque moyen que ce soit, sur le Web en général ou dans n'importe quel système d'information utilisant les technologies du Web.

RDFS (*Resource Description Framework Schema*). Il s'agit d'un langage utilisé pour spécifier un vocabulaire pour la description des données RDF de façon à ce qu'il puisse être interprétable et utilisable par une machine. RDFS permet de spécifier les types des ressources par des classes (par exemple *Musée*, *Ville*, *Personne*) et de spécifier leurs propriétés (par exemple *localiseA*, *estDirecteurDe*). Le langage RDFS permet d'organiser les classes en une hiérarchie de classes en utilisant des relations de subsomption entre classes *subClassOf* et d'organiser les propriétés en une hiérarchie de propriétés en utilisant des relations de subsomption entre propriétés *subPropertyOf*. De plus, RDFS offre le moyen de spécifier le typage des propriétés en indiquant leur domaines et leur co-domaine (range), par exemple, le domaine de la propriété *localiseA* (voir la Figure 5.2) est la classe *Musee* et son co-domaine est la classe *Ville*.

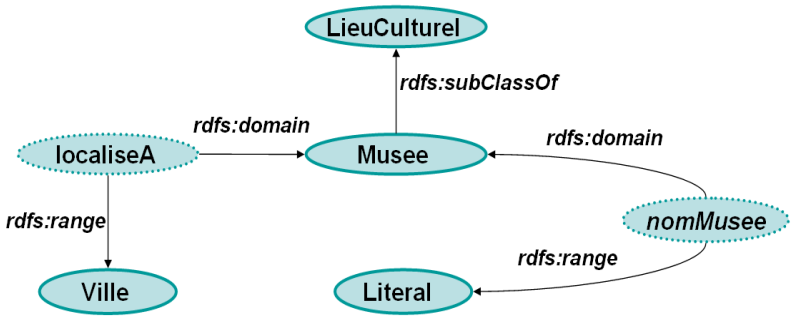

FIG. 5.2 – Un extrait du schéma RDF des données de la Figure 5.1

La sémantique en Logique du Premier ordre (LP) des déclarations RDFS de subsomption entre classes, de subsomption entre propriétés et de typage des propriétés est la suivante :

Constructeurs	**N-TRIPLE**	**Sémantique LP**
Subsomption entre Classes	C_1 *rdfs :subClassOf* C_2	(i) $\forall X(C_1(X) \Rightarrow C_2(X))$
Subsomption entre Propriétés	P_1 rdfs :subPropertyOf P_2	(ii) $\forall X\ \forall Y(P_1(X,Y) \Rightarrow P_2(X,Y))$
Typage du domaine d'une propriété	P rdfs :domain C	(iii) $\forall X\ \forall Y(P(X,Y) \Rightarrow C(X))$
Typage du co-domaine d'une propriété	P rdfs :range C	(iv) $\forall X\ \forall Y(P(X,Y) \Rightarrow C(Y))$

TAB. 5.1 – Sémantique des constructeurs RDFS en logique du premier ordre

À la différence des modèles orientés-objets où les propriétés des ressources sont encapsulées dans les classes, le langage RDFS permet de définir les classes et les propriétés séparément. Ainsi, l'évolution des schémas devient plus simple : des propriétés sur les ressources peuvent être ajoutées sans que la description de la classe d'origine soit redéfinie.

5.1.2 Langage OWL

Dans la section précédente, nous avons présenté le langage RDF/RDFS recommandé par le W3C pour représenter les ontologies. Cependant, RDF/RDFS a un pouvoir expressif limité. En effet, il est souvent nécessaire d'exprimer des contraintes supplémentaires sur les classes et sur les

propriétés, telles que, des contraintes de disjonction entre classes, des contraintes de fonctionnalité et des contraintes de cardinalité maximale et minimale sur les propriétés. C'est pour cela qu'il est souhaitable d'utiliser d'autres langages plus expressifs pour la représentation d'ontologies.

Le langage OWL (*Ontology Web Language*) est un langage de représentation d'ontologies avec un pouvoir expressif plus important et une complexité plus grande pour l'inférence. L'origine du langage OWL est le langage DAML+OIL.

Le langage DAML+OIL est le résultat de la combinaison du langage DAML (*DARPA Agent Markup Language*) et du langage OIL (*Ontology Inference Layer*), langages qui ont été réalisés dans le cadre de deux projets de recherche parallèlement lancés aux États-Unis et en Europe. Le langage DAML+OIL utilise un langage de représentation des connaissances fondé sur les Frames, introduit par Marvin Minsky [Min74], pour décrire la structure d'une ontologie dans un paradigme orienté-objet. Pour représenter une ontologie, le langage DAML+OIL permet de déclarer des classes, des propriétés et un ensemble d'axiomes déclarant les relations entre les classes et entre les propriétés, telle que l'équivalence entre classes et l'équivalence entre propriétés. Pour formaliser la sémantique des classes, des propriétés et des relations contenues dans l'ontologie, DAML+OIL utilise un modèle théorique équivalent à la logique de description $\mathcal{SHOIQ}$ [HPSvH03]. Il s'agit d'une logique de description qui est très expressive et qui est dotée de mécanismes d'inférence corrects et complets.

Le langage OWL est conçu au dessus des langages RDF/RDFS et DAML+OIL. Le langage OWL a une expressivité très élevée et une complexité de raisonnement très grande. Afin d'avoir un compromis entre l'expressivité et la complexité de ce langage, OWL a été conçu comme une famille de trois sous-langages OWL-Lite, OWL-DL et OWL-Full ayant une expressivité croissante et une complexité croissante. *OWL-Lite* est la version la plus simple des trois. Il permet la déclaration d'une hiérarchie de classes, d'une hiérarchie de propriétés et certaines contraintes (par exemple cardinalité à 0 ou à 1 sur les propriétés). *OWL-DL* a un pouvoir d'expression supérieur à celui de OWL-Lite, avec une capacité de raisonnement qui est complet (toutes les conclusions sont calculables) et décidable (les conclusions sont toujours calculables en un temps fini). De plus, OWL-DL correspond à la logique de description $\mathcal{SHOIQ}(\mathcal{D})$ (pour plus de précisions sur l'expressivité des logiques de description voir [Baa03]), d'où le nom OWL-DL. Enfin, *OWL-Full* est le sous-langage qui a l'expressivité maximale mais, en revanche, ne garantit pas la décidabilité du raisonnement.

5.1.3 Langage SWRL

Le langage SWRL (*Semantic Web Rule Language*) a été soumis comme candidat à la recommandation par le W3C comme langage de règles pour le Web sémantique. SWRL est un langage qui est fondé sur la combinaison du langage OWL-DL avec le sous langage Datalog RuleML (*Rule Markup Language*) [Dea01]. SWRL peut être vu comme une logique de description étendue par une syntaxe abstraite pour exprimer des règles de Horn[21] en logique du premier ordre sans symboles de fonctions. Par exemple, on peut déclarer des règles telles que :

$$filleDe(?x, ?z) \wedge freres(?z, ?w) \wedge filleDe(?y, ?w) \Longrightarrow cousines(?x, ?y)$$

Avec $FilleDe(?x, ?z)$, $Freres(?z, ?w)$ et $FilleDe(?y, ?w)$ sont les prémisses de la règle et $Cousines(?x, ?y)$ est la conclusion de la règle. Lorsque les prémisses sont satisfaites, la conclu-

[21] Une règle de Horn est une règle sous forme d'une implication d'un ensemble de formule atomiques booléennes vers une conclusion (une seule formule atomique). Avec la condition que toutes les variables apparaissant en conclusion de la règles apparaissent au moins une fois en prémisses de la règles.

sion de la règle est inférée. Ce type de relations ne peut pas être exprimé dans le langage OWL.

Cette règle est exprimée en syntaxe SWRL comme suit :

```
Implies(Antecedent(filleDe(I-variable(x) I-variable(z))
                   freres(I-variable(z) I-variable(w))
                   filleDe(I-variable(y) I-variable(w)))
Consequent(cousines(I-variable(x) I-variable(y))))
```

où, le constructeur *Antecedent* indique que ses arguments sont les prémisses de la règle, le constructeur *Consequent* indique que son argument est la conclusion de la règle et le constructeur $I - variable$ indique que son argument est une variable.

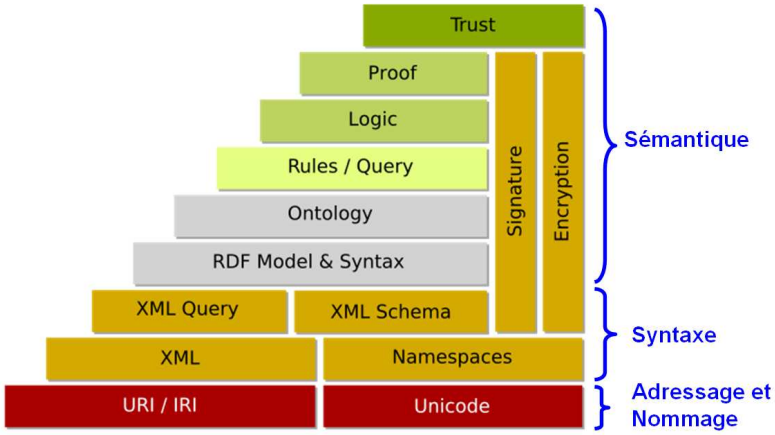

FIG. 5.3 – The Layer Cake proposé par Tim Berners-Lee [BL04]

D'après la vision de Tim Berners-Lee du Web sémantique, l'information pourra être accessible et compréhensible non seulement par les humains mais aussi par les machines. Dans cette vision, le Web sémantique peut être considéré comme une pile "*pièce montée*" de langages comme nous le montrons en Figure 5.3 et que l'on peut structurer sur trois niveaux principaux :

- ***Niveau d'adressage et de nommage***. Ce niveau est représenté par le standard d'adressage des ressources du Web URI et la norme Unicode pour le codage des caractères.

- ***Niveau syntaxique***. Ce niveau syntaxique est représenté par la définition des espaces de noms qui permettent d'identifier les ressources du Web, le langage XML, XML schéma et le langage de requêtes XML Query.

- ***Niveau Sémantique***. Ce niveau est représenté d'une part par les langages de représentation d'ontologies RDF/RDFS et OWL, et d'autre part par les langages de règles, de logique, de preuves et de confiance (trust).

5.2 Le modèle de données RDFS+

Le modèle de données que nous considérons est fondé sur le langage de représentation d'ontologies RDFS qui permet de décrire les données par des déclarations de classes et de propriétés. Nous avons choisi d'étendre RDFS par des primitives d'OWL-DL afin de pouvoir exprimer des connaissances sur le schéma telles que des axiomes de disjonction entre classes et des axiomes de fonctionnalité sur les propriétés. Nous utilisons le langage SWRL pour exprimer des axiomes sur le schéma et sur les sources de données sous forme d'un ensemble de règles de Horn. Ces axiomes ne peuvent pas être exprimés ni dans RDFS ni dans OWL-DL. Nous nommons ce modèle de données **RDFS+**. D'un point de vue "*bases de données*", RDFS+ peut être vu comme un fragment du modèle relationnel (restreint à des relations unaires et binaires) enrichi par la possibilité d'exprimer des contraintes de typage, d'inclusion ou d'exclusion entre relations, de dépendances fonctionnelles et par un ensemble de règles d'inférence de type Horn.

Nous décrivons d'abord le modèle de données RDFS+ que nous considérons, ensuite nous présentons les données et les axiomes qui leur sont associés.

5.2.1 Représentation du schéma et des axiomes du schéma

Nous considérons que nous disposons d'un schéma RDFS consistant en un ensemble de classes (relations unaires) structurées en une hiérarchie et en un ensemble de propriétés (relations binaires) qui peuvent être elles-mêmes structurées en une hiérarchie de propriétés. Les propriétés sont typées par des primitives RDFS permettant de leur associer un domaine et un co-domaine (range). Dans la terminologie RDFS, on distingue deux types de propriétés :

- ***relations***, sont des propriétés dont le domaine et le co-domaine sont des classes, et
- ***attributs***, sont des propriétés dont le domaine est une classe et le co-domaine est un type de données de base (numériques ou alpha-numériques).

On notera en relationnel :
- $R(C, D)$ pour indiquer que le domaine de la relation R est la classe C et que son co-domaine est la classe D, et
- $A(C, Literal)$ pour indiquer que l'attribut A a comme domaine la classe C et comme co-domaine un ensemble de valeurs (numériques ou alpha-numériques).

Le schéma de la figure 5.4 correspondant au domaine des lieux culturels et représenté dans une notation graphique, contient plusieurs classes organisées en hiérarchie, par exemple, la classe *Musee* subsume (est plus générale que) les classes *MuseeMoyenAge* et *MuseeContemporain*. Dans ce schéma nous avons comme relations *localiseA(Musee, Ville)*, *contient(Musee, Oeuvre)* et comme attributs *nomMusee(Musee, Literal)*, *anneeNaissance(Artiste, Date)*.

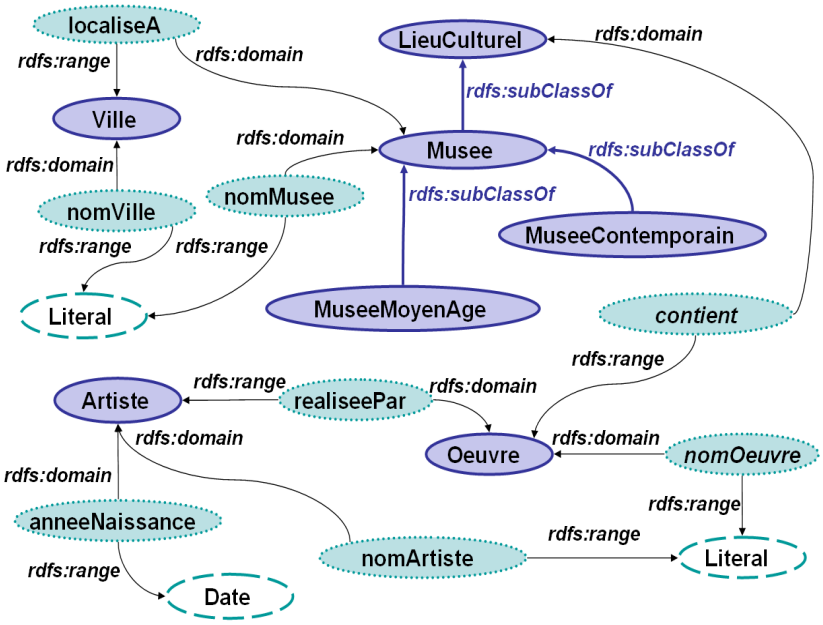

FIG. 5.4 – Exemple de schéma RDFS

Nous donnons la possibilité de déclarer des axiomes OWL-DL et des règles SWRL pour enrichir la sémantique d'un schéma RDFS. Les axiomes que nous considérons sont de plusieurs types. Nous précisons leur sémantique logique et nous donnons leur syntaxe dans le langage OWL-DL ou dans le langage SWRL.

Pour différencier l'inégalité des noms des références de la non réconciliation des objets que ces références réfèrent, nous déclarons un prédicat binaire *Reconile* qui prend en argument deux références. L'atome $Reconcile(X, Y)$ exprime le fait que les deux références X et Y forment une paire de références réconciliées, ce qui signifie qu'elles réfèrent la même entité du monde réel.

- **Axiomes de disjonction entre classes.** Nous notons $DISJOINT(C, D)$ l'axiome déclarant que les classes C et D sont disjointes, ce qui signifie que l'intersection des ensembles d'instances des classes C et D est vide. La sémantique logique est exprimée dans la règle suivante :

$$\forall X,\ C(X) \Rightarrow \neg D(X)$$

  Par exemple, un axiome de disjonction $DISJOINT(LieuCulturel, Artiste)$ est déclaré entre les classes *LieuCulturel* et la classe *Artiste* du schéma de la Figure 5.4.

  L'axiome $DISJOINT(LieuCulturel, Artiste)$ est exprimé en OWL-DL comme suit :

```
<owl:Class rdf:ID="LieuCulturel">
  <owl:disjointWith rdf:resource="#Artiste" />
</owl:Class>
```

- **Axiomes de fonctionnalité d'une propriété.** Nous notons $PF(P)$ l'axiome déclarant que la propriété P (relation ou attribut) est fonctionnelle, ce qui signifie que pour une valeur

du domaine de la propriété on peut associer au maximum une valeur du co-domaine. La sémantique logique est exprimée dans la règle suivante :

$$\forall X,\ Y,\ Z,\ P(X,Y) \wedge P(X,Z) \Rightarrow Y = Z$$

Par exemple, un axiome de fonctionnalité $PF(localiseA)$ est associé à la relation $localiseA$ du schéma de la Figure 5.4 pour exprimer la connaissance qu'un musée n'est situé que dans une seule ville.

L'axiome $PF(localiseA)$ est exprimé en OWL-DL comme suit :

```
<owl:FunctionalProperty rdf:ID="localiseA">
  <rdfs:domain rdf:resource="#Musee"/>
  <rdfs:range rdf:resource="#Ville"/>
</owl:FunctionalProperty>
```

Ces axiomes de fonctionnalité peuvent être généralisés à un ensemble $\{P_1, \ldots, P_n\}$ de relations ou d'attributs pour déclarer une combinaison des contraintes de fonctionnalité notée $\mathrm{PF}(P_1, \ldots, P_n)$, ce que l'on appelle dans le domaine des bases de données relationnelles *les dépendances fonctionnelles composées*. La sémantique logique est exprimée dans la règle suivante :

$$\forall Z,\ W \bigwedge_{i \in [1..n]} [P_i(X_i, Z) \wedge P_i(Y_i, W) \wedge (X_i = Y_i)] \Rightarrow Z = W$$

– **Axiomes de fonctionnalité de l'inverse d'une propriété.** Nous notons $PFI(P)$ l'axiome déclarant que l'inverse de la propriété P (relation ou attribut) est fonctionnelle ce qui signifie que pour une valeur du co-domaine de la propriété on peut associer au maximum une valeur du domaine. La sémantique logique est exprimée dans la règle suivante :

$$\forall X,\ Y,\ Z,\ P(Y,X) \wedge P(Z,X) \Rightarrow Y = Z$$

Par exemple, un axiome de fonctionnalité inverse $PFI(contient)$ est associé à la relation *contient* du schéma de la Figure 5.4 pour exprimer la connaissance qu'une œuvre ne peut être contenue que dans un seul lieu culturel.

L'axiome $PFI(contient)$ est exprimé en OWL-DL comme suit :

```
<owl:InverseFunctionalProperty rdf:ID="contient">
  <rdfs:domain rdf:resource="#LieuCulturel"/>
  <rdfs:range rdf:resource="#Oeuvre"/>
</owl:InverseFunctionalProperty>
```

Ces axiomes de fonctionnalité inverse peuvent être généralisés à un ensemble $\{P_1, \ldots, P_n\}$ de relations ou d'attributs pour déclarer une combinaison des contraintes de fonctionnalité inverse noté $\mathrm{PFI}(P_1, \ldots, P_n)$. La sémantique logique est exprimée dans la règle suivante :

$$\forall Z, W \bigwedge_{i \in [1..n]} [P_i(Z, X_i) \wedge P_i(W, Y_i) \wedge (X_i = Y_i)] \Rightarrow (Z = W)$$

5.2.2 Représentation des données et de leurs axiomes

Une donnée : elle est définie par un *identifiant* – une *référence* – et par une *description*. Comme les données que nous considérons sont hétérogènes et comme elles ont été créées de manière autonome alors nous ne pouvons pas faire l'hypothèse d'identifiant unique. En effet, dans le contexte d'un système d'intégration de données deux identifiants différents peuvent référer la même entité du monde réel. Nous préférons donc utiliser le terme *référence* au lieu d'identifiant. Nous dirons alors "la donnée référencée par la référence i" ou plus simplement "la référence i". Par exemple, *http ://www.lri.fr* et *http ://www.lri.fr/IdStaff/Paper123* sont des références.

La description d'une référence i : elle correspond à l'ensemble de faits RDF qui mentionnent la référence i. Nous notons $Desc(i)$ la description de la référence i. Un fait RDF est :

- soit un fait-classe de la forme $C(i)$ où C est une classe et i est une référence,
- soit un fait-relation de la forme $R(i_1, i_2)$ où R est une relation et i_1 et i_2 sont des références,
- soit un fait-attribut de la forme $A(i, v)$ où A est un attribut, i est une références et v est une valeur de base (numérique ou alpha-numérique).

Nous supposons que les données peuvent provenir de plusieurs sources et nous préfixons la référence d'une donnée par l'identifiant de la source dont elle provient.

Une *description étendue* d'une référence peut être obtenue en appliquant jusqu'à saturation les règles traduisant la sémantique des constructeurs RDFS en logique du premier ordre (voir la Table 5.1) : (i) la règle traduisant la subsomption entre classes, (ii) la règle traduisant la subsomption entre propriétés, (iii) la règle traduisant le typage du domaine d'une propriété et (iv) la règle traduisant le typage du co-domaine d'une propriété. Par exemple, à partir des faits $\{Musee(m), localiseA(m, v)\}$ nous obtenons cette description étendue $\{Musee(m), localiseA(m, v), LieuCulturel(m), Ville(v)\}$ en appliquant les règles (i) et (iv). Nous notons $EDesc(i)$ la description étendue de la référence i

Les axiomes appliqués aux données. Nous considérons deux types d'axiomes que l'on peut appliquer aux données. Ils concernent l'hypothèse du nom unique UNA (*Unique Name Assumption*) et l'hypothèse du nom unique locale LUNA (*Local Unique Name Assumption*) :

- *UNA* : elle déclare que deux données provenant de la même source et ayant deux références différentes réfèrent forcément deux entités distinctes du monde réel (et ainsi elles ne peuvent pas être réconciliées).

- *LUNA* : il s'agit d'une hypothèse plus légère que le l'UNA. Elle déclare que toutes les références associées à une même référence par une relation réfèrent forcément des entités du monde réel deux à deux différentes.
  La sémantique de la LUNA est exprimée, pour toute relation R par les deux règles suivantes :

$$\forall X,\ Y,\ Z,\ R(Z, X) \wedge R(Z, Y) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$$

$$\forall X,\ Y,\ Z,\ R(X, Z) \wedge R(Y, Z) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$$

Dans la syntaxe du langage SWRL nous avons les deux règles suivantes :

```
Implies(Antecedent(R(I-variable(Z) I-variable(X))
                   R(I-variable(Z) I-variable(Y))
                   notEqual(X, Y))
Consequent(differentFrom(I-variable(X) I-variable(Y))))

Implies(Antecedent(R(I-variable(X) I-variable(Z))
                   R(I-variable(Y) I-variable(Z))
                   notEqual(X, Y))
Consequent(differentFrom(I-variable(X) I-variable(Y))))
```

Par exemple, à partir des faits RDF $aCommeAuteur(p, a_1)$, $aCommeAuteur(p, a_2)$, $aCommeAuteur(p, a_3)$ provenant de la même source, nous pouvons inférer que les références a_1, a_2, a_3 correspondent à des auteurs deux-à-deux distincts de l'article ayant comme référence p.

Sauf mention explicite, nous posons par défaut l'axiome déclarant la *LUNA* sur chaque source.

Exemple de données représentées par des faits RDF. Soit S1 une source de données RDF conforme au schéma RDFS de la figure 5.4. Nous présentons le contenu de cette source sous forme d'un ensemble de faits RDF.

La source S1 :
LieuCulturel(S1_m1) ; Oeuvre(S1_p1) ; Artiste(S1_a1) ; Musee(S1_m2) ;
nomOeuvre(S1_p1,"La Joconde") ; realiseePar(S1_p1, S1_a1) ; contient(S1_m1, S1_p1),
nomMusee(S1_m1,"musée du LOUVRE") ; nomArtiste(S1_a1, "Leonard De Vinci") ;
anneeNaissance(S1_a1,"1452") ; Oeuvre(S1_p2) ; nomOeuvre(S1_p2,"La Cene") ;
realiseePar(S1_p2, S1_a1), Oeuvre(S1_p3) ; nomOeuvre(S1_p3,"Sainte Anne") ;
realiseePar(S1_p3, S1_a1) ; nomMusee(S1_m2, "musée des arts premiers") ;
localiseA(S1_m2, S1_c1) ; nomVille(S1_c1,"Paris") ; adresseMusee(S1_m2, "quai branly")

FIG. 5.5 – *Exemple de données RDF*

Chapitre 6

L2R : une méthode Logique pour la Réconciliation de Références

Sommaire

6.1 Problème de réconciliation de références considéré dans L2R . 128
6.2 Traduction des axiomes du schéma et des données par des règles logiques 128
6.2.1 Traduction des axiomes sur les données 129
6.2.2 Traduction des axiomes sur le schéma 129
6.3 Méthode de raisonnement pour la réconciliation de références . 131
6.3.1 Connaissances factuelles 131
6.3.2 Algorithme de réconciliation de références fondé sur la résolution . 132
6.3.3 Illustration de la méthode sur un exemple 137
6.4 Génération des dictionnaires de (non) synonymies 143
6.4.1 Dictionnaire de synonymies 144
6.4.2 Dictionnaire de non synonymies 145
6.4.3 Représentation des dictionnaires 145
6.5 Expérimentation 145
6.5.1 Présentation des jeux de données testés 146
6.5.2 Evaluation des résultats 147
6.6 Conclusion 150

La méthode Logique pour la Réconciliation de Références (L2R) que nous décrivons dans ce chapitre est une méthode dont la caractéristique est d'être fondée sur la logique, d'être *déclarative* et d'être *globale*. Dans cette méthode, les axiomes du schéma et des données sont traduits automatiquement en un ensemble de règles d'inférence de la logique du premier ordre qui va permettre d'inférer des réconciliations et des non réconciliations entre paires de références. L'ajout et la suppression d'axiomes du schéma et des données peuvent être effectués sans aucun impact sur l'algorithme de réconciliation de références. Notre méthode est globale car les règles logiques permettent de propager des nouvelles inférences à d'autres paires de références. L2R infère des réconciliations et des non réconciliations qui sont certaines grâce aux règles logiques.

Ce chapitre est organisé de la manière suivante. Dans la Section 6.1 nous présentons le problème de réconciliation de références considéré dans L2R. En Section 6.2 nous montrons comment les axiomes du schéma et des données sont traduits en un ensemble de règles d'inférence.

En Section 6.3 nous présentons l'algorithme d'inférence que nous utilisons pour la réconciliation de références et que nous illustrons ensuite sur un exemple. En Section 6.4 nous parlons des dictionnaires de synonymies et de non synonymies générés par L2R. Enfin, nous présentons en Section 6.5 les expérimentations et les résultats obtenus par L2R sur des données du domaine du tourisme et des données du domaine des publications scientifiques.

6.1 Problème de réconciliation de références considéré dans L2R

Pour la méthode logique L2R, nous considérons la réconciliation de références comme un calcul symbolique des décisions de réconciliation et de non réconciliation.

Soient S_1 et S_2 deux sources de données conformes au même schéma RDFS+. Soient I_1 et I_2 les deux ensembles de références correspondant respectivement aux données des deux sources S_1 et S_2. Le problème de réconciliation de références entre S_1 et S_2, comme nous l'avons déjà présenté dans l'introduction de cette partie (Section II), consiste à isoler deux sous-ensembles REC et $NREC$ à partir de l'ensemble $I_1 \times I_2$ des paires de références. Ces deux sous-ensembles REC et $NREC$ regroupent respectivement les paires de références représentant une même entité du monde réel, et les paires de références représentant deux entités différentes du monde réel.

Il s'agit donc, à partir de l'ensemble $I_1 \times I_2$, de construire les ensembles REC et NREC suivants :

$$\begin{cases} REC = \{(i, i') |\ Reconcile(i, i')\} \\ NREC = \{(i, i') |\ \neg Reconcile(i, i')\} \end{cases}$$

La méthode L2R est totale si elle produit un résultat ($Reconcile(i1, i2)$ ou $\neg Reconcile(i1, i2)$) pour toute paire de références $(i_1, i_2) \in I_1 \times I_2$.

Dans la suite du manuscrit, on utilisera la notation relationnelle plutôt que la notation ensembliste : $Reconcile(i1, i2)$ (respectivement $\neg Reconcile(i1, i2)$) pour $(i1, i2) \in REC$ (respectivement pour $(i1, i2) \in NREC$).

6.2 Traduction des axiomes du schéma et des données par des règles logiques

Les connaissances associées aux données sont déclarées sous la forme d'un ensemble d'axiomes au niveau du schéma RDFS+. Ces axiomes sont traduits de manière automatique par un ensemble de règles d'inférence de réconciliation et de non réconciliation entre références. De plus, certains axiomes sont traduits par un ensemble de règles permettant d'inférer des synonymies entre valeurs de base ou sous forme de règles permettant d'inférer des non synonymies entre valeurs de base. Nous notons $\mathcal{R}$ l'ensemble de toutes ces règles.

Dans la suite de cette section, nous présentons la traduction des axiomes sur les données sous forme de règles de non réconciliation. Ensuite, nous montrons la traduction des axiomes concernant le schéma RDFS sous forme d'un ensemble de règles de réconciliation, de non réconciliation, de synonymie et de non synonymie.

6.2.1 Traduction des axiomes sur les données

Les axiomes sur les données concernent l'hypothèse du nom unique (UNA) et l'hypothèse locale du nom unique (LUNA). Nous introduisons les prédicats unaires $src1$ et $src2$ pour typer chaque référence en fonction de sa source d'origine ($srci(X)$ signifie que la référence X provient de la source Si). L'hypothèse de l'UNA au niveau des sources $S1$ et $S2$ se traduit par les quatre règles suivantes :

R1 : $src1(X) \wedge src1(Y) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$

R2 : $src2(X) \wedge src2(Y) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$

R3 : $src1(X) \wedge src1(Z) \wedge src2(Y) \wedge Reconcile(X, Y) \wedge (Y \neq Z) \Rightarrow \neg Reconcile(Z, Y)$

R4 : $src1(X) \wedge src2(Y) \wedge src2(Z) \wedge Reconcile(X, Y) \wedge (X \neq Z) \Rightarrow \neg Reconcile(X, Z)$

Les deux premières règles traduisent la non réconciliation de deux références provenant d'une même source. Les deux dernières traduisent le fait qu'en présence de l'UNA une référence provenant d'une source $S1$ (resp. $S2$) peut être réconciliée avec au maximum une référence de la source $S2$ (resp. $S1$).

L'hypothèse de la LUNA n'est pas appliquée sur la totalité de la source de données mais sur la description des références en termes de relations. Pour toute relation R, l'hypothèse de la LUNA est automatiquement traduite sous forme de deux règles $R11(R)$ et $R12(R)$ suivantes :

R11(R) : $R(Z, X) \wedge R(Z, Y) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$

R12(R) : $R(X, Z) \wedge R(Y, Z) \wedge (X \neq Y) \Rightarrow \neg Reconcile(X, Y)$

6.2.2 Traduction des axiomes sur le schéma

De la même façon, les axiomes déclarés sur le schéma sont traduits en un ensemble de règles d'inférence.

Traduction des disjonctions entre classes par des règles de non réconciliation : pour chaque paire de classes C et D déclarées disjointes dans le schéma (DISJOINT(C,D)) ou inférées comme telles par héritage, la règle suivante est générée :

$$\textbf{R5(C,D)} : \; C(X) \; \wedge \; D(Y) \; \Rightarrow \; \neg Reconcile(X, Y)$$

Traduction des axiomes de fonctionnalité des propriétés.

- Pour toute relation R déclarée comme fonctionnelle par un axiome PF(R), la règle R6.1(R) est générée. Elle traduit le fait que pour une instance de la classe du domaine de R il existe au plus une instance du co-domaine.

$$\textbf{R6.1(R)} : Reconcile(X, Y) \wedge R(X, Z) \wedge R(Y, W) \Rightarrow Reconcile(Z, W)$$

– Pour tout attribut A déclaré comme fonctionnel par un axiome PF(A), la règle R6.2(A) est générée. Elle exprime le fait que pour une instance de la classe du domaine de A il existe au plus une valeur de base appartenant au co-domaine. Le prédicat binaire $SynVals$ qui prend en arguments deux valeurs de base. Il permet d'exprimer le fait que deux valeurs de base sont synonymes ou équivalentes. Il est l'équivalent du prédicat *Reconcile* sur les valeurs de base.

$$\textbf{R6.2(A)} : Reconcile(X,Y) \wedge A(X,Z) \wedge A(Y,W) \Rightarrow SynVals(Z,W)$$

– Pour toute relation R déclarée comme fonctionnelle inverse par un axiome $PFI(R)$, la règle $R7.1(R)$ est générée. Elle exprime le fait que pour une instance de la classe du co-domaine de R il existe au plus une instance de la classe du domaine.

$$\textbf{R7.1(R)} : Reconcile(X,Y) \wedge R(Z,X) \wedge R(W,Y) \Rightarrow Reconcile(Z,W)$$

– Pour tout attribut A déclaré comme fonctionnel inverse par un axiome PFI(A), la règle R7.2(A) est générée. Elle traduit le fait que pour deux valeurs de base synonymes appartenant au co-domaine il existe au plus une instance de la classe du domaine.

$$\textbf{R7.2(A)} : SynVals(X,Y) \wedge A(Z,X) \wedge A(W,Y) \Rightarrow Reconcile(Z,W)$$

Nous générons automatiquement les règles traduisant la généralisation des axiomes de fonctionnalité à un des ensembles de propriétés $PF(P_1,\ldots,P_n)$ exprimant un axiome de fonctionnalité composés de plusieurs propriétés. Par exemple, $PF(P_1,\ldots,P_n)$, quand tous les P_i sont des relations, est traduit en la règle :

$$R6(P_1,\ldots,P_n) : \bigwedge_{i\in[1..n]} [P_i(X_i,Z) \wedge P_i(Y_i,W) \wedge Reconcile(X_i,Y_i)] \Rightarrow Reconcile(Z,W)$$

De manière analogue, nous générons des règles traduisant la généralisation des axiomes de fonctionnalité inverse à un des ensembles de propriétés $PFI(P_1,\ldots,P_n)$. Quand tous les P_i sont des relations, alors l'axiome $PFI(P_1,\ldots,P_n)$ est traduit par la règle :

$$R7(P_1,\ldots,P_n) : \bigwedge_{i\in[1..n]} [P_i(Z,X_i) \wedge P_i(W,Y_i) \wedge Reconcile(X_i,Y_i)] \Rightarrow Reconcile(Z,W)$$

Si certains P_i, dans l'ensemble $PF(P_1,\ldots,P_i,\ldots,P_n)$ ou dans l'ensemble $PFI(P_1,\ldots,P_i,\ldots,P_n)$, sont des attributs alors les atomes $Reconcile(X_i,Y_i)$ correspondants doivent être remplacés par des atomes $SynVals(X_i,Y_i)$.

Règle de transitivité : elle permet d'inférer de nouvelles réconciliation en appliquant la transitivité sur l'ensemble des réconciliations entre références déjà inférées. Sa sémantique logique est la suivante :

$$\textbf{R8} : Reconcile(X,Y) \wedge Reconcile(Y,Z) \Rightarrow Reconcile(X,Z)$$

Cette règle de transitivité est générée uniquement dans le cas où l'axiome de l'UNA n'est pas déclaré au niveau de chaque source de données. En effet, dans le cas où l'UNA est posée,

une référence ne peut être réconciliée avec plus d'une référence, ce qui rend donc la règle de transitivité non applicable.

Une règle similaire ne peut pas être générée pour exprimer une relation de transitivité sur les synonymies entre valeurs de base car, d'après Fischer [Fis73] la synonymie en linguistique n'a pas la propriété de transitivité et cela à cause de la polysémie des valeurs.

6.3 Méthode de raisonnement pour la réconciliation de références

Pour inférer des réconciliations et des non réconciliations certaines entre références, nous appliquons une méthode de raisonnement automatique fondée sur le principe de *résolution* [Rob65]. Cette méthode est appliquée sur une base de connaissances représentant d'une part l'ensemble de règles, présentées dans la Section 6.2 et d'autre part l'ensemble de faits RDF concernant les références (voir Section 6.3.1).

Avant de présenter la méthode de raisonnement que nous utilisons pour inférer des (non) réconciliations certaines, nous présentons l'ensemble des faits de départ considérés.

6.3.1 Connaissances factuelles

L'ensemble $\mathcal{F}$ de connaissances factuelles est constitué de l'ensemble de faits RDF correspondant aux descriptions des références dans les sources S_1 et S_2. Ces connaissances factuelles sont complétées par un ensemble de faits générés automatiquement :

- de nouveaux faits-classes, faits-relations et faits-attributs obtenus par le calcul des descriptions étendues des références en appliquant les règles exprimant la sémantique des constructeurs RDFS, présentées en Table 5.1 : par exemple le fait *MuseeContemporain(i)* est présent dans une des sources, les faits-classes $Musee(i)$ et $LieuCulturel(i)$ sont ajoutés à la description de la référence i, par application successive des règles traduisant la subsomption entre classes.

- des faits représentant la provenance des références, de la forme $src1(i)$ et $src2(j)$ pour chaque référence $i \in I_1$ et pour chaque référence $j \in I_2$;

- des faits de synonymie de la forme $SynVals(v_1, v_2)$ pour toute paire (v_1, v_2) de valeurs de base identiques à une normalisation près. Il peut s'agir de ne pas tenir compte de la ponctuation, des variations de la casse ou des *mots vides* : par exemple, le fait *SynVals("La Joconde", "la joconde")* est ajouté car ces deux valeurs diffèrent uniquement de deux majuscules ; il peut aussi s'agir d'exploiter des méthodes de normalisation plus fines ou des ressources lexicales.

- des faits de non synonymie de la forme $\neg SynVals(v_1, v_2)$ pour toute paire (v_1, v_2) de valeurs de base distinctes d'un attribut fonctionnel (PF) et pour lesquelles il est connu qu'il n'existe pas de variations syntaxiques dans les valeurs de cette propriété ou que ces variations sont connues (e.g. les traductions). Par exemple, $\neg SynVals(''2004'','' 2001'')$, $\neg SynVals(''France'','' Algerie'')$ sont ajoutés à l'ensemble de faits.

6.3.2 Algorithme de réconciliation de références fondé sur la résolution

La résolution est un principe qui a été proposé par Alan Robinson en 1965 [Rob65] pour la démonstration automatique de théorèmes en logique propositionnelle. Ce principe consiste à déduire de nouvelles propositions à partir d'une théorie modélisée en logique propositionnelle. Le principe de résolution est appliqué sur la Forme Normale Conjonctive (CNF) d'un ensemble de formules booléennes modélisant un théorème donné. Le principe de résolution a été étendu pour la démonstration automatique en logique du premier ordre par Robinson et Wos dans [RW69].

Dans la suite, nous allons nous focaliser sur l'utilisation du principe de résolution en logique du premier ordre sur lequel notre algorithme de réconciliation de références est fondé.

Avant de présenter le principe de résolution en logique du premier ordre, introduisons tout d'abord les notions suivantes :

- ⋆ *Terme* : une constante ou une variable.
- ⋆ *Atome* : un prédicat n-aire de la forme $A(t_1, \ldots, t_n)$ où $t_1, \ldots, t_n$ sont des termes.
- ⋆ *Littéral* : un atome ou sa négation.
- ⋆ *Clause* : formule représentée sous la forme d'une disjonction de littéraux (e.g. $A \vee \neg B$).
- ⋆ *Clause Unitaire* : une clause est dite *unitaire* si elle est réduite à un seul littéral.
- ⋆ *Clause de Horn* : une clause qui comporte au plus un littéral positif.

Une forme normale conjonctive (CNF) d'un ensemble de formules est un ensemble (ou conjonction) de clauses.

Dans le cas où les clauses sont sans variables, le principe de résolution consiste à : pour chaque paire de clauses C_1 et C_2, s'il existe un littéral L_1 dans la clause C_1 et un un littéral L_2 dans la clause C_2 tel que L_1 et L_2 sont complémentaires alors supprimer les littéraux L_1 et L_2 des clauses C_1 et C_2 respectivement et construire ensuite la disjonction des littéraux restants. Cette clause construite est appelée la *résolvante* de C_1 et C_2. Cela consiste à appliquer la règle d'inférence suivante :

$$\frac{C_1 : (A \vee B)\ ,\ C_2 :\ (\neg A \vee C)}{C_{1.2} :\ (B \vee C)}$$

Avec $L_1 = A$ et $L_2 = \neg A$.

Dans le cas où les clauses contiennent des variables, le principe de résolution est similaire à celui présenté ci-dessus aux substitutions des variables près.

Définition –Substitution (cf. [CL97]). Une substitution est un ensemble fini de la forme $\{v_1/t_1, \ldots, v_n/t_n\}$, où chaque v_i, $1 \leq i \leq n$, est une variable et chaque t_i est un terme. Dans cet ensemble, une variable est substituée par au plus un terme. Soient $\sigma = \{v_1/t_1, \ldots, v_n/t_n\}$, la substitution des variables $v_1, \ldots, v_n$ par les termes $t_1, \ldots, t_n$ et e une expression. On note $e\sigma$ l'expression obtenue à partir de e en remplaçant toute variable v_i de e par le terme t_i. $e\sigma$ est ainsi appelée une instance de e.

Soient $\sigma_1 = \{x_1/t_1, \ldots, x_n/t_n\}$ et $\sigma_2 = \{y_1/s_1, \ldots, y_n/s_n\}$ deux substitutions disjointes. La composition de σ_1 et de σ_2 est une substitution, notée $\sigma_1 \circ \sigma_2$ qui est obtenue de la manière suivante :

$$\{x_1/t_1\sigma_2, \ldots, x_n/t_n\sigma_2, y_1/s_1, \ldots, y_n/s_n\}$$

Définition –Unificateur. Une substitution σ unifie deux expressions e_1 et e_2 si et seulement si $e_1\sigma = e_2\sigma$. Les deux expressions sont dites *unifiables* et σ est appelée leur *unificateur*.

L'ensemble de tous les unificateurs de deux expressions est caractérisé par le plus grand unificateur. Un unificateur σ de deux expressions e_1 et e_2 est l'*unificateur le plus général* si et seulement si pour chaque unificateur σ' il existe une substitution σ'' telle que $\sigma = \sigma' \circ \sigma''$.

Nous allons maintenant donner la définition du principe de résolution générale en logique du premier ordre appliqué sur des clauses de Horn avec ou sans variables. Pour deux clauses C_1 et C_2, s'il existe un littéral L_1 dans la clause C_1 et un littéral L_2 dans la clause C_2 tel que $L_1\sigma$ et $L_2\sigma$ sont complémentaires et σ est leur plus grand unificateur alors supprimer les littéraux $L_1\sigma$ et $L_2\sigma$ des clauses $C_1\sigma$ et $C_2\sigma$ respectivement et construire ensuite la disjonction des littéraux restants.

$$\frac{C_1 : (L_1 \vee B)\ ,\ \ C_2 : (L_2 \vee C)}{C_{1.2} : (B\sigma \vee C\sigma)}$$

Avec $L_1\sigma = \neg L_2\sigma$.

La nouvelle clause inférée $B\sigma \vee C\sigma$ est nommée la clause *résolvante* et le littéral $A\sigma$ est nommé le littéral *résolu*.

On note que dans le cas général où les clauses peuvent ne pas être de Horn, pour prouver un théorème, une deuxième règle est nécessaire. Il s'agit de la *règle de factorisation* qui est exprimé de la manière suivante :

$$\frac{(A \vee A \vee C)}{(A\sigma \vee C\sigma)}$$

Cependant, dans le cas de notre base de connaissances qui est sous la forme d'un ensemble de clauses de Horn, cette règle de factorisation n'est pas nécessaire.

Exemple : soit l'énoncé suivant : « *deux références provenant de la même source sont forcément différentes et les références (i1, i2) sont des références qui proviennent de la même source.* » Cet énoncé nous donne l'ensemble de formules suivant :

$$\{(src1(X) \wedge src1(Y) \Rightarrow \neg Reconcile(X, Y)),$$

$$src1(i1),$$

$$src1(i2)\}$$

Ce qui est équivalent à l'ensemble de clauses de Horn en forme CNF $\{C_1, C_2, C_3\}$:

$$\{C_1 : (\neg src1(X) \vee \neg src1(Y) \vee \neg Reconcile(X, Y))\ ,\ C_2 : src1(i1)\ ,\ C_3 : src1(i2)\}$$

Soit $\sigma_1 = \{X/i1\}$ et en appliquant la règle de résolution entre les clauses C_1 et C_2 sur le littéral $\neg src1(X)$, nous obtenons alors la clause résolvante $C_{1.2}$ suivante :

$$C_{1.2} : (\neg src1(Y) \vee \neg Reconcile(i1, Y))$$

Soit $\sigma_2 = \{Y/i2\}$ et en appliquant la règle de résolution entre les clauses $C_{1.2}$ et C_3 sur le littéral $\neg src1(Y)$, nous obtenons alors la clause résolvante $C_{1.2.3}$ suivante :

$$C_{1.2.3} : \neg Reconcile(i1, i2)$$

La résolution est une méthode de raisonnement qui permet de donner une preuve d'un théorème, s'elle existe, en appliquant successivement la règle de résolution sur l'ensemble de clauses exprimant ce théorème. La résolution est une méthode de raisonnement, la plupart du temps utilisée pour prouver un théorème par réfutation : on applique alors successivement la règle de résolution sur l'ensemble de clauses obtenues en niant le théorème à prouver, jusqu'à obtenir la clause vide. En revanche, l'efficacité du calcul n'est pas garantie car certaines résolutions ne sont pas forcément nécessaires pour aboutir à la preuve. Des stratégies de résolution ont été proposées afin de limiter le nombre de résolutions et par conséquent le nombre de résolvantes calculées pour aboutir à une preuve du théorème (pour plus de précisions sur ces stratégies voir [CL97, RN95]).

Pour résoudre notre problème de réconciliation de références nous cherchons à inférer tous les faits unitaires de type $Reconcile(i, j)$, $\neg Reconcile(i, j)$, $SynVals(v1, v2)$ et $\neg SynVals(v1, v2)$ où i et j sont des références et où $v1$ et $v2$ sont des valeurs de base. Pour ce faire nous avons choisi d'utiliser la stratégie de *résolution unitaire*, définie ci-après. En effet, comme nous nous intéressons à l'inférence de l'ensemble des clauses unitaires qui n'est qu'un sous ensemble de toutes les résolvantes possibles et que dans notre base de connaissances nous avons beaucoup de clauses unitaires, il s'agit donc d'une stratégie appropriée à notre problème.

Définition – Résolution unitaire : il s'agit d'une stratégie de résolution où au moins une des deux clauses impliquées dans la résolution est une clause unitaire, i.e. réduite à un seul littéral.

$$\frac{C_1 : (L_1)\ ,\ \ C_2 : (L_2 \vee C)}{C_{1.2} : (C\sigma)}$$

Avec $L_1\sigma = \neg L_2\sigma$.

Avec, la condition sur le fait qu'au moins une des deux clauses C_1 ou C_2 est unitaire. On note que l'application de la résolution unitaire conserve les conditions de l'application de la résolution générale, c'est-à-dire, l'existence de deux littéraux unifiables et complémentaires dans C_1 et C_2.

Pour l'inférence des réconciliations et des non réconciliations entre références, l'algorithme que nous proposons utilise la résolution unitaire. Elle est appliquée sur l'union de l'ensemble de règles $\mathcal{R}$ traduisant l'ensemble des axiomes et de l'ensemble de faits $\mathcal{F}$ portant sur les références (cf. Section 6.3.1). L'objectif attendu par cet algorithme est d'inférer tous les faits de réconciliation, de non réconciliation, de synonymie et de non synonymie qui sont une conséquence logique de $\mathcal{R} \cup \mathcal{F}$. Le raisonnement est fondé sur la sémantique standard de la logique du premier ordre.

Il est important de remarquer que les règles de réconciliation, bien qu'elles aient des conclusions négatives, restent sous forme de clauses de Horn. La méthode de preuve par résolution unitaire est incomplète dans le cas général elle est cependant complète pour la réfutation[22] sur des clauses de Horn [HW74]. De plus, la méthode de preuve par résolution unitaire est linéaire par rapport à la taille de l'ensemble de clauses [FdK93].

La complétude de la méthode de raisonnement utilisée.

Pour prouver la complétude de notre méthode de raisonnement pour la réconciliation de références, utilisant la résolution unitaire, nous devons prouver le théorème que nous énonçons ci-dessous.

Théorème – Complétude de la résolution unitaire sur des clauses de Horn. Soit un ensemble $\mathcal{R}$ de clauses de Horn sans fonction et un ensemble $\mathcal{F}$ de clauses unitaires instanciées (sans variables). La stratégie de preuve par résolution unitaire est complète pour la déduction logique de clauses unitaires instanciées, c'est-à-dire, que pour tout ensemble $\mathcal{R}$ de clauses de Horn sans symboles de fonctions et tout ensemble $\mathcal{F}$ de clauses unitaires instanciées, telles que $\mathcal{R} \cup \mathcal{F}$ est satisfiable, nous avons pour toute clause unitaire instanciée $p(\vec{a})$, où $\vec{a}$ un vecteur de constantes :

$$(\mathcal{R} \cup \mathcal{F} \models p(\vec{a})) \Rightarrow (p(\vec{a}) \in SatUnit(\mathcal{R} \cup \mathcal{F})),$$

où $SatUnit(\mathcal{R} \cup \mathcal{F})$ est l'ensemble des clauses unitaires instanciées contenues dans $\mathcal{F}$ ou produites par l'application de la résolution unitaire sur l'ensemble $\mathcal{R} \cup \mathcal{F}$ jusqu'à saturation.

Preuve. Soit $L_1(\vec{a})$ une clause unitaire instanciée, telle que $\mathcal{R} \cup \mathcal{F} \models L_1(\vec{a})$, avec $\mathcal{R} \cup \mathcal{F}$ est satisfiable. $\mathcal{R} \cup \mathcal{F} \cup \{\neg L_1(\vec{a})\}$ est donc insatisfiable.

Comme la résolution unitaire est complète pour la réfutation pour les clauses de Horn, il existe une preuve par résolution unitaire qui aboutit à la clause vide [HW74]. Le nombre de résolutions nécessaires pour la déduction de la clause vide représente ce que l'on appelle la *profondeur de la preuve*.

Nous montrons par récurrence sur la profondeur p d'une telle preuve que $L_1(\vec{a})$ appartient à $SatUnit(\mathcal{R} \cup \mathcal{F})$.

- ⋆ **p=1** : une seule résolution unitaire est appliquée pour déduire la clause vide. Si cette résolution ne fait pas intervenir $\neg L_1(\vec{a})$, cela signifie que $\mathcal{R} \cup \mathcal{F}$ est insatisfiable ce qui est supposé faux. Nous déduisons donc que la résolution unitaire est forcément appliquée entre $\neg L_1(\vec{a})$ et $L_1(\vec{a})$, et donc $L_1(\vec{a})$ appartient à $\mathcal{F}$ qui est inclus dans l'ensemble des faits unitaires $SatUnit(\mathcal{R} \cup \mathcal{F})$.

- ⋆ **Hypothèse de récurrence** : s'il existe une preuve de profondeur $p' \leq p$ de la clause vide en appliquant la résolution unitaire à l'ensemble $\mathcal{R} \cup \mathcal{F} \cup \{\neg L_1(\vec{a})\}$ alors $L_1(\vec{a})$ appartient à l'ensemble $SatUnit(\mathcal{R} \cup \mathcal{F})$.

[22] Prouver par réfutation qu'un littéral L est conséquence logique d'une théorie T revient à prouver l'insatisfiabilité de la théorie $T \cup \{\neg L\}$, i.e. déduire la clause vide.

Considérons une preuve de profondeur $p+1$ de la clause vide obtenue en appliquant la résolution unitaire à $\mathcal{R} \cup \mathcal{F} \cup \{\neg L_1(\vec{a})\}$.

Considérons la première étape de cette preuve, en prenant le cas où $L_1(\vec{a})$ est un atome. On note qu'un raisonnement similaire peut être effectué pour la cas où $L_1(\vec{a})$ est un atome négatif. Considérons donc la preuve pour le cas où $L_1(\vec{a})$ est un atome positif dans laquelle nous distinguons deux cas :

- ⋆ Cas 1 : la première étape de résolution ne fait pas intervenir le littéral $\neg L_1(\vec{a})$, il s'agit donc d'une résolution unitaire entre les clauses $L_2(\vec{b})$ et $\neg L_2(\vec{X}) \vee C$ qui produit la clause résolvante $C\sigma$, où $\sigma = \{\vec{X}/\vec{b}\}$ une substitution. Cette clause résolvante est nécessairement de Horn et il existe une preuve de profondeur p de la clause vide en appliquant la résolution unitaire $\mathcal{R} \cup \mathcal{F} \cup \{C\sigma\} \cup \{\neg L_1(\vec{a})\}$. $\mathcal{R} \cup \mathcal{F}$ qui est supposé satisfiable, $C\sigma$ est une conséquence logique de $\mathcal{R} \cup \mathcal{F}$ alors $\mathcal{R} \cup \mathcal{F} \cup \{C\sigma\}$ est aussi satisfiable. Par hypothèse de récurrence $L_1(\vec{a})$ appartient à $SatUnit(\mathcal{R} \cup \mathcal{F} \cup \{C\sigma\} \cup \{\neg L_1(\vec{a})\})$ qui est inclus dans $SatUnit(\mathcal{R} \cup \mathcal{F})$.

- ⋆ Cas 2 : la première étape de résolution fait intervenir $\neg L_1(\vec{a})$, il s'agit donc d'une résolution unitaire entre $\neg L_1(\vec{a})$ et $L_1(\vec{X}) \vee C$ où C est nécessairement une disjonction de littéraux tous négatifs, puisque $L_1(\vec{X}) \vee C$ est une clause de Horn. La clause résolvante est $C\sigma'$, où $\sigma' = \{\vec{X}/\vec{a}\}$ est une substitution.

  La seconde étape de la preuve est donc une résolution entre $L_2(\vec{b})$ et $\neg L_2(\vec{X})\sigma'$ où $\sigma' = \{\vec{X}/\vec{a}\}$ et $\neg L_2(\vec{X})\sigma'$ est un littéral de $C\sigma'$ qui peut être totalement ou partiellement instancié par la substitution σ'.
  Considérons la clause C_2 contenant à la fois le littéral $L_1(\vec{X})$ et $\neg L_2(\vec{X})$, elle est écrite de la façon suivante : $C_2 = L_1(\vec{X}) \vee \neg L_2(\vec{X}) \vee C$ où C est une clause de Horn ne contenant que des littéraux négatifs (puisque C_2 est une clause de Horn). La résolvante de cette seconde étape est donc $C\sigma' \circ \sigma$.
  Afin de montrer que pour ce deuxième cas on peut se ramener au premier cas, nous devons vérifier que l'ordre d'application des unificateurs σ et σ' n'a pas d'impact sur la clause résolvante obtenue après les deux premières étapes de la preuve par résolution unitaire, i.e. on obtient la même clause. Ce qui revient à montrer que :

  $$C\sigma \circ \sigma' = C\sigma' \circ \sigma$$

  Il est trivial de montrer que la composition des substitutions de variables par des constantes est computative.
  Ainsi, les deux étapes de résolution peuvent être échangées tout en garantissant d'aboutir à la même résolvante. Nous pouvons donc commencer d'abord par celle qui ne fait pas intervenir le littéral $\neg L_1(\vec{a})$ ce qui nous ramène au cas 1 où l'hypothèse de récurrence reste vraie pour une preuve par résolution de profondeur $p+1$. Nous déduisons alors $L_1(\vec{a}) \in SatUnit(\mathcal{R} \cup \mathcal{F})$.

Ainsi se termine la preuve de complétude de la résolution unitaire sur des clauses de Horn.

L'algorithme d'inférence que nous avons implémenté dans notre méthode logique applique la résolution unitaire sur l'ensemble de règles mises sous forme de clauses de Horn et sur l'ensemble de faits qui sont des clauses unitaires où toutes les variables sont instanciées. Nous donnons la description générale de la méthode dans l'algorithme 5. Il consiste à calculer $SatUnit(\mathcal{R} \cup \mathcal{F})$. Sa terminaison est garantie car il n'y a pas de symboles de fonction dans les règles.

Algorithm 5 L2R : ALGORITHME GENERAL DE RECONCILIATION DE REFERENCES

Inputs : $\mathcal{R}$: ensemble de règles traduisant les axiomes mises sous forme CNF
$\mathcal{F}$: ensemble de faits

Output : $SatUnit$: ensemble de clauses unitaires instanciées contenues dans $\mathcal{F}$ ou inférées par application de la résolution unitaire.

1: $SatUnit \leftarrow \mathcal{F}$;
2: $SatUnit_{prec} \leftarrow \emptyset$;
3: **While** ($SatUnit \neq SatUnit_{prec}$) **Do**
4: $SatUnit_{prec} \leftarrow SatUnit$;
5: $C_1 = \emptyset$;
6: $C_2 = \emptyset$;
7: **While** ($C_1 = C_2$) **Do**
8: $C_1 \leftarrow$ Selectionner-Clause($\mathcal{R} \cup \mathcal{F}$) ;
9: $C_2 \leftarrow$ Selectionner-Clause($\mathcal{R} \cup \mathcal{F}$) ;
10: **End While**
11: **If** (Est-Clause-Unitaire(C_1)) OU (Est-Clause-Unitaire(C_2)) **Then**
12: $\sigma \leftarrow$ Calculer-Unificateur(C_1, C_2) ;
13: **If** ($\sigma \neq \emptyset$) **Then**
14: $C_{resolvante} \leftarrow$ Appliquer-Resolution-Unitaire(C_1, C_2, σ) ;
15: $SatUnit \leftarrow SatUnit \cup C_{resolvante}$;
16: **End If**
17: **End If**
18: **End While**
19: **Return** $SatUnit$;

D'autres raisonneurs, comme par exemple les raisonneurs de la logique de description, auraient pu être utilisés pour la dérivation des faits de réconciliation. Cependant, les logiques de description ne sont particulièrement pas appropriées pour exprimer certaines règles de reconciliation que nous considérons qui nécessitent des liaisons explicites de variables. De plus, les raisonneurs des logiques de description existants ne garantissent pas la complétude du calcul d'impliqués premiers.

6.3.3 Illustration de la méthode sur un exemple

Nous allons nous appuyer, pour cet exemple, sur les données de la figure 6.1 et sur le schéma de la figure 5.4. Nous allons considérer les axiomes définis sur le schéma RDFS et sur les sources de données S1 et S2 que nous présentons ci-dessous. Nous allons montrer comment des synonymies

et des non synonymies peuvent être inférées par la méthode L2R. À travers cet exemple, nous allons également montrer comment les décisions de réconciliation et de non réconciliation entre références se propagent.

Soient S1 et S2 deux sources de données RDF conformes au schéma RDFS montré en Figure 5.4.

La source S1 :
LieuCulturel(S1_m1) ; Musee(S1_m2) ; MuseeMoyenAge(S1_m3) ; Oeuvre(S1_p1) ;
Oeuvre(S1_p2) ; Oeuvre(S1_p3) ;Oeuvre(S1_p4) ; Oeuvre(S1_p5) ;Artist(S1_a1) ;
Artiste(S1_a2) ; Ville(S1_c1) ; nomMusee(S1_m1,*"musee du LOUVRE"*) ;
Contient(S1_m1,S1_p1) ; nomMusee(S1_m2, *"musee des arts premiers"*) ; adresseMusee(S1_m2, *"quai branly"*) ;
LocaliseA(S1_m2,S1_c1) ; nomMusee(S1_m3,*"musee de Cluny"*) ; Contient(S1_m3, S1_p4) ; nomVille(S1_c1,"Paris") ;

nomOeuvre(S1_p1,*"La Joconde"*) ; realiseePar(S1_p1,S1_a1) ; nomArtiste(S1_a1, *"Leonard De Vinci"*) ;
anneeNaissane(S1_a1,*"1452"*) ; nomOeuvre(S1_p2,*"La Cene"*) ; realiseePar(S1_p2, S1_a1) ;
nomOeuvre(S1_p3,*"Sainte Anne"*) ; realiseePar(S1_p3, S1_a1) ;
nomOeuvre(S1_p4,*"Le sacre de David"*) ; nomArtiste(S1_a2,*"Van Gogh"*) ; anneeNaissance(S1_a2,*"1853"*) ;
realiseePar(S1_p5,S1_a2) ; nomOeuvre(S1_p5,*"Abricotiers en fleurs"*) ;

La source S2 :
Musee(S2_m1) ; Musee(S2_m2) ; ContemporaryMusee(S2_m4) ; Oeuvre(S2_p1) ;
Oeuvre(S2_p2) ; Oeuvre(S2_p3) ; Oeuvre(S2_p4) ; Artiste(S2_a1) ;
Ville(S2_c1) ; Ville(S2_c2) ;

nomMusee(S2_m1,*"Le LOUVRE"*) ; LocaliseA(S2_m1,S2_c1) ; Contient(S2_m1,S2_p2) ;
Contient(S2_m1, S2_p1) ; nomMusee(S2_m2,*"Musée du quai Branly"*) ; adresseMusee(S2_m2, *"37 quai branly, portail Debilly"*) ; Contient(S2_m1,S2_p3) ; LocaliseA(S2_m2,S2_c1) ;
nomMusee(S2_m4,*"Moca"*) ; adresseMusee(S2_m4, *"81, quai Charles de Gaule"*) ;
Contient(S2_m4,S2_p4) ;

nomVille(S2_c1, *"Ville de paris"*) ; nomVille(S2_c2,*"Lyon"*) ;

nomOeuvre(S2_p2, *"Vierge aux rochers"*) ; realiseePar(S2_p2,S2_a1) ; nomArtiste(S2_a1,*"De Vinci"*) ;
anneeNaissance(S2_a1,*"1452"*) ; nomOeuvre(S2_p3, *"Sainte Anne, la vierge et l'enfant jesus"*) ; nomOeuvre(S2_p1, *"la Joconde"*) ; nomOeuvre(S2_p4, *"Chilo Aochica"*) ;

FIG. 6.1 – *L2R – Exemple de données RDF du domaine des lieux culturels.*

Les axiomes sur le schéma et sur les données

Le schéma de la figure 5.4 contient plusieurs classes concernant le domaine des lieux culturels. Par exemple, la classe *Musee* subsume la classe *MuseeMoyenAge* et la classe *MuseeContemporain*. Il contient également un ensemble de relations et d'attributs permettant de décrire ces différentes classes. La relation *localiseA* lie la classe *Musee* à la classe *Ville*. Les attributs *nomArtiste* et *anneeNaissance* sont associés à la classe *Artiste*.

Ce schéma est enrichi par les axiomes suivants :

- Un ensemble d'axiomes de disjonction entre classes :
  {DISJOINT(*MuseeMoyenAge*,*MuseeContemporain*), DISJOINT(*Oeuvre*, *Artiste*),
  DISJOINT(*LieuCulturel*, *Ville*), DISJOINT(*LieuCulturel*, *Oeuvre*)}.
  Ces axiomes expriment, par exemple, qu'un *Musee* ne peut pas être à la fois un musée du Moyen Âge et un musée contemporain.

- Un ensemble d'axiomes de fonctionnalité sur les propriétés :
  {PF(*localiseA*), PF(*realiseePar*), PF(*nomArtiste*), PF(*anneeNaissance*), PF(*nomOeuvre*),
  PF(*nomVille*), PF(*nomMusee*), PF(*adresseMusee*)}.
  Ces axiomes expriment le fait que, par exemple, une œuvre ne peut avoir plusieurs noms différents.

- Un ensemble d'axiomes de fonctionnalité inverse sur les (ensembles de) propriétés :
  PFI(*nomMusee, adresseMusee*), {PFI(*nomOeuvre*), PFI(*contient*), PFI(*nomArtiste*),
  PFI(*nomVille*)}.
  Ces axiomes expriment le fait que, par exemple, une oeuvre ne peut pas être contenue dans plusieurs lieux culturels et qu'une adresse et un nom de musée ne peuvent pas être assignés à plusieurs musées (i.e. les deux sont nécessaires pour identifier un musée).

Nous considérons que l'UNA est vérifiée dans les sources de données S1 et S2 de la Figure 6.1.

Déroulement de l'algorithme de réconciliation fondé sur la résolution unitaire. Nous allons illustrer sur cet exemple le raisonnement fondé sur la résolution unitaire, ainsi que l'inférence des réconciliations, des non réconciliations, des synonymies et des non synonymies. À partir des axiomes du schéma et des données, présentés ci-dessus, les règles d'inférence $R1, \ldots, R7$ sont générées puis transformées en forme CNF comme suit :

Clauses traduisant les axiomes de disjonction entre classes R5 :

$R5(MuseeMoyenAge, MuseeContemporain)$:
$\neg MuseeMoyenAge(X) \vee \neg\ MuseeContemporain(Y) \vee \neg Reconcile(X, Y)$
$R5(Oeuvre, Artiste) : \neg\ Oeuvre(X) \vee \neg\ Artiste(Y) \vee \neg Reconcile(X, Y)$
$R5(Oeuvre, Artiste) : \neg\ Oeuvre(X) \vee \neg\ Artiste(Y) \vee \neg Reconcile(X, Y)$
$R5(LieuCulturel, Ville) : \neg\ LieuCulturel(X) \vee \neg\ Ville(Y) \vee \neg Reconcile(X, Y)$
$R5(LieuCulturel, Oeuvre) : \neg\ LieuCulturel(X) \vee \neg\ Oeuvre(Y) \vee \neg Reconcile(X, Y)$

Clauses traduisant les axiomes de fonctionnalité sur les propriétés :

$R6.1(localiseA) : \neg\ Reconcile(X,Y) \vee \neg\ localiseA(X,Z) \vee \neg\ localiseA(Y,W) \vee Reconcile(Z,W)$
$R6.1(realiseePar) : \neg\ Reconcile(X,Y) \vee \neg\ realiseePar(X,Z) \vee \neg\ realiseePar(Y,W) \vee Reconcile(Z,W)$
$R6.2(nomArtiste) : \neg\ Reconcile(X,Y) \vee \neg\ nomArtiste(X,Z) \vee \neg\ nomArtiste(Y,W) \vee SynVals(Z,W)$
$R6.2(anneeNaissance) : \neg\ Reconcile(X,Y) \vee \neg\ anneeNaissance(X,Z) \vee \neg\ anneeNaissance(Y,W)$
$\vee SynVals(Z,W)$
$R6.2(nomOeuvre) : \neg\ Reconcile(X,Y) \vee \neg\ nomOeuvre(X,Z) \vee \neg\ nomOeuvre(Y,W) \vee SynVals(Z,W)$
$R6.2(nomVille) : \neg\ Reconcile(X,Y) \vee \neg\ nomVille(X,Z) \vee \neg\ nomVille(Y,W) \vee SynVals(Z,W)$
$R6.2(nomMusee) : \neg\ Reconcile(X,Y) \vee \neg\ nomMusee(X,Z) \vee \neg\ nomMusee(Y,W) \vee SynVals(Z,W)$
$R6.2(adresseMusee) : \neg\ Reconcile(X,Y) \vee \neg\ adresseMusee(X,Z) \vee \neg\ adresseMusee(Y,W) \vee SynVals(Z,W)$

Clauses traduisant les axiomes de fonctionnalité inverse sur les (ensembles de) propriétés :

$R7(nomMusee, adresseMusee) : \neg\ SynVals(X1,Y1) \vee \neg\ nomMusee(Z,X1) \vee \neg\ nomMusee(W,Y1)$
$\vee \neg\ SynVals(X2,Y2) \vee \neg\ adresseMusee(Z,X2) \vee \neg\ adresseMusee(W,Y2) \vee Reconcile(Z,W)$

$R7.2(nomOeuvre) : \neg SynVals(X,Y) \vee \neg\ nomOeuvre(Z,X) \vee \neg\ nomOeuvre(W,Y) \vee Reconcile(Z,W)$
$R7.1(contient) : \neg Reconcile(X,Y) \vee \neg\ contient(Z,X) \vee \neg\ contient(W,Y) \vee Reconcile(Z,W)$
$R7.2(nomArtiste) : \neg SynVals(X,Y) \vee \neg\ nomArtiste(Z,X) \vee \neg\ nomArtiste(W,Y) \vee Reconcile(Z,W)$
$R7.2(nomVille) : \neg SynVals(X,Y) \vee \neg\ nomVille(Z,X) \vee \neg\ nomVille(W,Y) \vee Reconcile(Z,W)$

Clauses traduisant les axiomes de l'UNA :

$R1 : \neg\ src1(X) \vee \neg\ src1(Y) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R2 : \neg\ src2(X) \vee \neg\ src2(Y) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R3 : \neg\ src1(X) \vee \neg\ src1(Z) \vee\ (Y = Z) \vee \neg\ src2(Y) \vee \neg\ Reconcile(X,Y) \vee \neg Reconcile(Z,Y)$
$R4 : \neg\ src1(X) \vee \neg\ src2(Z) \vee\ (X = Z) \vee \neg\ src2(Y) \vee \neg\ Reconcile(X,Y) \vee \neg Reconcile(X,Z)$

Clauses traduisant les axiomes de la LUNA sur les relations *localiseA*, *contient* **et** *realiseePar* :

$R11(localiseA) : \neg\ localiseA(Z,X) \vee \neg\ localiseA(Z,Y) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R12(localiseA) : \neg\ localiseA(X,Z) \vee \neg\ localiseA(Y,Z) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R11(contient) : \neg\ contient(Z,X) \vee \neg\ contient(Z,Y) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R12(contient) : \neg\ contient(X,Z) \vee \neg\ contient(Y,Z) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R11(realiseePar) : \neg\ realiseePar(Z,X) \vee \neg\ realiseePar(Z,Y) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$
$R12(realiseePar) : \neg\ realiseePar(X,Z) \vee \neg\ realiseePar(Y,Z) \vee\ (X = Y) \vee \neg Reconcile(X,Y)$

Remarque. Afin de simplifier le déroulement de l'algorithme sur l'exemple, nous utilisons la formulation suivante : la résolution successive de la clause $C1$ avec les faits clos (ie. clause unitaire instanciées) $f1, f2, \ldots, fn$ signifie effectivement l'application de la résolution unitaire entre la clause $C1$ et le fait $f1$ (avec l'application de l'unificateur), ce qui donne la clause résolvante $C1.f1$, ensuite l'application de la résolution entre la clause résolvante $C1.f1$ et le fait $f2$, ce qui donne la clause résolvante $C1.f1.f2$, ainsi de suite.

Nous allons maintenant illustrer sur cet exemple quelques étapes de raisonnement, en déroulant l'algorithme 5 que nous avons décrit précédemment. Nous allons également montrer l'inférence des réconciliations, des non réconciliations, des synonymies et des non synonymies ainsi que la propagation des différentes inférences.

Enrichissement de l'ensemble de faits RDF. Les faits $SynVals(\text{“}La\ Joconde''\text{,}''\ la\ joconde'')$ et $\neg SynVals(\text{“}1853'', \text{“}1452'')$ sont générés à partir des données, le premier fait est obtenu grâce à l'égalité des chaînes de caractères apparaissant dans les valeurs (à deux majuscules près), le second fait est généré car il est connu qu'a priori il n'existe pas de variations dans les valeurs de l'attribut $anneeNaissance$. Comme nous avons posé l'hypothèse de l'UNA au niveau des sources S1 et S2, nous générons donc les faits suivants : $scr1(S1_m1)$, $scr1(S1_m2)$, $scr1(S1_p1)$, $scr1(S1_p2)$, $scr2(S2_m1)$, $scr2(S2_p1)$, $scr2(S2_p2)$. Ceci nous donne l'ensemble de faits $\mathcal{F}$ de départ pour l'algorithme 5 de réconciliation de références.

Application de la résolution unitaire sur $\mathcal{R} \cup \mathcal{F}$. Toutes les résolutions possibles sont calculées sur $\mathcal{R} \cup \mathcal{F}$. Dans cet exemple, nous ne montrerons pas les clauses résolvantes intermédiaires qui contiennent des littéraux non restreints aux littéraux qui sont de la forme : $Reconcile(i, j)$, $\neg Reconcile(i', j')$, $SynVals(u, v)$, ou $\neg SynVals(u', v')$.

- Après la résolution successive de :
  - ⋆ la clause R1 avec les faits clos $scr1(S1_m1)$, $scr1(S1_m2)$ et $(S1_m1 \neq S1_m2)$ nous inférons la clause unitaire $\neg Reconcile(S1_m1, S1_m2)$,
  - ⋆ la clause R1 avec les faits clos $scr1(S1_p1)$, $scr1(S1_p2)$ et $(S1_p1 \neq S1_p2)$ nous inférons la clause unitaire $\neg Reconcile(S1_p1, S1_p2)$,
  - ⋆ la clause R2 avec les faits clos $scr2(S2_m1)$, $scr2(S2_p1)$ et $(S2_m1 \neq S2_p1)$ nous inférons la clause unitaire $\neg Reconcile(S2_m1, S2_p1)$,
  - ⋆ la clause R2 avec les faits clos $scr2(S2_p1)$, $scr2(S2_p2)$ et $(S2_p1 \neq S2_p2)$ nous inférons la clause unitaire $\neg Reconcile(S2_p1, S2_p2)$
- Après la résolution successive de :
  - ⋆ la clause $R5(LieuCulturel, Oeuvre)$ avec les faits clos $LieuCulturel(S1_m1)$ et $Oeuvre(S2_p1)$ nous inférons la clause unitaire $\neg Reconcile(S1_m1, S2_p1)$,
  - ⋆ la clause $R5(Artiste, Oeuvre)$ avec les faits clos $Artiste(S1_a1)$ et $Oeuvre(S2_p2)$ nous inférons la clause unitaire $\neg Reconcile(S1_a1, S2_p1)$,
  - ⋆ la clause $R5(MuseeMoyenAge, MuseeContemporain)$ avec les faits clos $MuseeMoyenAge$ $(S1_m3)$ et $MuseeContemporain$ $(S2_m4)$ nous inférons la clause unitaire $\neg Reconcile(S1_m3, S2_m4)$
- Après la résolution successive de la clause $R7.2(nomOeuvre)$, qui traduit l'axiome exprimant que l'attribut $nomOeuvre$ est un attribut fonctionnel inverse, avec les faits clos $nomOeuvre(S1_p1,\text{“}La\ joconde\text{”})$, $nomOeuvre(\ S2_p1,''\ La\ \ Joconde'')$ et $SynVals(''La\ \ Joconde'',\ ''la\ \ joconde'')$ nous inférons la clause unitaire $Reconcile(S2_p1, S1_p1)$

– Le même raisonnement est appliquée sur les autres clauses.
Ainsi nous obtenons un ensemble de clauses intermédiaires complètement instanciées sur lesquelles nous continuons à appliquer la résolution unitaire. L'ensemble de ces clauses sont les suivantes :

C1 : $\neg Reconcile(S1_m1, S1_m2)$,
C2 : $\neg Reconcile(S1_p1, S1_p2)$,
C3 : $\neg Reconcile(S2_m1, S2_p1)$,
C4 : $\neg Reconcile(S2_p1, S2_p2)$,
C5 : $\neg Reconcile(S1_m1, S2_p1)$,
C6 : $\neg Reconcile(S1_a1, S2_p1)$,
C7 : $\neg Reconcile(S1_m3, S2_m4)$,
C8 : $Reconcile(S2_p1, S1_p1)$,
C9 : $\neg Reconcile(S1_a2, S2_a1)$
C10 : $\neg Reconcile(S2_p1, S1_p1) \vee Reconcile(S1_m1, S2_m1)$
C11 : $\neg Reconcile(S1_p4, S2_p4) \vee Reconcile(S1_m3, S2_m4)$
C12 : $Reconcile(S1_a2, S2_a1) \vee \neg Reconcile(S1_p5, S2_p2)$
C13 : $\neg Reconcile(S1_m1, S2_m1) \vee \neg Reconcile(S2_m2, S1_m1)$
C14 : $\neg Reconcile(S1_m1, S2_m1) \vee SynVals(\text{“}musee\ du\ LOUVRE''\text{, “}LE\ LOUVRE'')$
C15 : $\neg Reconcile(S1_m1, S2_m1) \vee Reconcile(S1_c1, S2_c1)$
C16 : $Reconcile(S1_p4, S2_p4) \vee \neg SynVals(''Le\ sacre\ de\ David'',''Chilo\ Aochica'')$
C17 : $\neg Reconcile(S1_c1, S2_c1) \vee SynVals(''ville\ de\ Paris'',''Paris'')$

La règle de résolution est appliquée sur l'ensemble de ces clauses. Toute nouvelle clause inférée est systématiquement ajoutée à l'ensemble courant de clauses afin qu'elle puisse être utilisée pour d'autres résolutions.

– La résolution entre la clause $C8$ et $C10$ sur le littéral $Reconcile(S2_p1, S1_p1)$ nous permet d'inférer la clause unitaire suivante **C8.10** : $Reconcile(S1_m1, S2_m1)$

– La résolution entre la clause $C7$ et $C11$ sur le littéral $\neg Reconcile(S1_m3, S2_m4)$ nous permet d'inférer la clause unitaire suivante **C7.11** : $\neg Reconcile(S1_p4, S2_p4)$

– La résolution entre la clause $C9$ et $C12$ sur le littéral $\neg Reconcile(S1_a2, S2_a1)$ nous permet d'inférer la clause unitaire suivante **C9.12** : $\neg Reconcile(S1_p5, S2_p2)$

– La résolution entre la clause $C8.10$ et $C13$ sur le littéral $Reconcile(S1_m1, S2_m1)$ nous permet d'inférer la clause unitaire suivante **C8.10.13** : $\neg Reconcile(S2_m2, S1_m1)$

– La résolution entre la clause $C8.10$ et $C14$ sur le littéral $Reconcile(S1_m1, S2_m1)$ nous permet d'inférer la clause unitaire suivante
C8.10.14 : $SynVals(\text{“}musee\ du\ LOUVRE''\text{, “}LE\ LOUVRE'')$

– La résolution entre la clause $C9.10$ et $C15$ sur le littéral $Reconcile(S1_m1, S2_m1)$ nous permet d'inférer la clause unitaire suivante **C8.10.15** : $Reconcile(S1_c1, S2_c1)$

– La résolution entre la clause $C7.11$ et $C16$ sur le littéral $\neg Reconcile(S1_p4, S2_p4)$ nous permet d'inférer la clause unitaire suivante
C7.11.16 : $\neg SynVals(''Le\ sacre\ de\ David'',''Chilo\ Aochica'')$

– La résolution entre la clause $C9.10.15$ et $C17$ sur le littéral $Reconcile(S1_c1, S2_c1)$ nous permet d'inférer la clause unitaire suivante
C8.10.15.17 : $SynVals(''ville\ de\ Paris'',''\ Paris'')$

À ce stade aucune résolution n'est encore possible. Ainsi nous avons inféré tous les faits de réconciliation, de non réconciliation, de synonymie et de non synonymie représentés par l'ensemble des clauses unitaires de $SatUnit(\mathcal{R} \cup \mathcal{F})$ suivantes :

C1 :$\neg Reconcile(S1_m1, S1_m2)$,
C2 : $\neg Reconcile(S1_p1, S1_p2)$,
C3 :$\neg Reconcile(S2_m1, S2_p1)$,
C4 :$\neg Reconcile(S2_p1, S2_p2)$,
C5 :$\neg Reconcile(S1_m1, S2_p1)$,
C6 :$\neg Reconcile(S1_a1, S2_p1)$,
C7 :$\neg Reconcile(S1_m3, S2_m4)$,
C8 :$Reconcile(S2_p1, S1_p1)$,
C9 :$\neg Reconcile(S1_a2, S2_a1)$
C8.10 : $Reconcile(S1_m1, S2_m1)$
C7.11 : $\neg Reconcile(S1_p4, S2_p4)$
C9.12 : $\neg Reconcile(S1_p5, S2_p2)$
C8.10.13 : $\neg Reconcile(S2_m2, S1_m1)$
C8.10.14 : $SynVals(\text{“}musee\ du\ LOUVRE'', \text{“}LE\ LOUVRE'')$
C8.10.15 : $Reconcile(S1_c1, S2_c1)$
C7.11.16 : $\neg SynVals(''Le\ sacre\ de\ David'',''\ Chilo\ Aochica'')$
C8.10.15.17 : $SynVals(''ville\ de\ Paris'',''\ Paris'')$

Nous avons vu comment les différentes inférences sont propagées à travers les clauses. Par exemple, l'inférence de la réconciliation des deux musées $Reconcile(S1_m1, S2_m1)$ a été propagée et a permis l'inférence de la réconciliation des deux villes de leur localisation $Reconcile(S1_c1, S2_c1)$ grâce à la clause traduisant la $PF(localiseA)$. Cette nouvelle réconciliation, à son tour, a été propagée et a permis l'inférence de la synonymie entre les valeurs de base $SynVals(''ville\ de\ Paris'',''\ Paris'')$ grâce à la clause traduisant la $PF(nomVille)$.

6.4 Génération des dictionnaires de (non) synonymies

L'ensemble de synonymies et de non synonymies inféré entre les valeurs de base par l'algorithme de réconciliation de références est conservé dans ce que nous appelons dictionnaire de synonymies et dictionnaire de non synonymies. Ces dictionnaires sont enrichis au fur et à mesure de l'application de la réconciliation de références. Ils peuvent être ré-utilisés lors des applications de L2R pour la réconciliation d'autres sources de données : les synonymies et les non synonymies peuvent être injectées à l'initialisation de l'ensemble de faits comme $SynVals(v, v')$ et $\neg SynVals(v, v')$. Ces synonymies et non synonymies peuvent également être utilisées par une méthode numérique de réconciliation de références, telle que la méthode N2R (présentée en chapitre 7) qui calcule des scores de similarité pour les différentes paires de références.

Les synonymies et les non synonymies entre valeurs peuvent être vues comme des connaissances apprises de manière automatique et non supervisée et qui peuvent être utiles pour les

méthodes de réconciliation de références. En effet, il s'agit d'un moyen par lequel notre méthode capitalise son expérience en apprenant de plus en plus de connaissances sur les variations syntaxiques et les homonymes qui caractérisent un domaine d'application donné.

6.4.1 Dictionnaire de synonymies

Les synonymies entre valeurs de base sont inférées grâce aux règles (clauses) traduisant les axiomes de fonctionnalité sur les attributs. Parmi ces synonymies, nous distinguons différents types :

- **Codifications** : qui correspondent à des synonymies entre valeurs qui peuvent être syntaxiquement différentes, mais qui représentent des vocabulaires équivalents pour représenter certaines données. Par exemple, on peut représenter : les étoiles d'un hotel soit par le symbole *(*)* soit par le mot *star*, représenter une valeur de vérité soit par *1* soit par *oui* ou encore représenter une ville soit par son nom *Paris* soit par son code postal *75*.

- **Abréviations et acronymes** : qui correspondent à des synonymies entre valeurs qui représentent une équivalence entre une valeur abrégée ou simplifiée et une valeur plus longue. Par exemple, *apt* est l'abréviation d'*appartement* ou "*ACM*" est l'acronyme de "*Association for Computing Machinery*".

- **Des vrais synonymes** : des valeurs qui représentent des synonymes au sens linguistique du terme, comme par exemple "*agréable*" pour "*confortable*".

- **Traductions** : des valeurs qui représentent la même donnée mais qui sont décrites dans deux langues différentes. Ce type de synonymes est inféré lorsqu'on traite des sources de données qui sont décrites dans des langues différentes. Par exemple, la synonymie entre "*United Kingdom*" et "*Royaume-Uni*" pour un nom de pays.

À partir de certaines synonymies entre des valeurs contenant des abréviations, des acronymes ou des variations syntaxiques, des mises en correspondance plus précises pourraient être obtenues par un traitement plus fin sur les valeurs. Par exemple, considérons la synonymie obtenue entre des noms de conférences *SynVals(*"in proceedings of the twenty-fifth annual acm symposium on theory of computing*", "*in proc. 25th annu. acm sympos. theory comput*")*. Si on traite de manière plus fine cette synonymie, on peut extraire certaines mises en correspondance comme des abréviations ("*proceedings*", "*proc*"), ("*annual*", "*annu*") et ("*symposium*", "*sympos*"). On peut extraire également certaines variations syntaxiques comme ("*twenty-fifth*","*25th*"). On peut envisager d'apprendre des patterns d'équivalence sur les mises en correspondance –mappings – que l'on peut représenter sous la forme d'expressions régulières. Par exemple, pour le mapping ("*twenty-fifth*","*25th*") un pattern peut être écrit sous la forme :

$$\Big(< [a-z]^*(-)[a-z]^*([st][nd][rd][th]) > \approx < [0-9]^*([st][nd][rd][th]) > \Big)$$

Ce pattern représente la correspondance possible entre une expression commençant par un mot suivi d'un trait d'union et qui se termine par *st*, *nd*, *rd* ou *th* et une expression une expression commençant par un nombre et qui se termine par *st*, *nd*, *rd* ou *th*.

Les synonymies entre valeurs de base peuvent être exploitées, par une méthode numérique, en leur assignant un score de similarité maximal (e.g. 1). Le nombre de synonymies inférées par L2R peut parfois être très grand. Nous ne nous intéressons donc qu'aux synonymies qui sont pertinentes. En effet, il est d'autant plus intéressant d'exploiter des synonymies que leurs valeurs sont syntaxiquement différentes (e.g. Monalisa vs. Joconde). De plus, les mesures de similarité (comme celles présentées en Section 4.2) donnent un score de similarité faible pour ce type de synonymies.

6.4.2 Dictionnaire de non synonymies

Les non synonymies entre valeurs sont inférées grâce aux règles (clauses) traduisant les axiomes de fonctionnalité sur les attributs. Ces non synonymies entre valeurs de base peuvent être exploitées, par une méthode numérique, en leur assignant un score de similarité minimal (e.g. 0). Comme c'est le cas pour les synonymies, le nombre des non synonymies inférées peut être très grand. C'est pour cela, que nous effectuons une sélection et nous ne conservons que celles qui sont pertinentes. En effet, les non synonymies intéressantes sont celles qui sont entre valeurs, qui sont syntaxiquement proches (ICDE vs. ICDT) et pour lesquelles les mesures de similarité donnent des scores de similarité élevés.

6.4.3 Représentation des dictionnaires

Les synonymies et les non synonymies sont, pour ainsi dire, contextuelles : elles dépendent de l'attribut qui a permis leur inférence. Par exemple, *SynVals("Joconde", "Monalisa")*, inférée par la règle traduisant l'axiome de fonctionnalité de l'attribut *nomOeuvre*, ne doit pas être utilisé pour la réconciliation de deux hôtels ayant comme valeur respectivement *"Joconde"* et *"Monalisa"* pour l'attribut *nomHotel*. C'est pour cela que nous conservons dans le dictionnaire pour chaque synonymie et pour chaque non synonymie le nom de la propriété qui a permis leur inférence et qui représente le contexte de la synonymie ou de la non synonymie.

Nous représentons donc les synonymies et les non synonymies par un ensemble de triplets de la forme : $(propriete, valeur1, valeur2)$. Par exemple, la synonymie *SynVals("Joconde", "Monalisa")* est représentée par le triplet : $(nomOeuvre,'' Joconde'','' Monalisa'')$

6.5 Expérimentation

La méthode de réconciliation de références L2R fondée sur la logique que nous avons présentée dans les sections précédentes a été implémentée et expérimentée sur des jeux de données concernant deux domaines différents : des données concernant le domaine du tourisme et des données collectées à partir d'un portail de publications scientifiques en informatique.

Dans le premier jeu de données, la réconciliation de références est appliquée dans le but d'intégrer sept sources de données différentes décrivant des instances d'une seule classe qui est la classe *Hotel*. L'hypothèse du nom unique UNA est vérifiée au niveau de chaque source. Le problème de réconciliation de références est posé sur vingt-et-une (21) paires de sources de données, car les sept sources de données doivent être toutes réconciliées entre elles.

Dans le second jeu de données, la réconciliation de références est appliquée dans le but de nettoyer une source contenant des données redondantes. Dans ce cas, le problème de réconciliation de références est posé sur un ensemble de références $I \times I$ où I représente l'ensemble de références de la source S qui doit être nettoyée. Pour cette source de données l'UNA ne peut pas être

posée. Les données de cette source décrivent des références appartenant à trois classes : *Article*, *Conference* et *Person*.

6.5.1 Présentation des jeux de données testés

Le premier jeu de données FT_HOTELS, fourni par notre partenaire France Telecom R&D dans le cadre du projet PICSEL3, est composé d'un ensemble de sept sources de données. Ces sources de données contiennent 28934 références d'hôtels localisés en Europe. L'hypothèse de l'UNA est posée pour chaque source. Le schéma RDFS conçu pour décrire les données du tourisme est présenté en Figure 6.2.

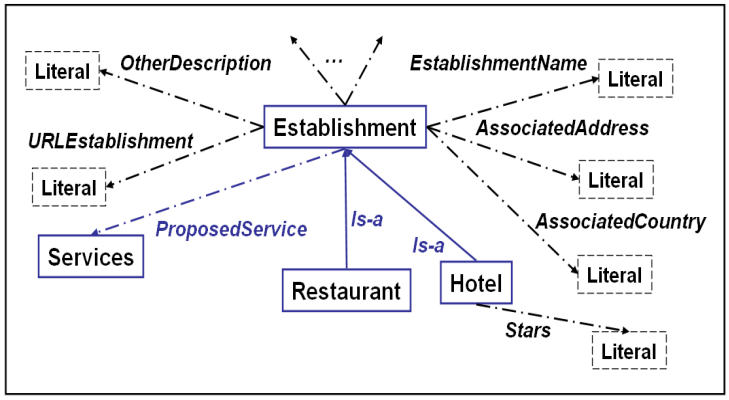

FIG. 6.2 – Schéma RDFS des données FT_HOTEL

Afin de rendre conformes ces différentes sources de données au schéma RDFS (Figure 6.2), nous avons écrit un adaptateur pour chacune des sources.

Nous avons enrichi le schéma RDFS par un ensemble d'axiomes :
- *axiome de disjonction* : $DISJOINT(Establishment, Service)$
- *axiomes de fonctionnalité sur les propriétés* : toutes les propriétés sont fonctionnelles, sauf *OtherService*, *OtherDescription* et *URLEstablishment* ;
- *axiomes de fonctionnalité inverse des propriétés* : nous considérons un seul axiome PFI(*EstablishmentName*, *AssociatedAddress*) qui exprime le fait que la combinaison des deux attributs *EstablishmentName* et *AssociatedAddress* forment une fonctionnalité inverse : deux établissements doivent être réconciliés s'ils possèdent le même nom et la même adresse.

Le vocabulaire utilisé pour la description des hôtels dans les différentes sources est très hétérogène : d'une part, l'ensemble des propriétés instanciées diffère d'une source à une autre, en particulier, la propriété *AssociatedAddress* faisant partie de l'axiome de fonctionnalité inverse n'est pas toujours renseignée, d'autre part, les valeurs sont parfois multilingues, contiennent des abréviations, etc.

Le second jeu de données est le benchmark Cora[23] qui a été utilisé entre autres par Dong et

[23] une autre version de Cora est rendue disponible par McCallum, (http ://www.cs.umass.edu/ mccallum/data/cora-refs.tar.gz)

al. [DHM05] et par Parag et Domingos [SD05]. Il s'agit d'une collection de 1295 citations de 112 articles de recherche différents en informatique. Ces citations ont été collectées à partir du moteur de recherche Cora spécialisé dans la recherche d'articles scientifiques en informatique. Une référence est associée à chaque citation d'article, à chaque conférence et à chaque auteur (personne).

Les données de Cora sont conformes au schéma RDFS donné en Figure 6.3.

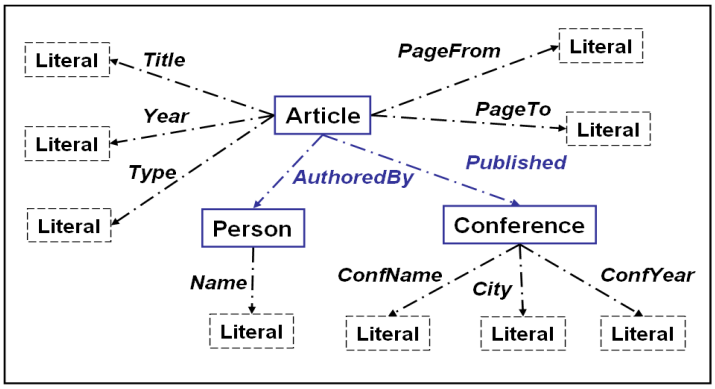

FIG. 6.3 – Schéma RDFS des données de Cora

Nous avons déclaré au niveau du schéma RDFS+ l'ensemble d'axiomes suivant :

- *axiomes de disjonction entre classes* : les trois classes (*Article*, *Conference* et *Person*) du schéma RDFS sont mutuellement disjointes. Ceci est exprimé par DISJOINT(*Article*, *Conference*), DISJOINT(*Article*, *Person*) et DISJOINT(*Conference*, *Person*) ;

- *axiomes de fonctionnalité des propriétés* : toutes les propriétés sont fonctionnelles, sauf la relation *AuthoredBy*, ce qui signifie qu'un article peut avoir plusieurs auteurs ;

- *axiomes de fonctionnalité inverse des propriétés* : nous déclarons deux axiomes de fonctionnalité inverse des propriétés. Le premier concerne la classe *Article* $PFI(Title, Year, Type)$, il déclare que deux articles ayant le même titre, la même année de publication et le même type de publication (e.g. journal, proceedings, report) doivent être réconciliés. Le second concerne la classe *Conference* $PFI(ConfName, ConfYear)$, il déclare que deux conférences ayant le même nom et qui se sont déroulées la même année doivent être réconciliées ;

Nous considérons par défaut que l'axiome de la LUNA appliqué sur les données est vérifié.

6.5.2 Evaluation des résultats

Pour l'évaluation des résultats de la méthode L2R, nous calculons les mesures du rappel et de la précision qui sont très utilisés en recherche d'information. Comme les décisions de réconciliation et de non réconciliation sont obtenues par l'application d'une méthode fondée sur la logique, toutes les inférences sont correctes, ce qui signifie que la précision des résultats est de 100% par construction. Ainsi, la mesure qui doit être calculée pour l'évaluation des résultats de notre expérimentation est le rappel.

Pour les données du benchmark *Cora* les résultats qu'un algorithme de réconciliation de références doit trouver, sont fournis. Par conséquent, le rappel peut être facilement obtenu par

le calcul du ratio des réconciliations et des non réconciliations obtenues par L2R parmi celles qui sont fournies.

En ce qui concerne les données du tourisme FT_HOTELS, les résultats (réconciliations ou non réconciliations) attendus ne sont pas fournis. Par conséquent, nous avons validé manuellement les réconciliations et les non réconciliations correctes inférées par L2R sur une paire de sources de données contenant respectivement 404 et 1392 références d'hôtels. Afin de faciliter la détection des réconciliations, nous avons vérifié l'ensemble des réconciliations possibles par des requêtes mots-clés sur les noms et les adresses des hôtels. Pour les autres paires de sources de données nous n'avons pas de résultats qualitatifs à fournir à ce jour, mais nous fournissons les résultats quantitatifs, i.e. le nombre de réconciliations et de non réconciliations obtenues.

Dans la suite, nous allons discuter les résultats obtenus sur le jeu de données FT_HOTELS et ensuite ceux obtenus sur le jeu de données Cora. Nous mettrons l'accent sur l'impact de l'enrichissement du schéma, par des axiomes sur l'augmentation du rappel de L2R. Nous montrerons également l'intérêt (sur les données de Cora) d'avoir une approche globale dans laquelle des inférences de (non) réconciliations et de (non) synonymies sont propagées à d'autres paires de références et ainsi voir le rappel augmenter.

Résultats obtenus sur les données FT_HOTELS

Pour les résultats quantitatifs, l'application de L2R sur les 21 paires de sources de données a conduit à 1063 réconciliations et 251 523 187 non réconciliations.

Pour les sources pour lesquelles nous avons évalué manuellement les résultats corrects, nous présentons les valeurs du rappel obtenues dans la Table 6.1. Nous distinguons le rappel calculé uniquement sur l'ensemble des paires de références réconciliées (REC) de celui calculé uniquement sur l'ensemble de paires de références non réconciliées (NREC).

	RDFS+	**RDFS+ et DA**
Rappel (REC)	54 %	54 %
Rappel (NREC)	8.2 %	75.9 %
Rappel global	8.3 %	75.9 %
Précision	100 %	100 %

TAB. 6.1 – *Résultats de L2R sur les données FT_HOTELS*

Comme nous le montrons dans la colonne "RDFS+" de la Table 6.1, nous avons obtenu un rappel global de L2R de 8.3 %. Si nous considérons uniquement l'ensemble des réconciliations (REC) le rappel est de 54 %. L'ensemble REC correspond aux réconciliations inférées en exploitant l'axiome de fonctionnalité inverse sur la classe Hôtel.

Il est très important de souligner que ces réconciliations correctes (Précision = 100 %) ont été inférées malgré les irrégularités dans la description des données : l'adresse est parfois non renseignée ainsi que les nombreuses variations syntaxiques dans les valeurs, en particulier dans les adresses où le degré de granularité des valeurs n'est pas le même pour les deux sources. Par exemple, "*parc des fées*" vs. "*parc des fées, (nearby Royan)*", ou bien "*11, place d'arme*" vs. "*place d'arme*". De plus, dans une des deux sources de données, plusieurs langages sont utilisés pour décrire les valeurs, par exemple, "*Chatatoa*" versus "*Chahatoenia*" dans la langue *basque*.

Si nous considérons uniquement l'ensemble des non réconciliations (NREC) le rappel est de 8.2%. Cette valeur faible du rappel est due au fait que les seules règles qui permettent d'inférer

des non réconciliations sont celles qui traduisent l'UNA au niveau de chaque source.

Maintenant, considérons un enrichissement du schéma RDFS+ uniquement par la déclaration d'axiomes de disjonction entre des spécialisations de la classe *Hotel* en distinguant les différents hôtels par leur pays de localisation, par exemple, la classe d'hôtels *Français* est disjointe de la classes d'hôtels *Portuguais*. Dans ce cas, nous obtenons une très forte augmentation du rappel calculé sur NREC, il passe de 8.2% à 75.9%, comme nous le présentons dans la colonne "RDFS+ et DA" de la Table 6.1.

Résultats obtenus sur les données de Cora

Nous nous focalisons sur les résultats obtenus sur les références de la classe *Article* et sur les références de la classe *Conference* qui contiennent respectivement 1295 références et 1292 références.

	RDFS+	**RDFS+ et DP**
Rappel (REC)	52.7 %	52.7 %
Rappel (NREC)	50.6 %	94.9 %
Rappel global	50.7 %	94.4%
Précision	100 %	100 %

TAB. 6.2 – *Résultats de L2R sur les données Cora (classes Article et Conference)*

Comme nous le montrons dans la colonne nommée "RDFS+" de la table 6.2, le rappel global obtenu sur les données du benchmark Cora est de 50.7 %. Ce résultat peut être présenté de manière plus fine par un rappel de 52.7 % calculé sur l'ensemble REC et un rappel de 50.6 % calculé sur l'ensemble des non réconciliations NREC.

L'ensemble des réconciliations inférées (REC) pour les références d'articles sont obtenues grâce à l'axiome PFI(*Title, Year, Type*) de fonctionnalité inverse, composé des attributs *Title*, *Year* et *Type*. Pour les références de conférences, 35.8% des réconciliations sont obtenues par l'exploitation de l'axiome PFI(*ConfName, ConfYear*) de fonctionnalité inverse composé des attributs $ConfName$ et $ConfYear$, alors que 64.1% sont obtenues par la propagation des décisions de réconciliation entre paires de références d'articles, grâce à l'axiome de fonctionnalité PF(*Published*) de la relation *Published* qui exprime la connaissance qu'un article ne peut pas être publié dans plusieurs conférences.

L'ensemble des non réconciliations (ensemble NREC) inférées sont obtenues par l'exploitation de l'axiome de disjonction entre la classe *Article* et la classe *Conference*.

Pour ces données, nous pouvons enrichir l'ensemble de faits de départs par un ensemble de faits de non synonymie entre les valeurs de l'attribut fonctionnel *ConfYear*, car il est connu qu'il n'existe pas de variations syntaxiques dans les valeurs représentant l'année. L'injection de ces faits de non synonymie va conduire à l'inférence de la non réconciliation entre des conférences qui se sont déroulées lors d'années différentes (e.g. AAAI'04 ne peut pas être réconciliée avec AAAI'07). En effet, dans ce cas, le rappel calculé sur l'ensemble REC reste inchangé (52.7%) mais nous obtenons une très forte augmentation du rappel calculé sur l'ensemble NREC, il croît de 50.6 % jusqu'à 94.9% (voir la colonne (RDFS+ et DP) de la Table 6.2). Cette augmentation

très significative est due à la propagation des non réconciliations à travers les règles : les non réconciliations entre paires de références de conférences pour lesquelles les valeurs de l'attribut *ConfYear* sont différentes impliquent l'inférence des non reconciliations. Ces non réconciliations concernent des références d'articles qui sont associées à ces conférences et elles sont obtenues grâce à l'exploitation de l'axiome de fonctionnalité sur la relation *Published* PF(*Published*).

Le rappel obtenu par Dong et al. [DHM05] sur les données de Cora est de 97%. Ce rappel est uniquement calculé sur l'ensemble des réconciliations REC. Sur cet ensemble leurs résultats sont meilleurs que ceux obtenus par notre méthode L2R. Cependant, leur algorithme est fondé sur des techniques d'apprentissage supervisé, où des pondérations associées aux arcs de leur graphe de dépendance entre paires de références sont apprises en utilisant des arbres de décision (voir la section 4.4.1 du chapitre État de l'art). Notre méthode L2R est non supervisée, garantit une précision de 100% et un bon rappel.

Pour les références de la classe *Person*, l'hypothèse de la LUNA nous permet d'inférer des non réconciliations. Pour 3521 références de la classe Person, 4298 non réconciliations ont été inférées en exploitant la sémantique de l'axiome déclarant la LUNA.

L'exploitation des axiomes de disjonction (DISJOINT(*Article*, *Person*) et DISJOINT(*Conference*, *Person*)) permet d'inférer 9 108 827 non réconciliations supplémentaires qui représentent 24.4 % de l'espace de réconciliation.

Résultats d'expérimentation concernant le dictionnaire de synonymies

Pour chaque jeu de données, les synonymies entre valeurs de base inférées par l'application de la méthode L2R ont été stockées dans le dictionnaire de synonymies. Pour le jeu de données FT_HOTELS l'ensemble des 1063 réconciliations inférées (pour les sept sources de données) ont permis l'inférence de 3671 synonymies. Elles représentent principalement des codifications comme $SynVals(''1'',''Y'')$, des traductions comme $SynVals(''Florence'',''FIRENZE'')$ et des descriptions syntaxiquement proches telles que $SynVals(''2100'',''DK-2100'')$ qui sont deux codes postaux au *Danemark*, mais aussi $SynVals(''Avignon-Le\ Pontet'',''LE\ PONTET'')$.

Pour le jeu de données CORA, 2142 synonymies ont été inférées. Par exemple, pour l'attribut *City*, la synonymie $SynVals(''cambridge,\ massachusetts''\ ,\ ''cambridge,\ mas-sachusetts'')$ a été inférée, les deux valeurs différant d'un trait d'union dans la seconde valeur. Pour l'attribut *ConfName*, la synonymie $SynVals(''in\ proceedings\ of\ the\ twenty-fifth\ annual\ acm\ symposium\ on\ theory\ of\ computing'',''in\ proc.\ 25th\ annu.\ acm\ sympos.\ theory\ comput'')$ a été inférée.

6.6 Conclusion

Nous venons de présenter dans ce chapitre la méthode logique de réconciliation de références L2R dont les principaux résultats ont été publiés dans [SPR07b, SPR07a, SPR07c, PS07]. Il s'agit d'une méthode fondée sur les connaissances déclarées sous forme d'axiomes sur le schéma et sur les données. Ces axiomes sont traduits automatiquement dans un ensemble de règles de réconciliation, de non réconciliation, de synonymie et de non synonymie qui sont ensuite utilisées par un algorithme d'inférence fondé sur la résolution SLD. L'expérimentation de L2R sur deux

jeux de données différents a montré, d'une part, l'intérêt d'une approche déclarative – en effet, nous avons vu que l'enrichissement du schéma par des axiomes peut augmenter très significativement le rappel – d'autre part, l'intérêt d'une approche globale qui exploite les relations pour la réconciliation de références, par exemple, dans le jeu de données Cora, 64.1 % des réconciliations des références de conférences ont été inférées grâce à la propagation des décisions de réconciliation des articles par le biais de la propriété fonctionnelle concernant la relation *Published*.

Pour avoir d'autres réconciliations probables, nous proposons une méthode numérique N2R (voir Chapitre 7) qui calcule des scores de similarité des différentes paires de références afin de pouvoir décider de leur réconciliation.

Les résultats obtenus par L2R sont partiels : nous n'obtenons pas de décision de réconciliation ou de non réconciliation pour toute paire de références de l'ensemble $I1 \times I2$. Néanmoins, tous les résultats de réconciliation, de non réconciliation, de synonymie et de non synonymie obtenus sont certains ce qui a permis d'avoir une précision de 100 %. En plus du rôle de ces résultats corrects dans le traitement partiel du problème de réconciliation de références, les (non) réconciliations et les (non) synonymies obtenues par L2R peuvent être exploitées pour de multiples intérêts :

- une méthode de réconciliation de références numérique telle que N2R puisse les exploiter en associant un score de similarité à 1 pour les réconciliations et les synonymies et un score de similarité à 0 pour les non réconciliations et les non synonymies.

- des méthodes supervisées, comme, par exemple, celle qui permet d'apprendre des pondérations associées aux relations et aux attributs dans un calcul numérique de la similarité des références, peuvent utiliser un échantillon de données étiqueté par les réconciliations et les non réconciliations certaines, obtenus par L2R.

- un dictionnaire de (non) synonymies peut être construit à partir des synonymies inférées par L2R. Il s'agit en quelque sorte, de conserver l'expérience de la méthode L2R dans un domaine d'application donné, ce qui lui permettra de s'améliorer au fil de son utilisation. De plus, des traitements plus fins peuvent être effectués sur le dictionnaire pour extraire des mises en correspondance plus précises entre valeurs (e.g. abréviations, acronymes), qui pourront être ensuite utilisées dans une éventuelle phase de normalisation des valeurs.

Dans le cas où un système d'intégration de données nécessite des (non) réconciliations sûres, la méthode L2R pourra fournir un premier ensemble de (non) réconciliations qui ne nécessite pas de validation manuelle. De plus, nous avons vu que malgré l'hétérogénéité des données testées, un nombre très significatif de décisions de réconciliation a été obtenu par L2R et de manière non supervisée. En effet, dans la méthode L2R, nous n'avons pas besoin d'échantillon de données étiquetées, échantillon qui doit être mis à jour à chaque fois que les caractéristiques des données changent.

Chapitre 7

N2R : une méthode Numérique pour la Réconciliation de Références

Sommaire

7.1 Problème de réconciliation de références considéré dans N2R . 154
7.2 Comparaison des paires de références 155
7.2.1 Exploitation des connaissances du domaine pour le filtrage 155
7.2.2 Description commune des paires de références 156
7.2.3 Calcul du score de similarité des valeurs et des ensembles de valeurs 157
7.3 Expression du calcul de similarité dans un système d'équations 159
7.4 Calcul de similarité informé par les connaissances du schéma . 162
7.4.1 Exploitation des axiomes de fonctionnalité du schéma 162
7.4.2 Construction du système d'équations non linéaire 165
7.4.3 Résolution du système d'équations non linéaire 170
7.4.4 Illustration du calcul itératif sur un exemple 172
7.5 Algorithme de calcul itératif de la similarité des références . . . 175
7.5.1 Présentation de l'algorithme général 175
7.5.2 Complexité de l'algorithme du calcul itératif de similarité 181
7.6 Expérimentations 182
7.6.1 Résultats de N2R obtenus sur le jeu de données *Cora* 183
7.6.2 Résultats de N2R obtenus sur le jeu de données FT_HOTELS . . . 184
7.7 Conclusion 186
7.1 Résumé des contributions et discussion 187
7.2 Travaux en cours et perspectives 190

La méthode Numérique pour la Réconciliation de Références (N2R) que nous décrivons dans ce chapitre est une méthode qui permet de calculer un score de similarité pour chaque paire de références. Cette méthode, est *déclarative*, *informée*, *globale* et fondée sur la résolution d'un système d'équations non linéaire. Notre méthode est *globale* car dans le calcul de similarité des paires de références, nous exploitons, non seulement les attributs décrivant les références, mais aussi les relations (e.g. $ecritPar(Article, Personne)$, $realiseePar(Oeuvre, Artiste)$) qui les lient entre elles. Ce calcul est exprimé dans une *équation* qui exprime que la similarité de deux références est fonction des scores de similarité des paires (ensembles) de valeurs de leurs attributs mais aussi en fonction des scores de similarité des paires (ensembles) de références qui leur sont

liées par les relations. L'expression du calcul de similarité des paires de références dans un système d'équations nous permet de tirer profit de l'expérience des nombreux travaux réalisés pour la résolution des systèmes d'équations. Nous proposons, pour notre calcul, l'utilisation d'une méthode itérative de résolution de systèmes d'équations inspirée de la méthode de Jacobi [GL89]. Le calcul de similarité des paires de références que nous proposons est un calcul *informé* qui est guidé par les connaissances du schéma. En effet, des connaissances sur l'importance variable des différents attributs et des différentes relations dans le calcul de similarité de deux références sont nécessaires. Dans des travaux tels que Debabrata et al. [DSD98b] et Dong et al [DHM05], ces connaissances sont codées dans des pondérations (des valeurs réelles dans [0..1]) spécifiées par un expert du domaine ou apprises sur un échantillon de données étiquetées comme réconciliées ou non réconciliées. Dans notre méthode N2R, nous souhaitons comme pour la méthode L2R être guidés au mieux par les connaissances du domaine déclarées dans le schéma. Ainsi, pour déterminer les différents impacts des relations et des attributs sur la similarité des paires de références, nous exploitons les axiomes de fonctionnalité et de fonctionnalité inverse déclarés dans le schéma RDFS+. De plus, des connaissances déclarées sous forme d'un ensemble d'axiomes de disjonction entre les classes du schéma RDFS+ et des axiomes exprimant l'UNA et la LUNA sur les données, sont également prises en compte dans notre calcul de similarité dans un but de filtrage des paires de références qui sont a priori non réconciliées. Comme dans le cas de la méthode logique, la méthode N2R est également déclarative car les connaissances qui y sont exploitées sont déclarées sous forme d'un ensemble d'axiomes et leur l'ajout ou leur suppression n'a aucun impact sur la définition de l'algorithme de réconciliation de références.

Ce chapitre est organisé de la manière suivante : en Section 7.1 nous présentons le problème de réconciliation de références considéré dans N2R. En Section 7.2 nous présentons les éléments considérés pour la comparaison des références. Ensuite en Section 7.3 nous montrons comment le calcul de similarité peut être exprimé dans un système d'équations. En Section 7.4 nous décrivons comment notre calcul de similarité de paires de références informé est représenté dans un système d'équations non linéaire. Dans cette même section nous donnons la méthode de résolution du système d'équation que nous utilisons. Puis nous illustrons le calcul itératif de similarité sur un exemple. En Section 7.5, nous présentons l'algorithme qui implémente cette méthode de calcul de similarité. Enfin, en Section 7.6 nous donnons les résultats obtenus lors de l'expérimentation de la méthode N2R ainsi que ceux obtenus lorsque N2R est précédée de l'application de la méthode logique L2R. Nous concluons ce chapitre en Section 7.7

7.1 Problème de réconciliation de références considéré dans N2R

Pour la méthode numérique N2R, nous considérons la réconciliation de références comme un calcul numérique des décisions de réconciliation. À partir des scores de similarité calculés pour chacune des paires de références et d'un seuil de réconciliation donné en paramètre nous cherchons à détecter les paires de références à réconcilier.

Dans la méthode N2R nous ne pouvons pas décider de la non réconciliation des paires de références car un score peu élevé de similarité de paires de références peut être dû à différentes raisons : (i) les paires de références réfèrent effectivement à deux entités différentes du monde réel, (ii) le manque d'informations dans les descriptions des références donne un score faible de similarité. Une telle ambiguïté, nous a conduit à ne pas décider de la non réconciliation des paires de références en se fondant sur les scores de similarité calculés.

Soient S_1 et S_2 deux sources de données conformes au même schéma RDFS+. Soient I_1 et

I_2 les deux ensembles de références correspondant respectivement aux données des deux sources S_1 et S_2. Le problème de réconciliation de références entre S_1 et S_2 consiste à déterminer parmi l'ensemble $I_1 \times I_2$ des paires de références, le sous-ensemble REC_{n2r} contenant les paires de références représentant une même entité du monde réel.

Nous déclarons Sim_r une fonction qui prend en argument une paire de références et qui renvoie une valeur réelle dans l'intervalle [0..1]. Nous notons cette fonction :

$$Sim_r : I_1 \times I_2 \longrightarrow [0..1]$$

Nous supposons disposer d'un seuil de réconciliation T_{rec} (une valeur réelle appartenant à [0..1]) donné par un expert, fixé expérimentalement ou appris sur un échantillon de données. Toute paire de références ayant un score de similarité dépassant cette valeur sera considérée comme réconciliée.

Il s'agit donc, à partir de l'ensemble $I_1 \times I_2$, de construire l'ensemble REC_{n2r} suivant :

$$REC_{n2r} = \{(i, i') |\ Sim_r(i, i') > T_{rec},\ avec\ (i, i') \in I_1 \times I_2\}$$

7.2 Comparaison des paires de références

La similarité des paires de références est obtenue à partir de la comparaison des descriptions des références qui sont représentées par l'ensemble des attributs et des relations qui portent sur ces références.

Puisque nous sommes dans un contexte où l'information peut être incomplète, la partie de leur description qu'il est naturellement pertinent à considérer est leur *description commune* : l'ensemble des attributs communs et des relations communes aux deux références. On peut noter que dans le travail de Doan et al. [DLLH03], les auteurs proposent d'exploiter en plus de la description commune des corrélations sémantiques entre des attributs[24] qui sont spécifiées par un expert du domaine. Dans le travail de Dong et al. [DHM05], des correlations syntaxiques entre attributs ont été exploitées lors de la définition des descriptions communes des paires de références, comme par exemple, considérer les attributs *nom* et *e-mail* comme deux attributs communs. Cependant, dans notre approche nous ne souhaitons pas avoir recours à un expert pour spécifier ce type de connaissances.

Dans cette section, nous montrons d'abord comment filtrer les paires de références à construire, puis nous définissons la description commune des paires de références. Ensuite, nous présentons comment les scores de similarité des valeurs de base et des ensembles de valeurs de base sont calculés.

7.2.1 Exploitation des connaissances du domaine pour le filtrage

Dans notre méthode numérique de réconciliation de références nous cherchons à déterminer l'ensemble REC_{n2r} de paires de références réconciliées à partir des ensembles $I1$ et $I2$. Pour ce faire, nous devons comparer l'ensemble de paires de références appartenant à l'espace de réconciliation $I1 \times I2$. Cependant, cet ensemble peut être de très grande taille. C'est pour cela que certaines méthodes, comme les méthodes dites de *blocking* (cf. Section 4.5) utilisent en pré-traitement des techniques de filtrage pour diminuer la taille de l'espace des réconciliations.

[24] Par exemple, une correlation sémantique entre les attributs âge et salaire d'une personne. En effet, il est par exemple invraisemblable d'avoir une personne âgée de 5 ans et ayant un salaire de 150K£.

Dans la méthode N2R, pour réduire la taille l'espace de réconciliation nous exploitons les connaissances suivantes :

- *les axiomes de disjonction entre classes.* Nous exploitons les axiomes de disjonction entre classes pour éliminer de l'espace de réconciliation, toute paire de références appartenant à deux classes disjointes.

- *hypothèse du nom unique UNA.* Nous exploitons également l'axiome de l'UNA pour éliminer de l'espace de réconciliation toute paire de références provenant d'une source de données qui vérifie l'hypothèse de l'UNA.

Pour réduire encore plus l'espace de réconciliation considéré dans N2R, nous pouvons appliquer au préalable la méthode L2R qui permet d'inférer des décisions de réconciliation et de non réconciliation entre paires de références. Dans ce cas, les paires de références pour lesquelles L2R a inféré une non réconciliation sont éliminées de l'espace de réconciliation de N2R.

7.2.2 Description commune des paires de références

La description commune d'une paire de références est représentée par l'ensemble des attributs communs et des relations communes. Nous nommons l'ensemble des attributs communs et des valeurs de base associées aux références par ces attributs, *la description locale.* Nous nommons l'ensemble des relations communes et des références associées aux références par ces relations, *la description globale.*

Exemple de données RDF du domaine des lieux culturels. Soient S1 et S2 deux sources de données RDF présentées en Figure 7.1 et conformes au schéma RDFS montré en Figure 5.4.

La source S1 :
nomMusee(S1_m1,*"LOUVRE"*) ; contient(S1_m1,S1_p1) ; localiseA(S1_m1,S1_c1) ;
nomVille(S1_c1,"Paris") ; nomOeuvre(S1_p1,*"La Joconde"*) ;
adresseMusee(S1_m1,*"99, r. RIVOLI"*) ;

La source S2 :
nomMusee(S2_m1,*"le Louvre"*) ; localiseA(S2_m1,S2_c1) ; Contient(S2_m1,S2_p1) ;
contient(S2_m1, S2_p2) ; nomVille(S2_c1, *"Ville de paris"*) ;
nomOeuvre(S2_p1, *"Abricotiers en fleurs"*) ; nomOeuvre(S2_p2, *"Joconde'*) ;
adresseMusee(S2_m1,*"rue Rivoli"*) ;

FIG. 7.1 – *N2R – Exemple de données RDF du domaine des lieux culturels.*

Définition – Attributs communs. Un attribut a est dit commun à une paire de références $< i, i' >$, s'il existe un fait-attribut de a appartenant à la description de la référence i et un fait-attribut de a appartenant à la description de i'. Nous notons $CAttr(< i, i' >)$ l'ensemble des attributs communs à une paire de références $< i, i' >$ qui est défini comme suit :

$$CAttr(< i, i' >) = \{a \mid \exists v \ \in Val, \exists \ v' \ \in Val, \ tq \ \ a(i, v) \in Desc(i) \ et \ a(i', v') \in Desc(i')\}$$

où Val représente l'ensemble des valeurs des attributs de l'ensemble de références à réconcilier.
Exemple : dans la Figure 7.1, nous avons $CAttr(< S1_m1, S2_m1 >) = \{nomMusee\}$

Définition – Relations communes. Une relation r est dite commune à une paire de références $< i, i' >$, s'il existe un fait-relation de r appartenant à la description de la référence i et un fait-relation de r appartenant à la description de i'. Nous notons $CRel(< i, i' >)$ l'ensemble des relations communes à une paire de références $< i, i' >$ qui est défini comme suit :

$$CRel(< i, i' >) = \{r \mid \exists j \ \in I, \exists j' \ \in I, \ tq \ \ [r(i,j) \in Desc(i) \ et \ r(i',j') \in Desc(i')] \ ou$$
$$[r(j,i) \in Desc(i) \ et \ r(j',i') \in Desc(i')]\}$$

où I représente l'ensemble de références à réconcilier qui correspond à $I1 \cup I2$.

Exemple : dans la Figure 7.1, nous avons $CRel(< S1_m1, S2_m1 >) = \{localiseA, contient\}$

Définition – Valeurs associées à une référence par un attribut. Nous notons $a^+(i)$ l'ensemble de valeurs associées à une référence i par l'attribut a. Il est défini comme suit :

$$a^+(i) = \{v|\ a(i,v) \ \in \ Desc(i)\}$$

Exemple : dans la Figure 7.1, nous avons $nomOeuvre^+(S1_p1) = \{\text{“}La\ Joconde''\}$

Définition – Références associées à une référence par une relation. Nous notons $r^+(i)$ l'ensemble des références associées à une référence i par la relation r. Il est défini comme suit :

$$r^+(i) = \{j|\ r(i,j) \ \in \ Desc(i)\}$$

Exemple : dans la Figure 7.1, nous avons $contient^+(S2_m1) = \{S2_p1, S2_p2\}$

Définition – Références auxquelles une référence est associée par une relation. Nous notons $r^-(i)$ l'ensemble de références associées à une référence i par la relation r. Il est défini comme suit :

$$r^-(i) = \{j|\ r(j,i) \ \in \ Desc(i)\}$$

Exemple : dans la Figure 7.1, nous avons $contient^-(S2_p1) = \{S2_m1\}$

7.2.3 Calcul du score de similarité des valeurs et des ensembles de valeurs

Les scores de similarité des paires de références sont obtenus en premier lieu en exploitant l'ensemble des scores de similarité calculés sur les éléments de leur description locale. Il s'agit des scores de similarité des valeurs et des ensembles de valeurs de base (e.g. chaînes de caractères, nombres) de leur description commune locale.

Similarité d'une paire de valeurs.

Pour calculer les scores de similarité des valeurs de base, nous utilisons des mesures de similarité syntaxique adaptées aux particularités des valeurs des attributs, telles que celles présentées en Section 4.2. Nous avons vu que dans des travaux comme ceux de Bilenko et al [BMC+03] et Cohen et al. [CRF], les mesures de similarité les plus reconnues ont été expérimentées sur différents corpus et ils ont pu retenir que leur efficacité varie en fonction des caractéristiques des valeurs, par exemple : des valeurs courtes, des valeurs longues ou des valeurs contenant des

abréviations.

Nous pouvons également exploiter les synonymies et les non synonymies inférées par L2R. Le score de similarité est alors directement fixé à 1 ou à 0.

Nous notons Sim_v la fonction qui prend en argument une paire de valeurs de base $(v1, v2)$ et qui renvoie un nombre réel dans l'intervalle [0..1] correspondant au score de similarité des valeurs $v1$ et $v2$.

Similarité de deux ensembles de valeurs.

Certains attributs associent aux références non pas une valeur mais un ensemble de valeurs, tels que des ensembles de chaînes de caractères ou deux ensembles de dates. Pour calculer le score de similarité de deux ensembles de valeurs, nous avons besoin d'une fonction permettant de prendre en compte les différentes caractéristiques des ensembles : la taille des ensembles et le score de similarité des paires de valeurs que l'on peut former entre deux ensembles de valeurs.

Nous nottons $SSim_v$ la fonction prenant en arguments deux ensembles de valeurs de base et qui renvoie une valeur réelle dans [0..1] représentant le score de similarité des deux ensembles de valeurs $S1$ et $S2$.

Nous proposons d'utiliser une mesure de similarité inspirée de la mesure SoftTFIDF 4.2 proposée par Bilenko et al [BMC+03]. Notre mesure appelée, *SoftJaccard*, est une adaptation de la mesure de *Jaccard* entre valeurs de base $Jaccard(v1, v2) = |v1 \cap v2| \div |v1 \cup v2|$. La mesure $SoftJaccard$ permet de relâcher la contrainte de l'*égalité stricte* des tokens contenus dans l'ensemble $|v1 \cap v2|$ et utilise plutôt une contrainte de *similarité* des tokens relativement à un seuil donné θ.

Une mesure de similarité entre valeurs de base Sim_v est utilisée pour calculer le score de similarité des paires de valeurs $(v1, v2) \in S1 \times S2$. Comme la mesure SoftTFIDF, l'ensemble $CLOSE_v(S1, S2, \theta)$ représente l'ensemble de valeurs ayant un score de similarité $Sim_v(v1, v2)$ supérieur à θ et il est défini comme suit :

$$CLOSE_v(S1, S2, \theta) = \{v_j \mid v_j \in S1 \; et \; \exists v_k \in S2 \; et \; tq \; Sim_v(v_j, v_k) > \theta\}$$

La mesure *SoftJaccard* que nous notons $SoftJaccard_v$ est alors définie par l'expression suivante :

$$SoftJaccard_v(S1, S2, \theta) = \frac{|CLOSE_v(S1, S2, \theta)|}{|S1|}, \; avec \; |S1| \geq |S2|$$

Exemple : $SoftJaccard_v(\{"Marie-Christine\ Rousset", "Fatiha\ Sais", "Helene\ Gagliardi"\}, \{"Nathalie\ Pernelle", "Fatiha\ Sais"\}, 0.7) = 1/3$

La fonction $SSim_v$ peut être également calculée par d'autres mesures de similarité appliquées aux ensembles de valeurs, telle que la mesure *BestMatch* utilisée dans Bisson [Bis92] et dans Euzenat et Valtchev [EV04].

7.3 Expression du calcul de similarité dans un système d'équations

Notre méthode de réconciliation de références étant globale, nous avons vu que le score de similarité d'une paire de références est également calculé en fonction des scores de similarité calculés pour d'autres références. Par exemple, la similarité d'une paire de référence de musées est calculée en fonction des valeurs de leurs attributs communs $nomMusee$ et $adresseMusee$ mais aussi en fonction des références qui leur sont reliées par les relations $localiseA$ et $contient$. La similarité d'une paire de références d'oeuvres est calculée en fonction de la valeur de leur attribut commun $nomOeuvre$ mais aussi en fonction des références qui leur sont liées par les relations $contient$ et la relation $realiseePar$. La similarité d'une paire de musées influence donc la similarité des œuvres qu'elles contiennent et inversement. Le calcul des scores de similarité des paires de références peut alors être exprimé dans un système d'équations où chaque équation représente le calcul de similarité d'une paire de références. Ainsi, le calcul revient à la résolution d'un système d'équations portant sur un ensemble de variables représentant les scores de similarité. Dans les travaux de Bisson [Bis92] et de Euzenat et Valtchev [EV04], les auteurs ont également exprimé le calcul de distance ou de dissimilarité entre objets dans un système d'équations.

Dans la suite, nous allons présenter comment le calcul des scores de similarité des paires de références s'exprime en fonction des scores de similarité des valeurs de leur attributs communs et en fonction des références qui leur sont liées par les relation communes. Tout d'abord, nous présentons comment ce calcul des scores de similarité s'exprime dans un *système d'équations*. Puis, nous montrons comment la prise en compte des ensembles de références et des connaissances du schéma nous conduit à exprimer le calcul sous forme d'un *système d'équations non-linéaire*.

Pour illustrer le calcul de similarité nous allons nous appuyer sur l'exemple que nous présentons ci-dessous.

Exemple : soient les trois paires de références représentant une paire de musées $< S1_m1, S2_m1 >$, une paire d'oeuvres $< S1_p1, S2_p1 >$ et une paire de villes $< S1_c1, S2_c1 >$.

À partir de leur description sous forme d'un ensemble de faits RDF présenté en Figure 7.1 et du schéma RDFS présenté en Figure 5.4, nous obtenons les descriptions communes de ces trois paires de références. Nous présentons dans la Table 7.1 les descriptions communes des paires de références dans deux colonnes : leur description locale représentée par les attributs et leur description globale représentée par les relations.

Les scores de similarité de ces trois paires de références peuvent être représentés comme le résultat de l'application d'un ensemble de fonctions appliquées sur les similarités des références et des valeurs qui lui sont liées.

$$Sim_r(< S1_m1, S2_m1 >) = f_1(Sim_v(\text{“}le\ Louvre''), \text{“}LOUVRE''),\\ Sim_r(< S1_p1, S2_p1 >), Sim_r(< S1_c1, S2_c1 >))$$

$$Sim_r(< S1_p1, S2_p1 >) = f_2(Sim_v(\text{“}La\ Joconde'', \text{“}Abricotiers\ en\ fleurs''),\\ Sim_r(< S1_m1, S2_m1 >))$$

$$Sim_r(< S1_c1, S2_c1 >) = f_3(Sim_v(\text{“}Paris'', \text{“}Ville\ de\ Paris''), Sim_r(< S1_m1, S2_m1 >))$$

Description locale (Attributs)	**Description globale (Relations)**
$CAttr(< S1_m1, S2_m1 >) = \{nomMusee\}$	$CRel(< S1_m1, S2_m1 >) = \{localiseA, contient\}$
$CAttr(< S1_p1, S2_p1 >) = \{nomOeuvre\}$	$CRel(< S1_p1, S2_p1 >) = \{contient\}$
$CAttr(< S1_c1, S2_c1 >) = \{nomVille\}$	$CRel(< S1_c1, S2_c1 >) = \{localiseA\}$
$nomMusee^+(S1_m1) = \{\text{“}LOUVRE\text{”}\}$	$localiseA^+(S1_m1) = \{S1_c1\}$ $contient^+(S1_m1) = \{S1_p1, S2_p2\}$
$nomMusee^+(S2_m1) = \{\text{“}le\ Louvre\text{”}\}$	$localiseA^+(S2_m1) = \{S2_c1\}$ $contient^+(S2_m1) = \{S2_p1\}$
$nomOeuvre^+(S1_p1) = \{\text{“}La\ Joconde\text{”}\}$	$contient^-(S1_p1) = \{S1_m1\}$
$nomOeuvre^+(S2_p1) = \{\text{“}Abricotiers\ en\ fleurs\text{”}\}$	$contient^-(S2_p1) = \{S2_m1\}$
$nomVille^+(S1_c1) = \{\text{“}Paris\text{”}\}$	$localiseA^-(S1_c1) = \{S1_m1\}$
$nomVille^+(S2_c1) = \{\text{“}Ville\ de\ Paris\text{”}\}$	$localiseA^-(S2_c1) = \{S2_m1\}$

TAB. 7.1 – *N2R-Exemple de description commune à des références*

A travers cet exemple nous pouvons voir que les différentes expressions du calcul des paires de références forment un système d'équations, que nous allons définir plus formellement.

Dans chaque expression, nous distinguons deux types de scores de similarité : (i) *des scores de similarité* ***constants***, représentant les scores de similarité des valeurs de base des attributs ; (ii) *des scores de similarité* ***variables***, représentant les scores de similarité des paires de références, qui se calculent en fonction des scores de similarité d'autres paires de références.

Soit n le nombre de paires de références pour lesquelles nous souhaitons calculer un score de similarité. Soit X le vecteur de variables $(x_1, x_2, \ldots, x_n)$ tel que chaque variable x_i représente le score de similarité d'une paire de références $< ref, ref' >$.

Équation pour le calcul de similarité. Soit une fonction réelle $f_i : R^n \longrightarrow R$. L'application de f_i sur un vecteur de variables $X = (x_1, x_2, \ldots, x_n)$ noté $f_i(X)$ renvoie une valeur réelle $x_i \in R$ représentant le score de similarité. Pour tout $i \in [1..n]$, une équation $x_i = f_i(X)$ est formée.

Système d'équations pour le calcul de similarité. Le système d'équations à résoudre est de la forme $F(X) = X$ où F une fonction vectorielle représentant le vecteur des n fonctions f_i. La fonction F qui opère des transformations sur le vecteur X dans R^n est définie comme suit :

$F : R^n \longrightarrow R^n$,
$(x_1, x2, \ldots\ldots x_n) \mapsto (f_1(x_1, x2, \ldots\ldots, x_n), \ldots\ldots, f_n(x_1, x2, \ldots\ldots, x_n))$
Avec,

$$F = \begin{pmatrix} f_1 \\ \vdots \\ \vdots \\ f_n \end{pmatrix}$$

Calcul de similarité représenté dans un système d'équations linéaire.

Dans le cas où le calcul est exprimé par une simple moyenne éventuellement pondérée, le calcul des scores de similarité peut être représenté dans un système d'équations linéaire.

Par exemple, pour les trois paires de références dont la description est donnée en Table 7.1, les scores de similarité peuvent alors être calculés comme suit :

$Sim_r(< S1_m1, S2_m1 >) = \lambda_{11} * Sim_v(\text{“}le\ Louvre''\text{, “}LOUVRE'')+$
$\lambda_{12} * Sim_r(< S1_p1, S2_p1 >) + \lambda_{13} * Sim_r(< S1_c1, S2_c1 >)$

$Sim_r(< S1_p1, S2_p1 >) = \lambda_{21} * Sim_v(\text{“}La\ Joconde''\text{, “}Abricotiers\ en\ fleurs'')+$
$\lambda_{22} * Sim_r(< S1_m1, S2_m1 >)$

$Sim_r(< S1_c1, S2_c1 >) = \lambda_{31} * Sim_v(\text{“}Paris''\text{, “}Ville\ de\ Paris'')+$
$\lambda_{32} * Sim_r(< S1_m1, S2_m1 >)$

La valeur des λ_{ij}, avec $(1 \leq i \leq 3$ et $1 \leq j \leq 2)$, représente un nombre réel dans [0..1] correspondant à la pondération exprimant l'importance d'un attribut ou d'une relation dans le calcul du score de similarité d'une paire de références.

$$\underset{\mathbf{C}}{\begin{pmatrix} 1 & -\lambda_{12} & -\lambda_{13} \\ -\lambda_{22} & 1 & 0 \\ -\lambda_{32} & 0 & 1 \end{pmatrix}} \times \underset{\mathbf{X}}{\begin{pmatrix} Sim_r(< S1_m1, S2_m1 >) \\ Sim_r(< S1_p1, S2_p1 >) \\ Sim_r(< S1_c1, S2_c1 >) \end{pmatrix}}$$

$$= \underset{\mathbf{B}}{\begin{pmatrix} \lambda_{11} * Sim_v(\text{“}le\ Louvre''\text{, “}LOUVRE'') \\ \lambda_{21} * Sim_v(\text{“}La\ Joconde''\text{, “}Abricotiers\ en\ fleurs'') \\ \lambda_{31} * Sim_v(\text{“}Paris''\text{, “}Ville\ de\ Paris'') \end{pmatrix}}$$

Ce calcul peut alors être représenté dans un système d'équations linéaire de la forme $C * X = B$. La matrice des variables X correspond à la matrice représentant les fonctions de calcul de similarité des paires de références $Sim_r(< ref, ref' >)$. Enfin, la matrice B représente le résultat de fonctions combinant les scores de similarité de valeurs de base $Sim_v(v1, v2)$ en considérant les pondérations des attributs.

Le système d'équations linéaire $C * X = B$ calculant les scores de similarité pour n paires de références est exprimé comme suit :

$$\underset{\mathbf{C}}{\begin{pmatrix} 1 & c_{1,2} & c_{1,3} & \dots & c_{1,n} \\ c_{2,1} & 1 & c_{2,3} & \dots & c_{2,n} \\ c_{3,1} & c_{3,2} & 1 & \dots & c_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n,2} & c_{n,2} & c_{n,3} & \dots & 1 \end{pmatrix}} \times \underset{\mathbf{X}}{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}} = \underset{\mathbf{B}}{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}}$$

Les coefficients $c_{i,j}$ de la matrice C associés aux n variables sont alors représentés par des constantes comme suit :

- le coefficient $c_{i,j} = -\lambda_{ij}$ dans le cas où il existe une relation entre la i-ème paire de références et la j-ème paire de références.
- le coefficient $c_{i,i} = 1$
- le coefficient $c_{i,j} = 0$, sinon.

Dans un tel système d'équations, pour montrer l'existence d'une solution, il suffit de montrer que la matrice des coefficients C et sa transposée C^T sont diagonale dominantes et donc positives ce qui signifie que le déterminant est strictement positif [GL89]. D'après Bisson [Bis92], Euzenat et Valtchev [EV04] il est facile de montrer qu'un système d'équations ayant une matrice de coefficients C définie comme ci-dessus admet un déterminant strictement positif.

Néanmoins, nous allons voir dans la section suivante que la fonction vectorielle F que nous avons définie ne nous permet pas d'aboutir à un système d'équations linéaire.

7.4 Calcul de similarité informé par les connaissances du schéma

Après avoir exprimé le calcul de similarité des références dans un système d'équations sous la forme $F(X) = X$, il nous reste à définir l'expression de la fonction vectorielle F qui permet de calculer les scores de similarité des paires de références.

En cohérence avec l'approche que nous avons proposée dans la méthode logique L2R, dans la méthode numérique N2R nous exploitons également les connaissances du schéma dans le calcul de similarité des paires de références.

La représentation du calcul de similarité des paires de références sous forme d'un *système d'équations linéaire* ne permet pas d'utiliser certaines fonctions qui capturent certaines connaissances du schéma ou qui calculent les scores de similarité des ensembles de références. Nous avons défini une fonction non linéaire de calcul de similarité exploitant des connaissances du schéma ce qui a aboutit à un système d'équations non linéaire. Afin de résoudre ce système d'équations non linéaire, nous avons adapté une méthode de résolution itérative.

Dans cette section, nous allons tout d'abord décrire les connaissances du schéma que nous exploitons dans la calcul de similarité. Ensuite, nous présentons la construction du système d'équations non linéaire et comment les connaissances du schéma sont intégrées dans l'expression de la fonction F. Puis, nous décrivons la méthode itérative qui permet de le résoudre notre système d'équations non linéaire. Enfin, nous illustrons le calcul itératif par un exemple.

7.4.1 Exploitation des axiomes de fonctionnalité du schéma

Afin de déterminer l'importance des différents attributs et des différentes relations à partir des connaissances du domaine, nous exploitons les axiomes de fonctionnalité et de fonctionnalité inverse déclarés dans le schéma RDFS+. Ainsi, nous dégageons deux cas de figures qui vont être intégrées différemment dans le calcul de similarité. Lorsque deux références dont on calcule la similarité sont en *dépendance fonctionnelle* d'un ensemble d'attributs ou de relations, la similarité des valeurs et des références qui leur sont liées par ces attributs et ces relations références doit avoir un impact élevé sur la similarité de la paire. Lorsque deux références dont on calcule la similarité ne sont pas en *dépendance fonctionnelle* d'un ensemble d'attributs ou de relations,

la similarité des valeurs et des références qui leur sont liées par ces attributs et ces relations références doit avoir un impact plus faible sur la similarité de la paire. Ces deux types de dépendance sont utilisés dans notre calcul de similarité de façon à ce que les résultats du calcul dans N2R soient cohérents avec ceux de la méthode logique L2R. Dans la méthode L2R, grâce aux règles logiques, la réconciliation d'une paire de références est propagée à d'autres paires de références qui sont en dépendance fonctionnelle avec cette paire. Par exemple, si on infère la réconciliation de deux musées alors on infère également la réconciliation des deux villes correspondantes d'après l'axiome de fonctionnalité déclaré pour la relation *localiseA*. En revanche, l'inverse n'est pas vrai, car ce n'est parce que nous inférons la réconciliation de deux villes que l'on infère aussi la réconciliation des deux musées car une ville peut contenir plusieurs musées. Ce principe de propagation va être calqué dans N2R de manière numérique : d'une part, par l'utilisation d'une fonction d'agrégation *maximum* des scores de similarité des paires de références et des valeurs impliquées dans une dépendance fonctionnelle et d'autre part, par l'utilisation d'une *moyenne pondérée* des scores de similarité des autres paires ou ensembles de références ou de valeurs non impliquées dans une dépendance fonctionnelle.

Dans les deux sous-sections suivantes nous allons définir les ensembles de relations ou d'attributs avec lesquels une paire de références est soit en dépendance fonctionnelle (DF) soit n'est pas en dépendance fonctionnelle (NDF). Pour chaque définition nous donnons un exemple qui appuiera sur les données montrées en Figure 7.1 et sur les axiomes du schéma présentés dans la Section 6.3.3 du chapitre 6.

Dépendance fonctionnelle des références (DF).

Une référence i est dite en *dépendance fonctionnelle* d'un attribut a si et seulement si a est un attribut fonctionnel inverse et s'il existe une paire $< i, v >$ instance de a. Nous notons $DF_A(< i, i' >)$ l'ensemble des attributs communs à la paire de références $< i, i' >$ et tels que les références i et i' sont en dépendance fonctionnelle de ces attributs. L'ensemble $DF_A(< i, i' >)$ est défini comme suit :

$$DF_A(< i, i' >) = \{a \mid a \in CAttr(< i, i' >) \; et \; PFI(a)\}$$

Exemple : $DF_A(< S1_p1, S2_p1 >) = \{nomOeuvre\}$, car nous avons déclaré $PFI(nomOeuvre)$ comme axiome du schéma RDFS et *nomOeuvre* apparaît dans la description commune des deux références.

Une référence i est dite en *Dépendance Fonctionnelle* d'une relation r si (i) r est fonctionnelle inverse et la paire $< i, j >$ est une instance de r ou si (ii) r est fonctionnelle et la paire $< j, i >$ est une instance de r. Nous notons $DF_R(< i, i' >)$ l'ensemble de relations communes à la paire de références $< i, i' >$ et tels que les références i et i' sont en dépendance fonctionnelle de ces relations. L'ensemble $DF_R(< i, i' >)$ est défini comme suit :

$$DF_R(< i, i' >) = \{r \mid r \in CRel(< i, i' >) \; et \; ([PFI(r) \; et \; (r(i, j) \; \in \; Desc(i) \; et \; r(i', j') \; \in \; Desc(i'))]$$

$$ou \; [PF(r) \; et \; (r(j, i) \; \in \; Desc(i) \; et \; r(j', i') \; \in \; Desc(i'))])\}$$

Exemple : $DF_R(< S1_m1, S2_m1 >) = \{contient\}$, car nous avons déclaré $PFI(contient)$ comme axiome du schéma RDFS et *contient* apparaît dans la description commune des deux références. De même, $DF_R(< S1_c1, S2_c1 >) = \{localiseA\}$, car nous avons déclaré $PF(localiseA)$

comme axiome du schéma RDFS et $localiseA$ apparaît dans la description commune des deux références.

Dépendance fonctionnelle multiple composée de plusieurs attributs et relations. Les ensembles $DF_A(< i, i' >)$ peuvent être définis de manière plus générale en considérant les axiomes de fonctionnalité et de fonctionnalité inverse généralisés aux ensembles d'attributs. On note $DF_A^M(< i, i' >)$ cet ensemble qui est défini comme suit :

$$DF_A^M(< i, i' >) = \{\{a1, a2, \ldots, an\} \mid \forall h \in [1..n], a_h \in CAttr(< i, i' >)\ , PFI(a1, \ldots, an)\}$$

Exemple : la combinaison d'attributs dans laquelle les références $S1_m1$ et $S2_m1$, décrites dans la Figure 7.1, sont en dépendance fonctionnelle est :
$DF_A^M(< S1_m1, S2_m1 >) = \{nomMusee, adresseMusee\}$, car nous avons déclaré $PFI(nomMusee$, $adresseMusee)$ comme axiome du schéma RDFS+.

Les ensembles $DF_R(< i, i' >)$ peuvent être également définis de manière plus générale en considérant les axiomes de fonctionnalité et de fonctionnalité inverse généralisés aux ensembles de relations. On note $DF_R^M(< i, i' >)$ cet ensemble qui est défini comme suit :

$$DF_R^M(< i, i' >) = \{\{r1, r2, \ldots, rn\} \mid \forall h \in [1..n], r_h \in CRel(< i, i' >)\ , et$$
$$([PFI(r1, r2, \ldots, rn)\ et\ (\forall h \in [1..n]\ r_h(i, j)\ \in\ Desc(i)\ et\ r_h(i', j')\ \in\ Desc(i'))]ou$$
$$[PF(r1, r2, \ldots, rn)\ et\ (\forall h \in [1..n],\ r_h(j, i)\ \in\ Desc(i)\ et\ r_h(j', i')\ \in\ Desc(i'))])\}$$

Non dépendance fonctionnelle des références (NDF).

De manière analogue nous considérons l'ensemble des attributs avec lesquels une référence i n'est pas en dépendance fonctionnelle. Parmi ces attributs nous distinguons ceux ayant une seule valeur associée à la référence i de ceux ayant plusieurs valeurs car dans le calcul de similarité des paires de références ces ensembles sont considérés différemment. Nous notons $NDF_A(< i, i' >)$ l'ensemble d'attributs communs aux références i et i', ayant une seule valeur et pour lesquels les références i et i' ne sont pas en dépendance fonctionnelle. Cet ensemble est défini comme suit :

$$NDF_A(< i, i' >) = \{a \mid a \in (CAttr(< i, i' >)\ \setminus\ (DF_A(< i, i' >) \bigcup_m DF_A^M(< i, i' >)))\ et$$
$$(|a^+(i)| = 1\ et\ |a^+(i')| = 1)\}$$

Avec, $\bigcup_m$ un opérateur qui applique une union entre un ensemble d'éléments et un ensemble d'ensembles d'éléments et qui renvoie un ensemble de tous les éléments. Par exemple :

$$\{e1, e2, e3\} \bigcup_m \{\{e1, e2, e4\}, \{e1, e5, e6\}\} = \{e1, e2, e3, e4, e5, e6\}$$

Nous notons $NDF_A^*(< i, i' >)$ l'ensemble des attributs communs ayant plusieurs valeurs et tels que les références i et i' ne sont pas en dépendance fonctionnelle de ces valeurs. Cet ensemble est défini comme suit :

$$NDF^{*}_{A}(< i, i' >) = \{a \mid a \in (CAttr(< i, i' >) \ \setminus \ (DF_A(< i, i' >) \bigcup_{m} DF^{M}_{A}(< i, i' >))) \ ou$$

$$(|a^{+}(i)| > 1 \ et \ |a^{+}(i')| > 1)\}$$

Nous considérons également l'ensemble de relations telles qu'une référence i n'est pas en dépendance fonctionnelle de ces relations. Parmi ces relations nous distinguons celles ayant une seule référence associée à la référence i de celles ayant plusieurs références associées. Nous notons $NDF_R(< i, i' >)$ l'ensemble des relations communes aux références i et i', ayant une seule référence et pour lesquelles les références i et i' ne sont pas en dépendance fonctionnelle. Cet ensemble est défini comme suit :

$$NDF_R(< i, i' >) = \{r \mid r \in (CRel(< i, i' >) \ \setminus \ (DF_R(< i, i' >) \bigcup_{m} DF^{M}_{R}(< i, i' >))) \ et$$

$$((|r^{+}(i)| = 1 \ et \ |r^{+}(i')| = 1) \ ou \ (|r^{-}(i)| = 1 \ et \ |r^{-}(i')| = 1))\}$$

Exemple : l'ensemble de relations avec lesquelles les références $S1_m1$ et $S2_m1$, décrites dans la Figure 7.1, ne sont pas en dépendance fonctionnelle est :
$NDF_R(< S1_m1, S2_m1 >) = \{localiseA\}$, car nous n'avons pas déclaré $PFI(localiseA)$ comme axiome du schéma RDFS.

Nous notons $NDF_{R*}(< i, i' >)$ l'ensemble de relations communes aux références i et i', ayant plusieurs références et avec lesquelles les références i et i' ne sont pas en dépendance fonctionnelle. Cet ensemble est défini comme suit :

$$NDF^{*}_{R}(< i, i' >) = \{r \mid r \in (CRel(< i, i' >) \ \setminus \ (DF_R(< i, i' >) \bigcup_{m} DF^{M}_{R}(< i, i' >))) \ et$$

$$((|r^{+}(i)| > 1 \ ou \ |r^{+}(i')| > 1) \ ou \ (|r^{-}(i)| > 1 \ ou \ |r^{-}(i')| > 1))\}$$

7.4.2 Construction du système d'équations non linéaire

Dans cette section nous montrons comment nous construisons le système d'équations non linéaire pour le calcul de similarité des paires de références. Pour que le calcul puisse exploiter différemment les différents types de dépendances fonctionnelles et les différentes similarités (ie. paires ou ensembles de valeurs et de références), nous avons dû définir différentes constantes et différentes variables.

Pour différencier les relations et donc les paires de références intervenant dans l'expression du calcul de similarité d'une paire de références, nous utilisons trois indices :

- i : indique le rang de la paire de références pour laquelle nous souhaitons calculer la similarité, avec $1 \leq i \leq n$, où n représente le nombre de paires de références considérées.

- j : indique le rang de la relation commune de la i-ème paire de références, avec $1 \leq j \leq n$.

⋆ l : indique le rang de l'attribut commun de la i-ème paire de références, avec $1 \leq l \leq m$, où m est le nombre d'attributs communs à la i-ème paire de références.

Considérons $< ref, ref' >$ une paire de références pour laquelle nous souhaitons calculer un score de similarité. Afin de différencier les types de dépendance fonctionnelle de la paire de références par rapport aux paires ou ensembles de valeurs et aux paires ou ensembles de références intervenant dans le calcul du score de similarité, nous introduisons trois indices supplémentaires que l'on associe à chaque constante et à chaque variable du système d'équations :

⋆ df : indique que la constante représente une paire de valeurs ou que la variable représente une paire de références d'un attribut ou d'une relation avec lesquels la paire de références $< ref, ref' >$ est en dépendance fonctionnelle.

⋆ dfm : indique que la constante représente une paire de valeurs ou que la variable représente une paire de références d'un attribut ou d'une relation avec lesquels la paire de références $< ref, ref' >$ est en dépendance fonctionnelle multiple.

⋆ ndf : indique que la constante représente une paire ou deux ensembles de valeurs d'un attribut ou que la variable représente une paire ou deux ensembles de références d'une relation avec lesquels la paire de références $< ref, ref' >$ n'est pas en dépendance fonctionnelle.

Constantes du système d'équations non linéaire :

Nous allons définir les constantes intervenant dans l'expression de la fonction de similarité F du système d'équations.

Constantes pour représenter le score de similarité des paires de valeurs de base. À toute paire de valeurs (v, v') du l-ème attribut de la i-ème paire de références, une constante b_{il} est associée. Cette constante représente le score de similarité de la paire de valeurs calculé par la fonction $Sim_v(v, v')$. Ces constantes prennent des valeurs réelles dans l'intervalle [0..1]. Pour la i-ème paire de références $< ref, ref' >$ trois constantes b_{il-df}, b_{il-dfm} et b_{il-ndf} représentant les scores de similarité des paires de valeurs, sont définies.

1. La constante b_{il-df} est définie comme suit :

$$b_{il-df} = Sim_v(v, v'),\ tq\ \exists\ a_l \in DF_A(< ref, ref' >)\ et\ (v, v') \in (a_l^+(ref) \times a_l^+(ref'))$$

2. La constante b_{il-dfm} est définie comme suit :

$$b_{il-dfm} = Sim_v(v, v'),\ tq\ \exists\ E \in DF_A^M(< ref, ref' >),\ a_l \in E\ et\ (v, v') \in (a_l^+(ref) \times a_l^+(ref'))$$

3. La constante b_{il-ndf} est définie comme suit :

$$b_{il-ndf} = Sim_v(v, v'),\ tq\ \exists\ a_l \in NDF_A(< ref, ref' >)\ et$$
$$(v, v') \in (a_l^+(ref) \times a_l^+(ref'))$$

Constante pour représenter le score de similarité des ensembles de valeurs de base. Pour exprimer le score de similarité de deux ensembles de valeurs de base calculé par la fonction $SSim_v$(S1, S2), à toute paire d'ensembles de valeurs de base (S1, S2) est associée une constante BS_{il}. Cette constante représente le score de similarité des deux ensembles de valeurs du l-ème attribut commun à la i-ème paire de références. Le seul type de dépendance associé aux ensembles de valeurs est la non dépendance fonctionnelle. Pour le calcul de similarité d'une paire de référence $< ref, ref' >$, nous définissons alors une constante BS_{il-ndf} qui représente le score de similarité des ensembles de valeurs :

$$BS_{il-ndf} = SSim_v(S1, S2),\ tq\ \exists\ a_l \in NDF_A^*(< ref, ref' >)$$
$$et\ S1 = a_l^+(ref)\ et\ S2 = a_l^+(ref')$$

Variables du système d'équations non linéaire :

Nous donnons la définition des variables sur lesquelles le système d'équations est défini.

Variables représentant le score de similarité des paires de références. À toute paire de références $< ref, ref' >$ une variable x_i est associée. Cette variable représente le score de similarité de la paire de références qui est calculé par la fonction $Sim_r(< ref, ref' >)$ que nous exprimons dans le système d'équations par une fonction $f_i(X) = x_i$ et que nous allons définir dans la suite.

Lorsqu'une paire de références $< ref_h, ref'_h >$, représentée par son score de similarité x_j, intervient dans le calcul du score de similarité d'une i-ème paire de références $< ref, ref' >$, un indice contextuel i (avec $1 \leq i \leq n$ et $i \neq j$) est ajouté à la variable x_j. Cet indice i indique le rang de la paire de références dont le score de similarité est fonction de x_j. Dans ce cas, la variable x_j est notée x_{ij}. Pour la i-ème paire de références $< ref, ref' >$ trois variables représentant les scores de similarité des paires de références x_{ij-df}, x_{ij-dfm} et x_{ij-ndf} sont définies.

1. La variable x_{ij-df} est définie comme suit :

$$x_{ij-df} = Sim_r(< ref_h, ref'_h >),\ tq\ \exists\ r_j \in DF_R(< ref, ref' >)$$
$$et < ref_h, ref'_h > \in (r_j^+(ref) \times r_j^+(ref')),\ ou < ref_h, ref'_h > \in (r_j^-(ref) \times r_j^-(ref'))$$

2. La variable x_{ij-dfm} est définie comme suit :

$$x_{ij-dfm} = Sim_r(< ref_h, ref'_h >),\ tq\ \exists\ E \in DF_R^M(< ref, ref' >),\ r_j \in E$$
$$et < ref_h, ref'_h > \in (r_j^+(ref) \times r_j^+(ref')),\ ou < ref_h, ref'_h > \in (r_j^-(ref) \times r_j^-(ref'))$$

3. La variable x_{ij-ndf} est définie comme suit :

$$x_{ij-ndf} = Sim_r(< ref_h, ref'_h >),\ tq\ \exists\ r_j \in NDF_R(< ref, ref' >)$$
$$et < ref_h, ref'_h > \in (r_j^+(ref) \times r_j^+(ref')),\ ou < ref_h, ref'_h > \in (r_j^-(ref) \times r_j^-(ref'))$$

Variable représentant le score de similarité des ensembles de références. Pour exprimer le score de similarité de deux ensembles de références calculé par la fonction $SSim_r$(S1, S2), à toute paire d'ensembles de références (S1, S2) est associée une variable XS_{ij}. Cette variable représente le score de similarité des deux ensembles de références de la j-ème relation commune à la i-ème paire de références. Le seul type de dépendance associé aux ensembles de références est la non dépendance fonctionnelle. Pour le calcul de similarité d'une paire de références $< ref, ref' >$, nous définissons alors une variables XS_{il-ndf} qui représente le score de similarité des ensembles de références :

$$XS_{ij-ndf} = SSim_r(S1, S2),\ tq\ \exists\, r_j \in NDF_R^*(< ref, ref' >)$$
$$et(S1 = r_j^+(ref)\ et\ S2 = r_j^+(ref')),\ ou\ (S1 = r_j^-(ref)\ et\ S2 = r_j^-(ref'))$$

Expression de la fonction vectorielle $F(X)$ de calcul de similarité.

Chaque composante de la fonction vectorielle de similarité $F(X)$ est une fonction $f_i(X)$ permettant de calculer le score de similarité d'une paire de références représentée dans le système d'équations par une variable x_i. Soit x_i une variable représentant le score de similarité d'une paire de références $< ref, ref' >$. Nous exprimons la fonction $f_i(X)$ comme suit :

$$f_i(X) = \max(f_{i-df}(X), f_{i-ndf}(X))$$

avec $f_{i-df}(X)$ une fonction permettant d'agréger les scores de similarité des paires de valeurs des attributs dont la i-ème paire de références est en dépendance fonctionnelle et des paires de références liées par des relations dont la i-ème paire de références est en dépendance fonctionnelle, et $f_{i-ndf}(X)$ une fonction permettant d'agréger les scores de similarité des paires ou ensembles de valeurs d'attributs et des paires ou ensembles de références liées par de relations dans lesquels la i-ème paire de références n'est pas en dépendance fonctionnelle.

Prise en compte des attributs et des relations fonctionnels Dans le calcul de similarité d'une paire de références $< ref, ref' >$, les scores de similarité des paires de valeurs des attributs et les scores de similarité des paires de références liées par des relations, avec ref et ref' sont en dépendance fonctionnelle, sont agrégés par une fonction *maximum*. En cohérence avec la méthode L2R où les décisions de réconciliation sont propagées à d'autres paires de références, dans le calcul de similarité, nous propageons à d'autres paires de références, par la fonction maximum, les scores de similarité des paires de valeurs et de références ayant un impact fort sur leur similarité. Ceci est exprimé dans la formule $f_{i-df}(X)$ ci-dessous.

$$f_{i-df}(X) = \max(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|}(b_{il-df}), moy(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm})),$$
$$\bigcup_{j=0}^{j=|DF_R(<ref,ref'>)|}(x_{ij-df}), moy(\bigcup_{j=0}^{j=|DF_R^M(<ref,ref'>)|}(x_{ij-dfm})))$$

On note que les scores de similarité des paires de valeurs de références dans lesquelles la i-ème paire de références est en dépendance fonctionnelle multiple sont tout d'abord agrégés par une *moyenne* notée *moy* avant d'être pris en compte dans la fonction maximum.

Prise en compte des attributs et des relations non fonctionnels Dans le calcul de similarité d'une paire de références $< ref, ref' >$, les scores de similarité des paires ou d'ensembles de valeurs des attributs et les scores de similarité des paires ou d'ensembles de références liées par des relations dont ref et ref' ne sont pas en dépendance fonctionnelle sont agrégés dans une moyenne pondérée que nous présentons ci-dessous.

Calcul des pondérations des attributs et des relations : la seule connaissance concernant l'impact de ces attributs et relations sur le score de similarité est qu'ils ont un impact *faible*. Nous définissons donc une pondération identique pour tous les attributs et pour toutes les relations impliqués dans le calcul de la moyenne pondérée. Nous définissons donc la constante λ_{ij}, qui est un nombre réel positif dans [0..1] représentant la pondération associée à chaque relation et à chaque attribut de rang j intervenant dans le calcul du score de similarité de la i-ème paire de références :

$$\lambda_{ij} = \frac{1}{|CAttr(< ref, ref' >)| + |CRel(< ref, ref' >)|}$$

Dans le cas des approches supervisées, les pondérations λ_{ij} peuvent être spécifiées par un expert ou apprises sur un échantillon de données étiquetées. Cependant, nous voulons que notre approche soit automatique et non supervisée. Cette façon de pondérer les attributs et relations non fonctionnels fait que même si toutes les valeurs des attributs et les références des relations non fonctionnels ont un score de similarité égal à 1, leur moyenne pondérée ne sera pas égale à 1 dans le cas où d'autres attributs ou d'autres relations sont fonctionnels. Nous avons choisi une pondération calculée en fonction des relations et des attributs communs renseignées. Nous pouvons modifier ce calcul dans le cas où nous traitons des données contenant peu d'attributs et de relations renseignées de façon à calculer la pondérations sur l'ensemble des relations et des attributs déclarés dans le schéma.

Afin de garantir des scores de similarité normalisés (ie. valeurs dans [0..1]), les pondérations λ_{ij} et λ_{il} utilisées dans le calcul de chaque paire de références $< ref, ref' >$ doivent satisfaire la condition suivante :

$$\left(\sum_{l=0}^{l=|NDF_A(<ref,ref'>|} \lambda_{il} + \sum_{l=0}^{l=|NDF_A^*(<ref,ref'>|} \lambda_{il} + \sum_{j=0}^{j=|NDF_R(<ref,ref'>|} \lambda_{ij} + \sum_{j=0}^{j=|NDF_R^*(<ref,ref'>|} \lambda_{ij} \right) \leq 1$$

Cette condition est vérifiée puisque l'on a l'inclusion suivante :

$$\Big(NDF_A(< ref, ref' >) \cup NDF_A^*(< ref, ref' >) \cup NDF_R(< ref, ref' >) \cup NDF_R^*(< ref, ref' >) \Big) \subseteq \Big(CAttr(< ref, ref' >) \cup CRel(< ref, ref' >) \Big)$$

Expression de la moyenne pondérée :

$$f_{i-ndf}(X) = \sum_{l=0}^{l=|NDF_A(<ref,ref'>)|} (\lambda_{il} * \ b_{il-ndf}) + \sum_{l=0}^{l=|NDF^*_A(<ref,ref'>)|} (\lambda_{il} * \ BS_{il-ndf}) +$$

$$\sum_{j=0}^{j=|NDF_R(<ref,ref'>)|} (\lambda_{ij} * \ x_{ij-ndf}) + \sum_{j=0}^{j=|NDF^*_R(<ref,ref'>)|} (\lambda_{ij} * \ XS_{ij-ndf})$$

Pour toute paire de références représentée par une variable x_i dans le système d'équations, avec $1 \leq i \leq n$, la fonction $f_i(X)$ permettant de calculer le score de la i-ème paire de références est donc exprimée comme suit :

$$f_i(X) = \max\Bigg(\max\Bigg(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|} (b_{il-df}), moy(\bigcup_{l=0}^{l=|DF^M_A(<ref,ref'>)|} (b_{il-dfm})),$$

$$\bigcup_{j=0}^{j=|DF_R(<ref,ref'>)|} (x_{ij-df}), moy(\bigcup_{j=0}^{j=|DF^M_R(<ref,ref'>)|} (x_{ij-dfm}))\Bigg),$$

$$(\sum_{l=0}^{l=|NDF_A(<ref,ref'>)|} (\lambda_{il} * \ b_{il-ndf}) + \sum_{l=0}^{l=|NDF^*_A(<ref,ref'>)|} (\lambda_{il} * \ BS_{il-ndf}) +$$

$$\sum_{j=0}^{j=|NDF_R(<ref,ref'>)|} (\lambda_{ij} * \ x_{ij-ndf}) + \sum_{j=0}^{j=|NDF^*_R(<ref,ref'>)|} (\lambda_{ij} * \ XS_{ij-ndf}))\Bigg)$$

7.4.3 Résolution du système d'équations non linéaire

La résolution du système d'équations à n variables consiste à à affecter des valeurs au vecteur $X = (x_1, x_2, \ldots, x_n)$ de façon à ce que toutes les équations soient vraies simultanément.

Pour résoudre notre système d'équations non linéaire nous nous sommes appuyés sur une méthode proposée pour la résolution de systèmes d'équations linéaires.

De nombreuses méthodes ont été proposées pour la résolution de tels systèmes d'équations. Il existe deux types d'approche pour la résolution d'un système d'équations. Tout d'abord, les méthodes directes qui fournissent des solutions exactes à tout type de système d'équations linéaire. La plupart de ces méthodes ont une très grande complexité de calcul. Par exemple, la méthode d'*Expansion de Laplace* qui est fondée sur le principe d'inversion de matrice est de complexité $n!$. Dans cette méthode la solution du système d'équations de la forme $(C * X = B)$ est considérée comme $(X = C^{-1} * B)$, où C^{-1} correspond à l'inverse de la matrice des coefficients C. Pour une comparaison des différentes méthodes de résolution de systèmes d'équations linéaires voir [Jah07].

Pour remédier à cette trop grande complexité, un autre type d'approches a été proposé. Il s'agit des approches itératives de résolution de systèmes d'équations telle que la méthode de

Jacobi [GL89]. Ce sont des méthodes fournissant des solutions approchées avec une précision ϵ et qui ne sont applicables que sur des systèmes d'équations à diagonale dominante. Le principe de ce type de méthodes est, tout d'abord, de considérer des valeurs initiales pour les variables du vecteur X. Ensuite, il faut itérer le calcul des valeurs des variables en utilisant les valeurs obtenues à l'itération précédente. Enfin, ce calcul termine lorsque la différence des valeurs des variables obtenues à une itération $k+1$ et celles obtenues à une itération k est inférieure à ϵ. La complexité de la méthode de *Jacobi* est de (n^2) à chaque itération.

Cette méthode de résolution de systèmes d'équations linéaires peut être étendue pour résoudre des systèmes d'équations non linéaires. La même démarche a été suivie dans Bisson [Bis92] et dans Euzenat et Valtchev [EV04].

Cependant, la convergence de cette méthode n'est pas garantie dans tous les cas. Nous avons donc dû prouver formellement la convergence de la méthode itérative que nous utilisons pour la résolution de notre système d'équations. Nous donnons ci-dessous l'idée de la preuve et la preuve complète est donnée en Annexe A.

Nous donnons, en Section 7.5, l'algorithme qui implémente la méthode de *Jacobi* appliquée à notre problème de réconciliation de références.

Calcul itératif des scores de similarité inspirée de la méthode de Jacobi

Grâce à la méthode itérative de résolution de ce système non linéaire inspirée de la méthode de Jacobi, nous calculons itérativement les valeurs des variables x_i par application des fonctions $f_i(X)$ sur le vecteur X formé des valeurs des variables x_j obtenues à l'itération précédente. Ainsi, à toute itération k nous calculons tous les x_i^k pour $1 \leq i \leq n$ où n représente le nombre de paires de références :

$$x_i^k = f_i(x_1^{k-1}, x_2^{k-1}, \ldots\ldots, x_n^{k-1})$$

jusqu'à ce qu'un point fixe à précision ϵ soit atteint. Ce dernier est atteint dans le cas où :

$$\forall\ i,\ |x_i^k\ -\ x_i^{k-1}|\ \leq\ \epsilon$$

La valeur de ϵ est fixée à un nombre réel positif très petit.

Dans le calcul itératif, la valeur du vecteur X à une itération k est donc exprimée par : $X^k = F(X^{k-1})$.

À partir d'un vecteur initial $X^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ nul, on génère une suite de vecteurs réels noté U_n^k, où n représente la taille des vecteurs et k représente la taille de la suite et donc le nombre d'itérations :

$$U_n^k = \begin{cases} X^1 = F(X^0) \\ X^2 = F(X^1) \\ \vdots \\ \vdots \\ X^k = F(X^{k-1}) \end{cases}$$

Pour calculer l'écart des valeurs d'une variable x_i calculées à des itérations *k-1* et *k*, on définit un vecteur erreur e_i^k exprimé par la formule suivante :

$$e_i^k = |x_i^k - x_i^{k-1}|$$

Soit E_n^k la suite générée par le calcul des différents vecteurs erreur e^k :

$$E_n^k = \begin{cases} e^1 = (|x_1^1 - x_1^0|, |x_2^1 - x_2^0|, \ldots\ldots, |x_n^1 - x_n^0|) \\ e^2 = (|x_1^2 - x_1^1|, |x_2^2 - x_2^1|, \ldots\ldots, |x_n^2 - x_n^1|) \\ \vdots \\ \vdots \\ e^k = (|x_1^k - x_1^{k-1}|, |x_2^k - x_2^{k-1}|, \ldots\ldots, |x_n^k - x_n^{k-1}|) \end{cases}$$

La fonction F admet un point fixe si la suite E_n^k converge quand k tend vers ∞. De manière formelle :

$$\exists K,\ tq\ \forall k \geq K,\ \forall i\ e_i^k \leq \epsilon$$

Idée de la preuve de convergence du calcul itératif des scores de similarité.

Pour prouver la convergence du calcul de similarité, nous pouvons prouver que le calcul vérifie les différentes propriétés suivantes :

1. $\forall i$, $\forall k$ la valeur de x_i^k calculée par le système d'équations est bornée supérieurement par 1 : $\forall\ i, \forall k,\ x_i^k\ \leq\ 1$
2. $\forall$ i, la fonction f_i est croissante :

$$\forall\ (x_1, x_2, \ldots, x_n)\ \in\ R^n\ et\ \forall\ (y_1, y_2, \ldots, y_n)\ \in\ R^n,$$

$$(x_1 \leq y_1), (x_2 \leq y_2), \ldots, (x_n \leq y_n)\ \Longrightarrow\ f_i(x_1, x_2, \ldots, x_n)\ \leq\ f_i(y_1, y_2, \ldots, y_n)$$

3. après K itérations, le calcul admet un point-fixe. Si l'on montre que pour tout i, (x_i^k) converge vers une limite $l_i \leq 1$, on a par définition :

$$\forall i, \exists K_i, tq\ \ \forall k,\ k \geq K_i,\ |x_i^{k+1} - x_i^k| \leq \epsilon$$

Si l'on choisit $K = \max\{K_1, \ldots, K_n\}$, on a alors :

$$\forall k,\ tq\ k \geq K,\ \forall i,\ |x_i^{k+1} - x_i^k| \leq \epsilon$$

La preuve complète de la convergence est donnée en Annexe A.

7.4.4 Illustration du calcul itératif sur un exemple

Pour illustrer ce calcul, nous allons nous appuyer sur les données de la Figure 7.1 conformes au schéma RDFS+ donnée en Figure 5.4. Nous allons exploiter les axiomes du schéma RDFS+ donnés en Section 6.3.3 du chapitre 6. Nous supposons également que l'hypothèse de l'UNA est vérifiée dans les deux sources S1 et S2.

Nous rappelons ci-après l'ensemble des axiomes déclarés dans le schéma RDFS+ de la Figure 5.4 donné en chapitre 5 :

Description locale (Attributs)	Description globale (Relations)
$CAttr(< S1_m1, S2_m1 >) =$ $\{nomMusee, adresseMusee\}$	$CRel(< S1_m1, S2_m1 >) =$ $\{localiseA, contient\}$
$CAttr(< S1_p1, S2_p1 >) = \{nomOeuvre\}$	$CRel(< S1_p1, S2_p1 >) = \{contient\}$
$CAttr(< S1_p1, S2_p2 >) = \{nomOeuvre\}$	$CRel(< S1_p1, S2_p2 >) = \{contient\}$
$CAttr(< S1_c1, S2_c1 >) = \{nomVille\}$	$CRel(< S1_c1, S2_c1 >) = \{localiseA\}$
$nomMusee^+(S1_m1) = \{\text{“}LOUVRE''\}$ $adresseMusee^+(S1_m1) = \{\text{“}99,\ r.\ ROVOLI''\}$	$localiseA^+(S1_m1) = \{S1_c1\}$ $contient^+(S1_m1) = \{S1_p1\}$
$nomMusee^+(S2_m1) = \{\text{“}le\ Louvre''\}$ $adresseMusee^+(S2_m1) = \{\text{“}rue\ Rivoli''\}$	$localiseA^+(S2_m1) = \{S2_c1\}$ $contient^+(S2_m1) = \{S2_p1, S2_p2\}$
$nomOeuvre^+(S1_p1) = \{\text{“}La\ Joconde''\}$ $nomOeuvre^+(S2_p1) = \{\text{“}Abricotiers\ en\ fleurs''\}$ $nomOeuvre^+(S2_p2) = \{\text{“}Joconde'\}$ $nomVille^+(S1_c1) = \{\text{“}Paris''\}$ $nomVille^+(S2_c1) = \{\text{“}Ville\ de\ Paris''\}$	$contient^-(S1_p1) = \{S1_m1\}$ $contient^-(S2_p1) = \{S2_m1\}$ $contient^-(S2_p1) = \{S2_m1\}$ $localiseA^-(S1_c1) = \{S1_m1\}$ $localiseA^-(S2_c1) = \{S2_m1\}$

TAB. 7.2 – *Illustration du calcul – Exemple de description commune à des références*

⋆ Un ensemble d'axiomes de disjonction entre classes :
{DISJOINT(*MuseeMoyenAge*,*MuseeContemporain*), DISJOINT(*Oeuvre*, *Artiste*), DISJOINT(*LieuCulturel*, *Ville*), DISJOINT(*LieuCulturel*, *Oeuvre*)}.
Ces axiomes expriment, par exemple, qu'un *Musee* ne peut pas être à la fois un musée du Moyen Âge et un musée contemporain.

⋆ Un ensemble d'axiomes de fonctionnalité sur les propriétés :
{PF(*localiseA*), PF(*realiseePar*), PF(*nomArtiste*), PF(*anneeNaissance*), PF(*nomOeuvre*), PF(*nomVille*), PF(*nomMusee*), PF(*adresseMusee*)}.
Ces axiomes expriment le fait que, par exemple, une œuvre ne peut avoir plusieurs noms différents.

⋆ Un ensemble d'axiomes de fonctionnalité inverse sur les (ensembles de) propriétés :
PFI(*nomMusee, adresseMusee*), {PFI(*nomOeuvre*), PFI(*contient*), PFI(*nomArtiste*), PFI(*nomVille*)}.
Ces axiomes expriment le fait que, par exemple, une oeuvre ne peut pas être contenue dans plusieurs lieux culturels et qu'une adresse et un nom de musée ne peuvent pas être assignés à plusieurs musées (i.e. les deux sont nécessaires pour identifier un musée).

Nous considérons que l'UNA est vérifiée dans les sources de données S1 et S2 de la Figure 7.1.

Pour cet exemple, le système d'équations non linéaire contient quatre variables représentant les scores de similarité des quatre paires de références considérées. L'ensemble de ces paires est le résultat d'une étape de filtrage effectuée en pré-traitement où toute paire de références appartenant à des classes disjointes, est éliminée de l'espace de réconciliation. Toutes les paires de références provenant de la même source sont également éliminées de l'espace de réconciliation grâce à l'axiome de l'UNA. La description locale et globale de ces références est donnée en Table 7.2.

Le système d'équations construit pour le calcul des scores de similarité des quatre paires de références est composé de :

Variables. Les quatre variables correspondant aux scores de similarité des quatre paires de références sont :

- $x_1 = Sim_r(< S1_m1, S2_m1 >)$
- $x_2 = Sim_r(< S1_p1, S2_p1 >)$
- $x_3 = Sim_r(< S1_p1, S2_p2 >)$
- $x_4 = Sim_r(< S1_c1, S2_c1 >)$

Constantes. A toute paire de valeurs d'attributs, on associe une constante b_{il} qui représente son score de similarité. Nous avons utilisé la mesure Jaro-Winkler [CRF] pour calculer la valeur de leur score de similarité.

- $b_{11} = Sim_v(\text{“}LOUVRE'', \text{“}le\ Louvre'') = 0.88$
- $b_{12} = Sim_v(\text{“}99,\ r.\ RIVOLI'', \text{“}rue\ Rivoli'') = 0.76$
- $b_{21} = Sim_v(\text{“}La\ Joconde'', \text{“}Abricotiers\ en\ fleurs'') = 0.0$
- $b_{31} = Sim_v(\text{“}La\ Joconde'', \text{“}Joconde'') = 0.9$
- $b_{41} = Sim_v(\text{“}Paris'', \text{“}Ville\ de\ Paris'') = 0.42$

Pondérations. A toute relation et à tout attribut intervenant dans le calcul de similarité des paires de références, est associée une pondération calculée en fonction de la taille des ensembles des attributs communs et des relations communes des paires de références. Nous avons donc les constantes suivantes :

- $\lambda_{11} = \frac{1}{|CAttr(<S1_m1,S2_m1>)| + |CRel(<S1_m1,S2_m1>)|} = \frac{1}{4}$
- $\lambda_{21} = \frac{1}{|CAttr(<S1_p1,S2_p1>)| + |CRel(<S1_p1,S2_p1>)|} = \frac{1}{2}$
- $\lambda_{31} = \frac{1}{|CAttr(<S1_p1,S2_p2>)| + |CRel(<S1_p1,S2_p2>)|} = \frac{1}{2}$
- $\lambda_{41} = \frac{1}{|CAttr(<S1_c1,S2_c1>)| + |CRel(<S1_c1,S2_c1>)|} = \frac{1}{2}$

Précision du point fixe. Considérons une valeur de ϵ égale à 0.005

Calcul itératif des scores de similarité. En considérant les ensembles d'attributs et de relations dans lesquels les paires de références sont en dépendance fonctionnelle et ceux dans lesquels elles ne sont pas en dépendance fonctionnelle (voir la définition en Section 7.4.1), nous exprimons le calcul de similarité comme suit :

- $x_1 = \max(\max(x_2, x_3, moy(b_{11}, b_{12})), (\frac{1}{4} * x_4))$
- $x_2 = \max(b_{21}, (\lambda_{21} * x_1))$
- $x_3 = \max(b_{31}, (\lambda_{31} * x_1))$
- $x_4 = \max(\max(x_1, b_{41}))$

La solution du système d'équations est représentée dans le vecteur $X = (0.9, 0.45, 0.9, 0.9)$, ce qui correspond aux scores de similarité des quatre paires de références considérées dans le calcul. Cette solution a été obtenue après 4 itérations où le calcul a atteint un point-fixe. En effet, le vecteur erreur calculé pour les valeurs des itérations 3 et 4, vaut 0 ce qui est inférieur à ϵ.

Itérations	0	1	2	3	4
$x_1 = \max(\max(x_2, x_3, moy(0.88, 0.76)), (\frac{1}{4} * x_4))$	0	0.82	0.9	0.9	0.9
$x_2 = \max(0.0, (\frac{1}{2} * x_1))$	0	0	0.41	0.45	0.45
$x_3 = \max(0.9, (\frac{1}{2} * x_1))$	0	0.9	0.9	0.9	0.9
$x_4 = \max(\max(x_1, 0.42))$	0	0.42	0.82	0.9	0.9

TAB. 7.3 – *Illustration du calcul – itération du calcul de similarité.*

Dans cet exemple, dont le calcul est présenté en Table 7.3, nous avons montré comment les scores de similarité sont propagés à travers les liens de dépendance fonctionnelle entre les paires de références c'est-à-dire, à travers des relations et des attributs ayant un impact fort sur la similarité des paires de références. Par exemple à l'itération (2), la similarité 0.9 de la paire de peintures $< S1_p1, S2_p2 >$ a été propagée à la paire de musées $< S1_m1, S2_m1 >$ à travers la relation *contient* qui appartient à l'ensemble $DF_R(< S1_m1, S2_m1 >)$. À l'itération suivante (3) ce même score de similarité a été propagé à la paire de villes $< S1_c1, S2_c1 >$ à travers la relation $localiseA$ qui appartient à l'ensemble $DF_R(< S1_c1, S2_c1 >)$. Ceci est cohérent avec les résultats de L2R, si on réconcilie deux musées alors on réconcilie les deux villes de leur localisation ($PF(localiseA)$). De plus, nous avons également une propagation plus faible des scores de similarité vers des paires de références qui ne sont pas en dépendance fonctionnelle. Par exemple, à l'itération (3) le score de similarité des musées 0.90 obtenu à l'itération (2) a été propagé à la paire de peintures $< S1_p1, S2_p1 >$, qui lui a donné 0.45 comme valeur de score de similarité.

En ce qui concerne les décisions de réconciliation, si nous posons par exemple un seuil de réconciliation $T_{rec} = 0.80$, nous réconcilions les trois paires de références : deux villes, deux musées et deux œuvres.

7.5 Algorithme de calcul itératif de la similarité des références

L'algorithme itératif de calcul de similarité pour les paires de références est un algorithme qui est fondé sur la méthode de résolution itérative de systèmes d'équations non linéaires. Nous avons adapté la méthode de *Jacobi* pour résoudre notre système d'équations non linéaire. Cette méthode comporte deux étapes principales : (i) initialisation du vecteur de variables du système d'équations et (ii) itération du calcul des valeurs des variables du système d'équations en utilisant leur valeur calculée à l'itération précédente. Cette méthode termine quand le vecteur erreur e^k est inférieur à une constante ϵ représentant la précision du point fixe que nous souhaitons obtenir.

Dans cette section, nous décrivons l'algorithme général de réconciliation de références que nous proposons dans la méthode N2R. Nous donnons une estimation de sa complexité en Section 7.5.2.

7.5.1 Présentation de l'algorithme général

Dans l'algorithme 6, nous décrivons les différentes étapes nécessaires pour calculer les scores de similarité des paires de références sur lesquelles les décisions de réconciliation sont fondées. Cet algorithme comprend quatre étapes principales.

Algorithm 6 N2R : ALGORITHME GENERAL DE RECONCILIATION DE REFERENCES

Inputs : P : ensemble de paires de références.
D : ensemble des descriptions des références sous la forme de faits RDF.
A : ensemble des axiomes de fonctionnalité et de fonctionnalité inverse des propriétés.
$L2RRes$: ensemble de résultats de réconciliation de références et de (non) synonymies.
T_{rec} : seuil de réconciliation des paires de références
ϵ : la valeur de la précision du point fixe.

Output : REC_{n2r} : ensemble de paires de références réconciliées.

*//**Étape 1** : Construction de la description commune des références*
1: $CD \leftarrow$ Description-Commune(P, D) ;
2: $DfNdf \leftarrow$ Construction-Dependances-Fonctionnelles(CD, A) ;
3: $\Lambda \leftarrow$ Calcul-Ponderations(CD) ;
*//**Étape 2** : Construction et initialisation du système d'équations non linéaire*
4: $B \leftarrow$ Initialisation-Const(CD, $DfNdf$, $L2RRes.SynV$, $L2RRes.NSynV$, Λ) ;
5: $X^0 \leftarrow$ Initialisation-Variables(CD, $DfNdf$, $L2RRes.Rec$, Λ) ;
*//**Étape 3** : calcul itératif des scores de similarité des paires de références*
6: $X \leftarrow$ Resolution-Iterative-Systeme-Equations(X^0,B, Λ, ϵ) ;
*//**Étape 4** : Prise des décisions de réconciliation pour les paires de références ayant un score $x_i \geq T_{rec}$*
7: $REC_{n2r} \leftarrow \emptyset$
8: **Foreach** ($x_i \in X$) **Do**
9: **If** ($x_i \geq T_{rec}$) **Then**
10: $REC_{n2r} \leftarrow REC_{n2r} \cup P[i]$;
11: **End If**
12: **End For**
13: **Return** REC_{n2r} ;

Description des paramètres de l'algorithme 6.

- *Le paramètre* P : ensemble de paires de références appartenant à l'ensemble $I1 \times I2$. Sur cet ensemble de paires de références, un filtrage peut avoir été effectué en pré-traitement en exploitant des connaissances telles que la disjonction entre classes, l'hypothèse du nom unique (UNA) et les non réconciliations inférées par L2R. On note $P[i]$ la i-ème paire de références de P.

- *Le paramètre* D : descriptions des références (notée $Desc(ref)$ pour une référence ref) sous la forme d'un ensemble de faits-classes, de faits-relations et de faits-attributs portant sur les références appartenant à P.

- *Le paramètre* A : ensemble des axiomes de fonctionnalité et de fonctionnalité inverse déclarés sur des relations et des attributs du schéma RDFS+.

- *Le paramètre* $L2RRes$: dans le cas où l'application de la méthode N2R a été précédée par l'application de L2R, nous exploitons les résultats obtenus par L2R. Ils consistent en un ensemble de paires de références réconciliées (noté $L2RRes.Rec$), de paires de valeurs de

base synonymes (noté $L2RRes.SynVals$) et de paires de valeurs de base non synonymes (noté $L2RRes.NSynVals$).

- *Le paramètre* T_{rec} : une valeur réelle appartenant à [0..1] qui représente le seuil de similarité utilisé pour prendre des décisions de réconciliation des paires de références.

- *Le paramètre* ϵ : une valeur réelle fixée très petite appartenant à [0..1] et qui représente la valeur de la précision du point fixe.

Étape 1 : Construction des descriptions communes et des dépendances. C'est une étape où l'on construit les composantes élémentaires pour notre calcul de similarité. Il s'agit de la description commune des paires de références (instruction 1 de l'algorithme 6) et des dépendances fonctionnelles ou non fonctionnelles existant entre les différentes références (instruction 2 de l'algorithme 6).

⋆ *Description commune aux différentes paires de références* : cette tâche est effectuée par la primitive DESCRIPTION-COMMUNE(P, D) qui permet de construire l'ensemble des attributs communs $CAttr(< ref, ref' >)$ et l'ensemble des relations communes $CRel(< ref, ref' >)$, pour toute paire de références $< ref, ref' >$. Ensuite, à chaque référence ref on construit l'ensemble des valeurs associées par les attributs $(a^+(ref))$, l'ensemble de références associées par des relations $(r^+(ref))$ et l'ensemble de références auxquelles la référence ref est associée par des relations $(r^-(ref))$. Le résultat de cette primitive est stocké dans la variable locale CD. Cette variable est un ensemble contenant quatre *tables de hachage*[25] :

1. La première représente l'association de l'ensemble d'attributs communs à toute paire de références.
2. La deuxième représente l'association de l'ensemble de relations communes à toute paire de références.
3. La troisième est une table de hachage qui représente l'association de l'ensemble des paires *attribut–valeur* de la forme $(a_j, v1)$ à toute référence.
4. La quatrième représente l'association de l'ensemble de paires *relation–référence* de la forme $(r_j, ref1)$ à toute référence.

⋆ *Dépendance et non dépendance fonctionnelle des paires de références* : cette tâche est effectuée par la primitive CONSTRUCTION-DEPENDANCES-FONCTIONNELLES(CD, A) qui, à partir des ensembles d'attributs communs, de relations communes et de l'ensemble d'axiomes de fonctionnalité et de fonctionnalité inverse des relations et des attributs, détermine l'importance des relations et des attributs dans le calcul de similarité. Le résultat de cette primitive est stocké dans la variable locale $DfNdf$. Elle représente pour chaque paire de références $< ref, ref' >$ les ensembles :

1. d'attributs $DF_A(< ref, ref' >)$ avec lesquels la paire de références est en dépendance fonctionnelle.

[25] [wikipedia] En informatique, une table de hachage est une structure de données qui permet une association clé-élément

2. d'attributs $DF_A^M(< ref, ref' >)$ avec lesquels la paire de références est en dépendance fonctionnelle multiple.
3. de relations $DF_R(< ref, ref' >)$ avec lesquelles la paire de références est en dépendance fonctionnelle.
4. de relations $DF_R^M(< ref, ref' >)$ avec lesquelles la paire de références est en dépendance fonctionnelle multiple.
5. d'attributs $NDF_A(< ref, ref' >)$ avec lesquels la paire de références n'est pas en dépendance fonctionnelle et tels que ces attributs ont une seule valeur .
6. d'attributs $NDF_A^*(< ref, ref' >)$ avec lesquels la paire de références n'est pas en dépendance fonctionnelle et tels que ces attributs ont plusieurs valeurs. Ceci nécessitera l'utilisation de la fonction $SSim_v$ de calcul de similarité d'ensembles de valeurs.
7. de relations $NDF_R(< ref, ref' >)$ avec lesquelles la paire de références n'est pas en dépendance fonctionnelle des références qu'elles lient et telles que ces relations ont une seule référence.
8. de relations $NDF_R^*(< ref, ref' >)$ avec lesquelles la paire de références n'est pas en dépendance fonctionnelle et telles que ces relations ont plusieurs références (multi-valuées). Ceci nécessitera l'utilisation de la fonction $SSim_r$ pour le calcul de similarité d'ensembles de références.

La définition de ces ensembles en fonction des axiomes de fonctionnalité PF et de fonctionnalité inverse PFI est donnée en Section 7.4.1.

⋆ *Calcul des pondérations associées aux attributs et aux relations* : pour chaque paire de références $< ref, ref' >$, une pondération λ_{ij} est calculée à partir des ensembles de relations et des attributs communs instanciés de cette paire. $\lambda_{ij} = \frac{1}{|CAttr(<ref,ref'>)|+|CRel(<ref,ref'>)|}$

Cette tâche est effectuée par la primitive CALCUL-PONDERATIONS(CD) (instruction 3 de l'algorithme 6). Le résultat de cette primitive est une table de hachage Λ où l'on associe à chaque paire de références la valeur de λ_{ij}.

Étape 2 : Construction du système d'équations. Dans cette étape nous construisons le vecteur de constantes et le vecteur de variables sur lesquels le système d'équations est fondé. Des valeurs initiales sont soit calculées soit fixées pour chacune des constantes et chacune des variables.

⋆ ***Les constantes***. L'initialisation des constantes est effectuée par la primitive INITIALISATION-CONST(CD, $DfNdf$, $L2RRes.SynV$, $L2RRes.NSynV$, Λ), cf. l'instruction 4 de l'algorithme 6. Tout d'abord, pour chaque paire de références $< ref, ref' >$, on calcule le score de similarité de toute paire de valeurs et pour toute paire d'ensembles de valeurs qui lui sont associées par tout attribut commun $(a^+(ref) \times a^+(ref'))$. Nous rappelons que les scores de similarité des paires de valeurs des attributs sont calculés par des algorithmes existants de mesure de similarité de valeurs de base, tels que ceux présentés dans Cohen et al. [CRF]. Pour les ensembles de valeurs, nous pouvons utiliser $SoftJaccard_v$ la mesure de similarité entre ensembles de valeurs que nous avons présentée en Section 7.2.3. Concernant les paires de valeurs apparaissant comme synonymes dans l'ensemble $L2RRes.SynV$, un score de similarité à 1 leur est directement associé (nous ne calculons pas de score de similarité pour ces valeurs). De même, pour les paires de valeurs apparaissant comme non

synonymes dans $L2RRes.NSynV$, un score de similarité à 0 leur est directement associé. Ensuite, nous faisons correspondre : (i) tout score de similarité d'une paire de valeurs à une constante b_{il} et nous ajoutons un indice *df*, *dfm* ou *ndf* selon le type de l'attribut dans $DfNdf$ et (ii) tout score de similarité d'une paire d'ensembles de valeurs à une constante BS_{il} et nous ajoutons un indice *df*, *dfm* ou *ndf* selon le type de l'attribut dans $DfNdf$.

⋆ ***Les variables***. L'initialisation des variables est effectuée par la primitive INITIALISATION-VARIABLES(CD, $DfNdf$, $L2RRes.Rec$, Λ), cf. l'instruction 5 de l'algorithme 6. Tout d'abord, nous faisons correspondre : (i) à toute paire de références une variable x_i initialisée à 1 si la paire de références à déjà été réconciliée par L2R et à 0 sinon ; (ii) pour toute paire de références intervenant dans le calcul de similarité d'une autre paire de références, nous ajoutons le rang j de la relation qui les lient, notée x_{ij}. Ensuite, nous ajoutons un indice *df*, *dfm* ou *ndf* selon le type de la relation dans $DfNdf$ et (iii) toute paire d'ensembles de références d'une relation de rang j à une variable XS_{ij} initialisée à 0 et nous ajoutons un indice *df*, *dfm* ou *ndf* selon le type de la relation dans $DfNdf$. On rappelle que la valeur de la variable XS_{ij} est calculée en utilisant la fonction $SSim_r$ selon la mesure $SoftJaccard_r$, par exemple.

Étape 3 : Résolution itérative du système d'équations non linéaire. Dans cette étape, nous effectuons la résolution itérative du système d'équations construit et initialisé à la deuxième étape. Nous rappelons que la méthode que nous proposons pour résoudre notre système d'équations non linéaire est une adaptation de la méthode itérative de *Jacobi* pour la résolution des systèmes d'équation linéaires. Tout d'abord, la condition de terminaison de notre algorithme est fondée sur le calcul d'un vecteur erreur e^k dont chaque composante représente la valeur absolue de la différence des valeurs des x_i^k et des x_i^{k-1} avec k le numéro de l'itération courante et i le rang de la paire de références. Initialement, le vecteur e^k est assigné à 2 (cf. instruction 3 de l'algorithme 7). L'algorithme opère itérativement un calcul des valeurs des variables x_i^k en utilisant les valeurs de x_i^{k-1} jusqu'à ce que la condition de terminaison soit satisfaite.

⋆ ***Condition de terminaison***. Il s'agit de comparer les valeurs du vecteur erreur e^k par rapport à la valeur de la précision ϵ. Cela est effectué par la primitive VECTEUR-ERREUR-SUP(e^k, ϵ) à l'instruction 4 de l'algorithme 7. Cette primitive permet de vérifier si la valeur de toute composante du vecteur e^k est supérieure à ϵ. Si c'est le cas alors l'algorithme exécute l'itération suivante sinon, l'algorithme s'arrête et le point fixe à précision ϵ est considéré comme atteint. D'après la preuve de convergence du calcul itératif de similarité (voir en Annexe A) cette condition peut toujours être satisfaite après un nombre fini k d'itérations.

⋆ ***Itérations***. À chaque itération k les scores de similarité des n paires de références sont mis à jour en utilisant les scores de similarité des paires de références, calculés à l'itération $k-1$, dont ils sont fonction. Il s'agit donc pour chaque paire de rang i avec $0 \leq i \leq n$ de calculer son score de similarité x_i^k qui est obtenu comme suit :

1. Calculer le maximum des scores de similarité (instruction 6 de l'algorithme 7) : (i) des paires de valeurs représentées par les constantes que nous récupérons par $B.df(x_i^k)$ ou de la moyenne des paires de valeurs représentées par des constantes que nous récupérons par $B.dfm(x_i^k)$ et (ii) des paires de références représentées par les variables que

nous récupérons par $X^{k-1}.df(x_i^k)$ ou de la moyenne des paires de références représentées par des constantes que nous récupérons par $X^{k-1}.dfm(x_i^k)$ dont la i-ème paire de références est en dépendance fonctionnelle (DF) ou en dépendance fonctionnelle multiple (DF^M).

2. Calculer la moyenne pondérée des scores de similarité (instruction 7 de l'algorithme 7) : (i) des paires de valeurs représentées par les constantes que nous récupérons par $B.ndf(x_i^k)$, (ii) des ensembles de valeurs représentées par les constantes que nous récupérons par $B.ndfS(x_i^k)$, (iii) des paires de références représentées par les variables que nous récupérons par $X^{k-1}.ndf(x_i^k)$ et (iv) des ensembles de références représentés par les variables que nous récupérons par $X^{k-1}.ndfS(x_i^k)$ dont la i-ème paire de références n'est pas en dépendance fonctionnelle (NDF).

3. Calculer le maximum des deux valeurs obtenues ci-dessus et ainsi nous mettons à jour la valeur de la variable x_i^k (instruction 8 de l'algorithme 7).
4. Mise à jour de la valeur de la i-ème composante du vecteur erreur, correspondant à la variable x_i^k, par la valeur absolue de la différence des valeurs de x_i^k et de x_i^{k-1} (instruction 9 de l'algorithme 7).
5. Passer à la variable x_{i+1}^k suivante.

⋆ ***Résultat.*** L'algorithme retourne le vecteur X^k représentant les scores de similarité des n paires de références calculés à la k-ème itération durant laquelle le point fixe a été atteint.

Étape 4 : Décision de réconciliation. C'est la dernière étape de l'algorithme de réconciliation de références où l'on prend les décisions de réconciliation pour les paires de référence qui satisfont la condition de réconciliation. Toute paire de références ayant un score de similarité supérieur à la valeur du seuil T_{rec} donné en paramètre est considérée comme réconciliée. Elle est donc ajoutée à l'ensemble REC_{n2r}. Cette étape est décrite par les instructions 7 à 12 de l'algorithme 6.

Algorithm 7 N2R : ALGORITHME DE RESOLUTION ITERATIVE DU SYSTEME D'EQUATIONS

Inputs : X^0 : un vecteur de valeurs initiales associées aux variables x_i du système d'équations.
B : un vecteur de valeurs associées aux constantes b_i du système d'équations.
Λ : pondération des relations et des attributs dans le calcul de la moyenne pondérée.
ϵ : la valeur réelle très petite dans [0..1] représentant la précision du point fixe souhaitée.

Output : X^k : un vecteur de valeurs calculées pour les variables x_i après k itérations.

1: $X^{k-1} \leftarrow X^0$;
2: $X^k \leftarrow X^{k-1}$;
//Initialisation du vecteur erreur $e^k = (e_1^k, e_2^k, \ldots, e_n^k)$ *à* $(2, 2, \ldots, 2)$
3: $e^k \leftarrow (2, 2, \ldots, 2)$;
4: **While** (Vecteur-Erreur-Sup(e^k, ϵ)) **Do**
5: **Foreach** ($x_i^k \in X^k$) **Do**
6: $f_{i-df}(X^{k-1}) \leftarrow \max(B.df(x_i^k), moy(B.dfm(x_i^k)), X^{k-1}.df(x_i^k), moy(X^{k-1}.dfm(x_i^k)))$;
7: $f_{i-ndf}(X^{k-1}) \leftarrow \text{moyP}(\Lambda[i], B.ndf(x_i^k), B.ndfS(x_i^k), X^{k-1}.ndf(x_i^k), X^{k-1}.ndfS(x_i^k))$;
8: $x_i^k \leftarrow \max(f_{i-df}(X^{k-1}), f_{i-ndf}(X^{k-1}))$;
9: $e_i^k \leftarrow |x_i^k - x_i^{k-1}|$;
10: **End For**
11: $X^{k-1} \leftarrow X^k$;
12: **End While**
13: **Return** X^k ;

7.5.2 Complexité de l'algorithme du calcul itératif de similarité

Pour obtenir la complexité de l'algorithme, nous allons manipuler les paramètres suivants :

- n, le nombre de paires de références considérées dans le calcul ;
- n_{ra}, le nombre maximum de références associées à une référence par une relation.
- n_{ac}, le nombre maximum d'attributs communs pour une paire de références ;
- n_{rc}, le nombre maximum de relations communes pour une paire de références ;

Parmi tous ces paramètres, le nombre n de paires de références considérées dans le calcul est le paramètre le plus influant sur la complexité de l'algorithme. Les autres paramètres sont bornés par une valeur dépendante du domaine. Le nombre n_{ra} est égal à 1 pour les relations mono-valuées et est inférieur à une centaine de relations pour les relations et les attributs multi-valués. Si le nombre de références associées est grand, ce qui en théorie peut arriver, ce paramètre devient alors aussi important que n.

Nous estimons tout d'abord la complexité du calcul de similarité pour une itération. À chaque itération, pour chaque paire de références le calcul de la fonction $f_i(X)$ s'effectue en un temps $n * (n_{ac} + (n_{rc} * n_{ra}))$. Ce qui nous donne donc, pour le calcul de similarité de toutes les paires, pour une itération une complexité de $n^2 * (n_{ac} + n_{rc} * n_{ra})$. Pour un nombre d'itérations k la complexité est de $(k * n^2 * (n_{ac} + n_{rc} * n_{ra}))$. On note que dans le cas où chaque équation est exprimée uniquement en fonction d'au moins une variable ou une constante de type *df* ou *dfm* (prises en compte dans la partie maximum) nous pouvons assurer qu'à chaque itération au moins une variable prend sa valeur définitive. Par conséquent, la valeur de k est, dans le pire des cas,

égale à n. Ce qui nous donne donc dans le pire des cas une complexité de $O(n^3)$. En revanche, dans les autres cas nous n'avons qu'une évaluation empirique de la valeur de k et nous avons remarqué que pour des valeurs grande de n la valeur de k reste très faible. Sur les jeux de données expérimentées dans N2R nous avons :

- *Cora* : pour $n = (6108 * 6108)$, N2R a atteint la convergence pour un $k = 9$;
- *HOTELS* : pour $n = (404 * 1392)$, N2R a atteint la convergence pour un $k = 1$.

On note également que dans le cas où le nombre de références associées n_{ra} (la taille des ensembles de références) est de l'ordre de n alors nous obtenons une complexité en n^3 pour cette étape. On note que ce cas est très rare en pratique, donc l'estimation de la complexité à une complexité quadratique est plus juste.

L'étape de décision de réconciliation est effectuée en un temps linéaire n, car il suffit de parcourir l'ensemble des variables et de comparer leur valeur au seuil de réconciliation T_{rec}. Donc la complexité de cette opération est : n.

Nous obtenons donc une complexité estimée à :

$$[n] + [k * n^2 * (n_{ac} + n_{rc} * n_{ra})]$$

Après la simplification de tous les paramètres bornés, nous obtenons alors une complexité globale qui est quadratique en $O(n^2)$

7.6 Expérimentations

La méthode numérique de réconciliation de références que nous avons présentée dans les sections précédentes a été implémentée et expérimentée sur les mêmes jeux de données utilisés dans l'expérimentation de L2R. Il s'agit d'un jeu de données FT_HOTELS du domaine du tourisme et d'un jeu de données *Cora* du domaine des publications scientifiques en informatique. Une présentation détaillée des deux jeux de données est donnée en Section 6.5.1 du chapitre 6.

Nous avons expérimenté notre méthode N2R dans deux cas :

⋆ application de la méthode N2R sans l'exploitation des résultats obtenus par la méthode L2R sur le même jeu de données.

⋆ application de la méthode N2R en exploitant les résultats obtenus par la méthode L2R sur le même jeu de données : les décisions de réconciliation et de (non) synonymies.

Pour valider les résultats que nous avons obtenus par N2R et LN2R, nous avons évalué les résultats en mesurant le rappel, la précision et la F-Mesure. Dans la méthode L2R les décisions de réconciliation et de non réconciliation entre références obtenues sont correctes car elles sont obtenues par un algorithme d'inférence logique. Dans la méthode N2R les décisions de réconciliation sont fondées sur les scores de similarité des paires de références et sur un seuil de réconciliation donné. Donc, à la différence de L2R, nous calculons aussi la précision qui peut être différente de 100% à cause des erreurs qui peuvent surgir dans les décisions de réconciliation. De plus, à la différence de L2R, dans la méthode N2R nous n'effectuons que les décisions de réconciliations. Donc les mesures d'évaluation des résultats (rappel, précision et F-Mesure) ne sont calculées que sur l'ensemble des réconciliations REC_{n2r}.

Comme nous n'avons pas de moyens automatiques pour fixer le seuil de réconciliation T_{rec}, nous l'avons fixé de manière expérimentale. Autrement dit, nous avons fait varier la valeur du seuil entre 1 et 0.5 en baissant sa valeur de 0.05 à chaque étape de test. À chaque étape de test nous relevons les résultats concernant les décisions de réconciliation que nous pouvons prendre en fonction de la valeur courante du seuil. La précision ϵ du calcul itératif a été fixé à une valeur de 0.005, pour les deux jeux de données, ce qui a aboutit à un nombre d'itérations maximal de 10. Ensuite, en fonction de ces résultats, nous calculons pour chaque valeur du seuil, le rappel et la précision. Nous calculons également la F-Mesure qui permet en quelque sorte de mesurer le compromis entre le rappel et la précision. En effet, nous allons voir dans les résultats d'expérimentation, que pour une valeur de seuil élevée, nous pouvons avoir une précision maximale mais un rappel insuffisant et pour une valeur faible du seuil le phénomène inverse se produit.

Nous présentons, dans ce qui suit, les résultats obtenus par la méthode N2R, sans l'exploitation des résultats de L2R, sur les jeu de données FT_HOTELS et *Cora*.

Dans cette expérimentation pour mesurer la similarité des paires de valeurs des attributs nous avons utilisé le package open-source rendu disponible par William Cohen à l'adresse [Coh03]. Pour calculer le score de similarité des ensembles de valeurs et de références nous avons utilisé la mesure $SoftJaccard_v$ et $SoftJaccard_r$.

7.6.1 Résultats de N2R obtenus sur le jeu de données *Cora*

Avant de présenter les résultats, voici les axiomes de fonctionnalité et de fonctionnalité inverse déclarés au niveau du schéma RDFS+ du jeux de données *Cora*.

- *axiomes de fonctionnalité des propriétés* : toutes les propriétés sont fonctionnelles, sauf la relation *AuthoredBy*, ce qui signifie qu'un article peut avoir plusieurs auteurs ;

- *axiomes de fonctionnalité inverse des propriétés* : nous déclarons trois axiomes de fonctionnalité inverse des propriétés. Le premier concerne la classe *Article* $PFI(Title, Year, Type)$, il déclare que deux articles ayant le même titre, la même année de publication et le même type de publication (e.g. journal, proceedings, report) doivent être réconciliés. Le deuxième concerne la classe *Conference* $PFI(ConfName, ConfYear)$, il déclare que deux conférences ayant le même nom et qui se sont déroulées à la même année doivent être réconciliées ; et le troisième concerne la classe *Person* $PFI(Name)$ et il déclare que dans ce jeu de données une même personne ne peut pas avoir deux noms différents.

Ces différents axiomes sont exploités par notre méthode N2R pour déterminer le type de l'impact (df, dfm et ndf) des différents attributs et relations sur le score de similarité des paires de références.

Les résultats présentés en Figure 7.2 montrent, tout d'abord, que les résultats obtenus par N2R sont en cohérence avec ceux obtenus par L2R. Au seuil $T_{rec} = 1$, l'ensemble des réconciliations obtenues par L2R ont été aussi obtenues par N2R. Cela est dû à l'exploitation des axiomes de fonctionnalité et de fonctionnalité inverse par la méthode N2R et au système de propagation des scores de similarité qui est également induit des axiomes du schéma. Nous remarquons également l'évolution intéressante des valeurs du rappel et de la précision en fonction du seuil T_{rec}. En effet, lorsque le seuil varie entre 1,0 et 0,85 on note une augmentation du rappel de 33% alors que la précision ne chute que de 6%. Les meilleurs résultats de réconciliation obtenus par N2R se produisent lorsque le seuil T_{rec} vaut 0,85, car c'est là où la valeur de la F-mesure est à son maximum, 88%. De plus, lorsque la valeur du rappel est presque à 100% pour un seuil $T_{rec} = 0,5$

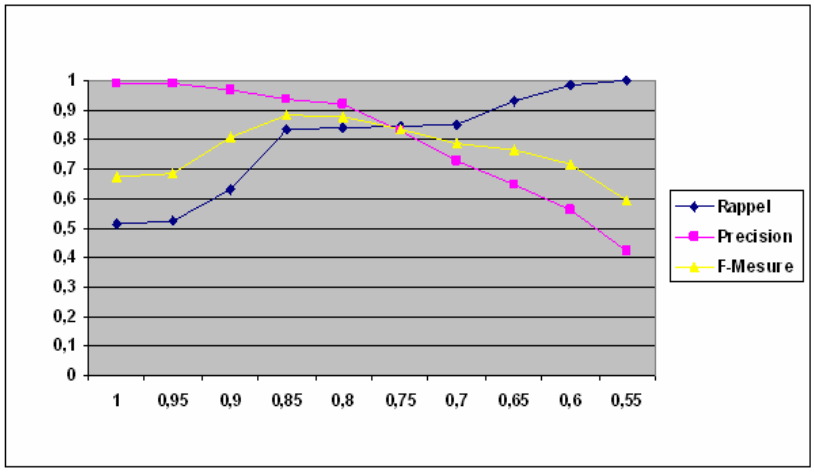

FIG. 7.2 – Résultats obtenus par N2R sur *Cora*

la valeur de la précision est quand même d'environ 40%.

Ces résultats montrent également l'intérêt d'une méthode complètement automatique et non supervisée. En effet, la valeur de la F-Mesure, obtenue par N2R, est meilleure que celle obtenue sur le même jeu de données par Parag et Domingo [SD05] qui est de 87%. En revanche, les résultats obtenus par Dong et al. [DHM05] par une méthode supervisée sont meilleurs que ceux obtenus par N2R. Il serait donc intéressant d'envisager d'améliorer notre méthode N2R par exemple en proposant un moyen d'affiner la pondération des attributs et des relations en exploitant les résultats obtenus par L2R pour étiqueter automatiquement un échantillon de données.

Dans le cas où la méthode N2R, appliquée sur le jeu de donnée *Cora*, est précédée par la méthode L2R nous obtenons une amélioration légère des résultats. Lorsque le seuil T_{rec} est à 0,85, valeur à laquelle la valeur de la F-Mesure est à son maximum, nous obtenons une augmentation de 0,2 % pour le rappel et une augmentation de 0,02 % pour la précision ce qui donne une augmentation de 0,2 % pour la F-Mesure. Cette faible amélioration des résultats est due au fait que les synonymies inférées par L2R pour ce jeu de données sont pour la plupart syntaxiquement très proches. Il s'agit donc de synonymies pour lesquelles N2R trouve un score de similarité élevé. De plus, L2R n'a pas permis d'inférer des non synonymies pour ce jeu de données. Toutefois, l'exploitation des non réconciliations inférées par L2R permettent de réduire très significativement la taille de l'espace de réconciliation considéré dans N2R. Pour le jeu de données *Cora* la taille de l'espace de réconciliation qui est environ de 37 millions de paires de références a pu être réduit de 32,8 % grâce aux non réconciliations certaines inférées par la méthode L2R, ce qui représente environ 12 millions paires de références. On peut également noter que les réconciliations obtenues pas L2R ne sont pas re-calculées par N2R.

7.6.2 Résultats de N2R obtenus sur le jeu de données FT_HOTELS

Avant de présenter les résultats obtenus par N2R pour les données FT_HOTELS, nous rappelons, tout d'abord, les axiomes déclarés au niveau du schéma RDFS+ et au niveau des sources de données :

- *axiome de disjonction* : $DISJOINT(Establishment, Service)$
- *axiomes de fonctionnalité sur les propriétés* : toutes les propriétés sont fonctionnelles, sauf *OtherService*, *OtherDescription* et *URLEstablishment* ;

- *axiomes de fonctionnalité inverse des propriétés* : nous considérons un seul axiome PFI(*EstablishmentName*, *AssociatedAddress*) qui exprime le fait que la combinaison des deux attributs *EstablishmentName* et *AssociatedAddress* forment une fonctionnalité inverse : deux établissements doivent être réconciliés s'ils possèdent le même nom et la même adresse.
- *axiome de l'UNA* : cet axiome est déclaré pour toutes les sources de données considérées.

On rappelle que ces différents axiomes sont exploités par notre méthode N2R pour déterminer le type de l'impact (*df*, *dfm* et *ndf*) des différents attributs et relations sur le score de similarité des paires de références.

Les résultats obtenus par l'application de la méthode N2R sur le jeu de données FT_HOTELS sont présentés en Figure 7.3. Ces résultats confirment la cohérence des résultats de la méthode numérique N2R avec ceux obtenus par la méthode logique L2R. En effet, pour un seuil $T_{rec} = 1$, N2R obtient exactement les même résultats que ceux de L2R : un rappel à 54 % et une précision à 100 %. Une autre observation importante est que lorsque le rappel augmente de 31 %, pour une valeur de seuil T_{rec} qui varie entre 1 et 0,70, la précision reste à 100%. Les meilleurs résultats obtenus par N2R sont obtenus lorsque le seuil de réconciliation $T_{rec} = 0,55$. En effet, c'est pour cette valeur de seuil que le valeur de F-Mesure est à sa valeur maximum 94 %, avec un rappel à 98 % et une précision de 91 %. Certaines fausses réconciliations sont dues au fait que certaines des références ont le même nom et une adresse différente et viceversa, par exemple, le nom d'hôtel "*Hostellerie du château*" est un nom d'hotel très fréquent dans les sources que nous avons considérées.

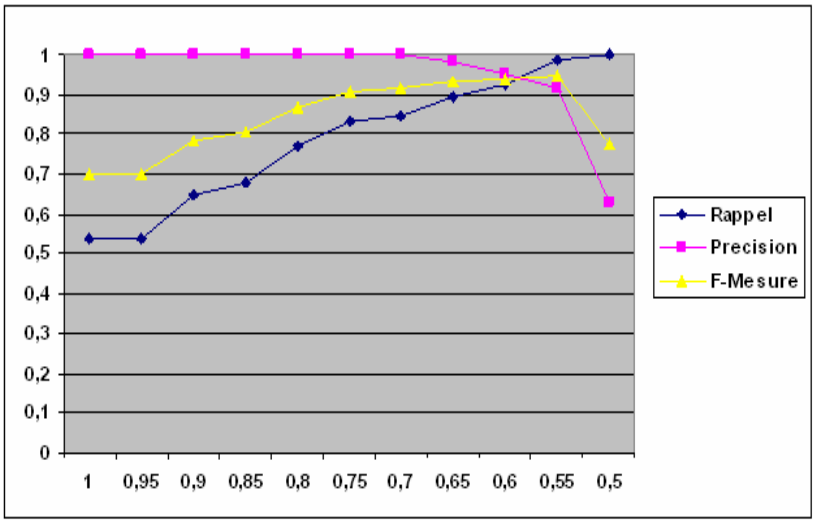

FIG. 7.3 – Résultats obtenus par N2R sur *FT_HOTELS*

Nous avons considéré le cas où la méthode N2R est précédée de l'application de la méthode L2R : les résultats que nous avons obtenus sont exactement les mêmes que ceux obtenus par l'application seule de N2R. Il s'agit encore une fois d'un cas où les réconciliations entre références et les synonymies entre valeurs n'ont pas contribué à une amélioration des résultats de N2R. Cependant, les non réconciliations inférées de manière certaines par L2R sont exploitées, dans N2R, par l'étape de filtrage des paires de références qui sont a priori non réconciliables. Grâce aux axiomes de disjonction entre hotels localisés dans des pays différents et aux axiomes de l'UNA, l'espace de réconciliation considéré dans N2R a pu être réduit de 75,9 % ce qui correspond à 427 212 paires de références parmi 562 368 paires de références au total.

En comparaison avec les résultats obtenus par N2R pour le jeu de données *Cora*, les meilleurs résultats (ie. meilleure valeur de la F-Mesure) ont été obtenus pour le jeu de données FT_HOTELS à une valeur de seuil de réconciliation T_{rec} sensiblement inférieure à celle du jeu de donnée *Cora*, $0,55$ vs $0,85$. Cette, différence est due en partie à la différence des caractéristiques des données. En effet, dans les données de FT_HOTELS, les valeurs sont syntaxiquement très variables : des langues différentes sont utilisées, des codifications différentes des valeurs (e.g. Y vs 1). Certains attributs ont des valeurs de grande taille, par exemple, une valeur de l'attribut *OtherDescription* est d'environ 80 mots (500 caractères), il s'agit d'attributs pour lesquels le score de similarité est souvent très faible. De plus, des attributs tels que *longitude* et *latitude* nécessitent des mesures de similarité plus adaptées au calcul de similarité des coordonnées géographiques.

7.7 Conclusion

Nous avons présenté dans ce chapitre notre méthode numérique de réconciliation de références (N2R). Cette méthode est fondée sur les connaissances déclarées sous forme d'axiomes sur les schéma et sur les données. Ces axiomes sont exploités par le calcul numérique des réconciliations entre paires de références, de façon à ce que les résultats de réconciliation soient cohérents avec ceux inférés logiquement par la méthode L2R. Les résultats d'expérimentation effectuées sur les mêmes jeux de données, ont pu confirmer la cohérence des résultats de N2R avec ceux obtenus par L2R.

Nous avons exprimé le calcul numérique des réconciliation dans un système d'équations non linéaire qui permet de calculer des scores de similarité pour les paires de références. Pour résoudre le système d'équations, nous avons adapté une méthode itérative de résolution de systèmes d'équations linéaire (Jacobi), pour laquelle nous avons prouvé la convergence. Cet aspect itératif permet, dans le cas où il existe une contrainte forte de temps d'execution, de fixer un nombre maximal d'itérations au bout duquel la méthode devra s'arrêter.

Par l'expérimentation de la méthode N2R sur le *benchmark Cora*, nous avons pu montrer la faisabilité d'une méthode complètement automatique, non supervisée et qui exploite les connaissances du schéma et des données. En effet, les résultats obtenus sont comparables à ceux obtenus par des approches supervisées [DHM05] et parfois meilleurs [SD05].

Conclusion et Perspectives

L'intégration de données a pour objectif de combiner des sources de données autonomes, distribuées et hétérogènes afin d'obtenir une vue homogène et uniforme des données intégrées. Pour y parvenir, toutes les données doivent être représentées selon un même schéma global et selon une sémantique unifiée. Cette tâche inclut deux problèmes : (i) la réconciliation des données avec le schéma global et (ii) la réconciliation de données conformes à un même schéma.

Pour traiter ces deux problèmes, nous avons montré dans cette thèse l'intérêt et la faisabilité d'approches complètement automatiques, non supervisées et guidées uniquement par une ontologie.

Nous présentons tout d'abord une synthèse de nos contributions et nous terminons par les différentes perspectives que nous avons dégagées pour nos travaux.

7.1 Résumé des contributions et discussion

Nous donnons, dans cette section, les apports et les originalités des solutions que nous avons proposées pour les problèmes de réconciliation.

Réconciliation de données avec un schéma.

Pour réconcilier des données avec un schéma, nous avons proposé une méthode permettant de décrire des informations semi-structurées représentées dans des tableaux en fonction des termes et des relations sémantiques déclarés dans l'ontologie du domaine. Nous avons formulé ce problème comme un enrichissement sémantique des informations structurées. La méthode d'enrichissement sémantique XTab2SML que nous avons proposée permet d'enrichir un tableau, donné dans un format XML indépendant du domaine (XTab), par la sémantique du domaine représenté dans l'ontologie. L'objectif est de présenter chaque ligne du tableau XTab par un ensemble de relations sémantiques parfaitement ou partiellement identifiées. De plus chaque valeur du tableau est associée à un ou plusieurs termes de l'ontologie qui lui sont similaires. Pour représenter le résultat de cet enrichissement sémantique, nous avons proposé un format nommé SML (*Semantic Markup Language*) qui est un format de documents XML où la plupart des tags et des valeurs proviennent de l'ontologie. La définition du type de ces documents SML est donnée dans une DTD qui peut être générée de manière automatique à partir de l'ontologie du domaine.

Le premier apport de notre méthode XTab2SML est le fait qu'elle permette de représenter des *ambiguïtés* qui seront fréquentes dans le cadre d'un traitement complètement automatique. En effet, dans le cas où plusieurs relations sémantiques ont été identifiées dans le tableau, elles sont alors toutes représentées dans le document SML résultat. Le même raisonnement est suivi lorsque l'on cherche à associer aux valeurs du tableau des termes de l'ontologie qui lui sont similaires : si plusieurs termes ont été trouvés comme correspondants à une valeur, ils sont alors

tous conservés dans le document SML.

Le deuxième apport de notre méthode XTab2SML est de conserver les informations qui sont *imparfaitement identifiées* et qui seront a priori nombreuses dans une approche complètement automatique. Ces dernières peuvent être exploitées lors de l'interrogation pour compléter les réponses retournées à l'utilisateur, ce qui peut conduire à préciser ou modifier son interprétation des réponses. Ces informations peuvent aussi être exploitées pour découvrir des relations sémantiques et des termes candidats à l'enrichissement de l'ontologie du domaine.

Le troisième apport de notre méthode XTab2SML est la représentation du résultat de l'enrichissement sémantique dans un format *flexible* permettant de garder trace des traitements effectués sur les données et du type de l'identification des termes et des relations sémantique de l'ontologie. En effet, dans les documents SML nous ajoutons des indicateurs de traitement permettant de conserver des méta-données telles que des informations contextuelles, des informations concernant les types de procédures d'appariement utilisées et des informations sur la présence d'informations imparfaitement identifiées. Toutes ces informations peuvent être exploitées par un moteur d'interrogation lui permettant ainsi de s'adapter aux données interrogées.

Le quatrième apport de notre méthode d'enrichissement sémantique est sa capacité à s'*auto-améliorer* au fur et à mesure de son utilisation. En effet, l'enrichissement de l'ontologie par de nouvelles relations et par de nouveaux termes permet à notre méthode XTab2SML d'identifier de plus en plus d'informations dans le tableau.

Enfin, une caractéristique potentiellement importante de notre approche est la généricité de la méthode. En effet, dans le cas d'un changement de domaine, et donc d'ontologie, il suffit de générer la DTD SML à partir de la nouvelle ontologie. Une autre caractéristique importante est l'aspect automatique de notre approche qui rend notre méthode facilement intégrable dans une plate-forme complètement automatique telle qu'une chaîne de services Web. La méthode a un bon rappel par construction puisque l'on garde toutes les interprétations candidates et toutes les relations parfaitement ou imparfaitement identifiées. La précision obtenue nous paraît assez élevée pour conclure qu'un enrichissement complètement automatique est faisable.

Réconciliation de données conformes à un même schéma.

Réconcilier des données conformes à même schéma consiste à décider si deux descriptions provenant de sources distinctes réfèrent ou non à la même entité du monde réel. Ce problème est un problème de réconciliation de références. Dans la seconde partie de cette thèse, nous avons étudié ce problème lorsque le schéma des données est décrit dans le langage RDFS étendu par certaines primitives de OWL-DL et par un ensemble de règles de Horn exprimées dans le langage SWRL. Nous avons nommé ce schéma RDFS+. Nous avons proposé deux méthodes qui sont : (i) *globales*, elles exploitent les relations entre références pour propager les scores de similarité et les décisions de (non) réconciliation ; (ii) *non-supervisées*, aucun échantillon étiqueté manuellement n'est utilisé et (iii) *déclaratives*, toutes les connaissances exploitées sont déclarées sous la forme d'axiomes au niveau du schéma RDFS+, ce qui garantit la généricité de l'approche que nous proposons. En effet, en cas de changement de domaine ou de sources de données il suffit de mettre à jour ou de modifier l'ensemble des axiomes déclarés.

La première méthode est une méthode logique, nommée L2R, qui exploite les connaissances déclarées sous la forme d'axiomes sur le schéma et sur les données, telles que des connaissances concernant la disjonction entre classes, la fonctionnalité des propriétés ainsi que des connaissances concernant l'hypothèse du nom unique. Ces axiomes sont traduits automatiquement dans un ensemble de règles de réconciliation et de non réconciliation entre références mais également sous

la forme d'un ensemble de règles de synonymies et de non synonymies entre valeurs de base. Ces règles traduisent de façon déclarative les dépendances entre les décisions de réconciliation qui découlent de la sémantique du schéma et des données. Pour inférer les réconciliations et les non réconciliations entre les références, nous avons utilisé la méthode de résolution unitaire que nous appliquons sur des clauses de Horn exprimées en logique du premier ordre sans symboles de fonction. Nous avons prouvé la complétude de cette méthode pour la déduction logique de tous les faits de réconciliation, de non réconciliation, de synonymie et de non synonymie déductibles à partir de notre base de connaissances. Nous avons effectué des expérimentations pour la méthode L2R sur des données réelles fournies par notre partenaire France Télécom dans le projet PICSEL3 [pic05] et sur des données d'un benchmark utilisées par d'autres travaux sur la réconciliation de références. Les résultats que nous avons obtenus sont comparables et parfois meilleurs que ceux obtenus par une méthode supervisée.

Le premier apport original de cette méthode logique est l'aspect certain des décisions de (non) réconciliation et de (non) synonymie qu'elle permet d'inférer. Ces résultats certains, bien que partiels, peuvent être exploités par une méthode numérique dans le calcul de similarité des paires de références, d'une part en ne recalculant pas leur similarité et d'autre part en les considérant dans le calcul de similarité d'autres références. De plus, ces résultats certains peuvent être exploités pour étiqueter automatiquement un échantillon de données. Des connaissances peuvent être apprises telles que les pondérations des attributs et des relations très utiles pour une méthode numérique fondée sur un calcul de similarité des références. Un deuxième apport de la méthode L2R est l'ensemble des non réconciliations certaines, que la plupart des méthodes existantes ne permettent pas d'obtenir. Ces non réconciliations sont très utiles pour la réduction de la taille de l'espace de réconciliation. En effet, ces non réconciliations peuvent être exploitées lors du filtrage des paires de références à traiter par une méthode numérique de réconciliation de références. Un troisième apport de cette méthode est l'inférence des synonymies certaines pouvant permettre de constituer un dictionnaire de synonymes enrichi au fur et à mesure de l'application de la méthode L2R. Ce dictionnaire représente en quelque sorte les connaissances acquises sur les variations syntaxiques d'un domaine d'application et peut être exploité par la méthode L2R elle-même ou par une méthode numérique de réconciliation telle que N2R. Par cette méthode L2R nous avons donc pu montrer l'intérêt d'une approche logique pour résoudre le problème de réconciliation de références qui est un problème très courant dans le domaine de l'intégration de données.

La seconde méthode est une méthode numérique, nommée N2R, qui exploite également les connaissances du schéma dans un calcul informé de similarité des paires de références. Les axiomes de fonctionnalité et de fonctionnalité inverse des propriétés sont exploités pour définir les impacts des différents attributs et relations impliqués dans le calcul de similarité d'une paire de références. De plus, les axiomes de disjonction de classes et d'UNA sur les données sont exploités dans une phase de pré-traitement où l'on élimine de l'espace de réconciliation toutes les paires de références qui sont a priori non réconciliables.

Une des originalités de notre méthode numérique de réconciliation est l'expression du calcul numérique de similarité des paires de références dans un système d'équations non linéaire. En effet, par le biais des équations, nous avons pu exprimer les dépendances des similarités entre paires de références, permettant ainsi de propager leur score de similarité selon les impacts des relations définis grâce aux axiomes du schéma RDFS+. L'expression du calcul de similarité dans un système d'équations nous a permis d'adapter la méthode de Jacobi [GL89] de résolution itérative de systèmes d'équations linéaire pour résoudre notre système d'équations. De plus, nous avons prouvé la convergence de cette méthode pour la résolution de notre système d'équations.

La méthode N2R peut être appliquée seule ou précédée de l'application de la méthode logique L2R. En effet, les décisions de (non) réconciliation et de (non) synonymie peuvent être exploitées par la méthode N2R, soit à la phase de filtrage, soit à l'initialisation du système d'équations.

Nous avons expérimenté la méthode N2R sur les mêmes jeux de données que ceux utilisés dans l'expérimentation de L2R. Nos résultats ont montré par cette expérimentation que les résultats obtenus par N2R sont cohérents avec ceux obtenus par L2R. Les résultats obtenus ont pu montrer qu'avec une méthode complètement automatique et non supervisée nous obtenons un bon rappel sans faire chuter la précision.

Dans cette thèse nous avons proposé deux méthodes de réconciliation de références qui peuvent être appliquées indépendamment l'une de l'autre ou en combinaison. Lors de l'intégration de données si l'on souhaite des résultats certains on applique alors uniquement la méthode L2R. En revanche, dans le cas où on souhaite obtenir le plus de décisions de reconciliation possibles, on applique alors les deux méthodes : la méthode L2R suivie de la méthode N2R.

7.2 Travaux en cours et perspectives

Dans cette section nous présentons, tout d'abord, les limites des solutions que nous avons proposées puis les travaux en cours ainsi que les perspectives que nous envisageons de réaliser.

Autour de l'approche d'enrichissement sémantique.

Si notre méthode d'enrichissement sémantique possède de nombreux avantages, elle présente aussi des limites. La première limite est liée au choix de mesures de similarité afin qu'elle soient mieux adaptées aux caractéristiques des données traitées. La seconde limite concerne l'exploitation des valeurs numériques qui sont très fréquentes dans les tableaux. En effet, pour mieux identifier les colonnes du tableau, il est possible d'améliorer l'exploitation des valeurs numériques en s'aidant, par exemple, d'unités de mesure, si elles sont définies dans l'ontologie, pouvant être associées à certains termes de l'ontologie. Un autre aspect important qu'il est également possible d'améliorer est la définition d'un ordre de pertinence des différentes relations identifiées et des différents termes associés aux valeurs du tableau pouvant être exploité lors de l'interrogation. Enfin, il serait intéressant d'étendre notre méthode d'enrichissement sémantique pour traiter des données semi-structurées non tabulaires et donc permettre de traiter l'ensemble du document.

Notre travail a abouti à l'identification de plusieurs améliorations et extensions possibles. Nous commençons tout d'abord par présenter les extensions en cours de réalisation dans le cadre de deux thèses. Puis nous donnons les perspectives que nous envisageons d'étudier et de mettre en œuvre.

- ⋆ **Annotation sémantique floue de tableaux guidée par une ontologie.** Une des limites de notre méthode d'enrichissement sémantique que nous avons identifiée est l'absence d'ordre de pertinence des différentes relations sémantiques et des termes de l'ontologie identifiés dans les données. Dans la thèse de *Gaëlle Hignette*, se déroulant actuellement au sein de l'équipe BIA-INRA, ce problème a été partiellement traité dans [HBDBH07] en représentant l'ensemble des termes associés à une valeur par un ensemble flou [Zad65]. Ces ensembles flous permettent de décrire la similarité d'une valeur du tableau avec les termes de l'ontologie. Dans ce même travail [HBDBH07] une méthode, permettant de différencier entre des colonnes symboliques et des colonnes numériques, a été proposée. En s'aidant d'une ontologie plus riche dans laquelle certains termes sont décrits par des unités de mesure (par exemple, °C ou °F pour la température) ou par un intervalle de valeurs (par

exemple la valeur de pH est comprise entre 0 et 14) il est possible d'améliorer l'identification des colonnes numériques.

⋆ **Enrichissement sémantique de documents semi-structurés**. Notre méthode d'enrichissement sémantique peut être utilisée par une méthode d'enrichissement sémantique plus générale. Dans la thèse de *Mouhamadou Thiam*, se déroulant actuellement au sein de l'équipe IASI/Gemo du LRI, l'intérêt est porté sur l'étude d'une méthode automatique d'enrichissement sémantique de données semi-structurées [TBP07], telles que les pages XHTML ou XML, en s'appuyant sur une description du domaine exprimée dans le langage RDFS étendu par des règles. Ainsi, il est possible d'interroger une collection de documents enrichis sémantiquement par des requêtes exprimées dans la description du domaine. Dans une telle méthode, il serait possible d'utiliser notre méthode XTab2SML pour enrichir sémantiquement les parties du document XHTML ou XML représentant des tableaux.

⋆ **Méthode d'enrichissement de l'ontologie du domaine**.
Dans cette thèse, nous avons étudié comment exploiter les informations imparfaitement identifiées pour découvrir des relations sémantiques candidates à l'enrichissement de l'ontologie où à son extension (cf. Section 2.4). Cependant, il reste à définir une méthode permettant de mettre en oeuvre les idées que nous avons présentées mais aussi d'approfondir cette étude. De plus, il est possible d'exploiter le résultat de l'enrichissement sémantique pour découvrir des termes candidats à l'enrichissement de l'ontologie. En effet, il est intéressant de s'appuyer sur les parties les plus structurées des documents semi-structurés pour découvrir de nouveaux termes pouvant être utilisés dans l'enrichissement sémantique des parties moins structurées du document.

⋆ **Réécriture des requêtes utilisateur en requêtes plus flexibles**. Un des aspects importants du format SML est sa flexibilité. Afin de tirer profit de cette flexibilité et de permettre au moteur d'interrogation de s'adapter, les requêtes de l'utilisateur exprimées en termes de l'ontologie du domaine doivent être réécrites pour pouvoir interroger des informations partiellement identifiées, des informations non identifiées mais également des informations contextuelles.

⋆ **Projet WebContent [web06]**. Il s'agit d'un projet national labelisé RNTL qui a débuté en 2006 et qui réunit une dizaine de partenaires, des industriels et des équipes de recherche. Ce projet a pour vocation de concevoir une plate-forme logicielle générique combinant des outils d'extraction et de gestion du contenu du Web. Dans le cadre de ce projet, notre travail sur l'enrichissement sémantique qui a été amélioré et complété par les contributions apportées dans [HBDBH07] va être intégré. La généricité d'une telle approche pourrait être évaluée puisque il s'agit d'utiliser la méthode d'enrichissement sémantique de tableaux traitant d'autres domaines tel que l'aéronautique.

Autour de l'approche de réconciliation de références.

Si notre approche de réconciliation de références possède de nombreux avantages elle présente aussi des limites. La première limite concerne la méthode numérique, elle est liée aux pondérations *identiques* que nous calculons pour les relations et pour les attributs qui ne sont pas de type *dépendance fonctionnelle*. Il est possible d'affiner ces pondérations en exploitant des

connaissances supplémentaires telles que la stabilité dans le temps des références et des valeurs associées aux relations et aux attributs. Par exemple le prix d'un séjour est une valeur instable dans le temps, elle devrait avoir une pondération plus faible que celle de l'attribut destination du séjour. La seconde limitation concerne les deux méthodes et elle est liée à leur applicabilité dans un système pair-à-pair où les schémas RDFS+ peuvent être différents et distribués. Un autre aspect, qu'il serait précieux d'assurer, est le passage à l'échelle de notre approche de réconciliation de références. Enfin, il serait également intéressant d'étendre nos deux méthodes de réconciliation de références à des méthodes capables de réconcilier des références présentées dans des documents semi-structurés (XML) avec des schémas non réconciliés.

- ⋆ **Affinement des pondérations des relations et des attributs**. Pour garantir l'aspect non supervisé de notre méthode numérique nous avons défini une pondération identique pour les attributs et les relations qui ne sont pas de type dépendance fonctionnelle, calculées sur la description commune des paires de références. Toutefois, ce choix est discutable car tous les attributs et toutes les relations ne représentent pas le même type d'informations et donc leur importance diffère. Il est possible pour la méthode N2R d'utiliser un échantillon de données étiqueté automatiquement par les résultats certains de L2R. Ainsi, une méthode d'apprentissage automatique peut être mise en œuvre afin d'apprendre ces différentes pondérations en exploitant par exemple la fréquence et la similarité des valeurs.

- ⋆ **Adapter les méthodes L2R et N2R pour une utilisation dans un contexte pair-à-pair**.
  Afin d'assurer le passage à l'échelle du Web des méthodes de réconciliation de références, on peut envisager leur application dans un contexte pair-à-pair. En d'autres termes, en plus de la distribution et de l'autonomie des sources de données, les schémas RDFS+ sont également autonomes et distribués. Certaines approches travaillent dans ce cadre et estiment que l'utilisation de mappings entre ontologies locales des pairs interconnectés est plus réaliste que la création d'une grande ontologie commune. Un système tel que *SomeWhere* [ACG+06] est une approche complètement pair à pair dans laquelle chaque pair stocke localement ses propres axiomes et un ensemble de mappings vers les ontologies des autres pairs. Il exploite les mappings pour répondre de façon correcte et complète à une requête. Une telle approche peut être complétée par une méthode permettant de découvrir et d'exploiter des réconciliations de références afin d'intégrer les données provenant de différents pairs lors de l'évaluation d'une requête. Dans [KAM03] les auteurs proposent une définition des tables de réconciliation dans lesquelles sont stockées les références réconciliées entre les différents pairs.

- ⋆ **Généraliser L2R pour raisonner sur des connaissances probables**. Dans le but de tirer profit au maximum de l'ensemble de règles traduisant les axiomes du schéma et des données, il serait intéressant d'étendre la méthode L2R. Au lieu de considérer des faits de synonymie de départ obtenus uniquement pour des valeurs syntaxiquement identiques on pourrait envisager de considérer les faits de synonymie avec leur score de similarité même si les valeurs ne sont pas identiques. En effet, il est possible de modéliser le problème de réconciliation de références dans un réseau Bayésien et d'utiliser ensuite un système d'inférence Bayésien pour décider de la réconciliation ou de la non réconciliation entre références. Pour chaque paire de références concernées par les connaissances du schéma pour laquelle nous cherchons à décider de sa (non) réconciliation, nous inférons une probabilité de réconciliation. Une première solution consisterait à considérer comme probabilités a priori, les

scores de similarité entre les valeurs de base, scores dont les valeurs sont comprises entre 0 et 1. La structure du réseau Bayesien peut être construite à partir des instances des règles traduisant les axiomes du schéma et des données.

⋆ **Projet PICSEL3 [pic05]**. PICSEL3 [pic05] est un projet de recherche en collaboration entre l'équipe IASI/Gemo du LRI et France Télécom R&D qui a débuté en 2005. L'objectif est de concevoir un système d'intégration mixte de données pouvant jouer le rôle de médiateur entre des sources distantes et un entrepôt local de données. Le schéma de l'entrepôt local joue également le rôle du schéma médiateur – ontologie du domaine d'application. L'interrogation de données dans le système PICSEL3 combine l'interrogation de l'entrepôt local et celle des sources distantes décrites par un ensemble de vues. Dans ce projet, les solutions que nous avons proposées pour la réconciliation de références sont en cours d'étude afin de les intégrer dans des produits logiciels dont la description ne peut pas, à présent, être rendue publique.

⋆ **Réconciliation de références représentées dans des documents semi-structurés**. La réconciliation de références contenues dans des documents XML dont les schémas sont hétérogènes rajoute des problèmes supplémentaires par rapport à ceux que nous avons considérés jusque là dans cette thèse : (i) l'hétérogénéité structurelle des données. En effet, la flexibilité de XML permet de définir des éléments de différentes manières avec des éléments et des attributs optionnels. Il s'agit d'une difficulté supplémentaire car, avant de procéder à la réconciliation il est d'abord nécessaire de définir la description commune en termes d'attributs et de relations à comparer tout en considérant l'hétérogénéité des schémas ; (ii) l'identification des frontières des descriptions à considérer lors de la réconciliation des paires de références, en évitant de considérer l'ensemble du document comme description de chaque références. Cette perspective nous permettra de montrer comment la réconciliation des schémas peut aider à améliorer la réconciliation de référence et de manière analogue comment la réconciliation de références peut à son tour améliorer la réconciliation de schémas.

Table des figures

1 *Architecture d'un système médiateur* . . . viii
2 *Architecture d'un entrepôt de données* . . . ix

3 Extrait de la taxonomie de termes de l'ontologie du risque alimentaire Sym'Previus 7
4 *Exemple de tableaux de données à intégrer* . . . 8

1.1 Une classification des mesures de similarité pour l'alignement . . . 11
1.2 Extrait du schéma relationnel de l'ontologie du risque alimentaire Sym'Previus . 23
1.3 Une architecture fonctionnelle du projet e.dot . . . 24
1.4 Le module Any2XTab . . . 26
1.5 *Représentation XTab du tableau 1.7* . . . 26

2.1 *Une représentation simplifiée en SML du tableau XTab de la Figure 1.5.* . . . 30
2.2 *Représentation SML de relations complètement identifiées* . . . 35
2.3 *Représentation SML de relation partiellement identifiée avec des attributs Null* . . 36
2.4 *Représentation SML d'une relation partiellement identifiée avec un attribut représenté par une constante* . . . 37
2.5 *Représentation d'une relation générique en SML* . . . 38
2.6 *La représentation SML d'une relation avec prise en compte de listes de valeurs* . 40
2.7 *Liste de valeurs avec segmentation* . . . 40
2.8 *Liste de valeurs sans segmentation* . . . 40
2.9 *Représentation SML de la composition nutritionnelle des aliments* . . . 41
2.10 Extrait du schéma relationnel de l'ontologie du risque alimentaire (Sym'Previus) 47
2.11 Extrait de la DTD SML (risque alimentaire) . . . 48
2.12 *Microorganismes et aliments fumés* . . . 49
2.13 *Aliments et Microorganismes* . . . 49
2.14 *Représentation SML du tableau de la fig.2.13 - Attribut supplémentaire* . . . 50
2.15 *Doses infectieuses* . . . 51
2.16 *Représentation SML du tableau de la fig.2.15 - Relation générique* . . . 51
2.17 *Une présentation possible de la réponse* . . . 52
2.18 *La réponse à la requête Q1* . . . 52
2.19 *Réponse à la requête Q2* . . . 53
2.20 Une représentation des trois vues de MIEL++ . . . 53

3.1 Schéma représentant les entrées et la sortie du service Web XTab2SML . . . 58
3.2 *Les résultats obtenus par l'expert après la vérification des relations sémantiques.* . 62
3.3 Rappel, Précision et F-Mesure pour un seuil $th = 0,3$ . . . 63
3.4 *Résultat de l'évaluation de l'identification des attributs* . . . 64

5 *Exemple de descriptions de données à réconcilier* 70

4.1 Smith et Waterman sur les deux chaînes σ_1="Bd de Paris" et σ_2="Boulevard de Paris" . 77
4.2 *Exemple de relations à intégrer* 80
4.3 Les régions du modèle de Fellegi et Sunter 85
4.4 *Les coûts correspondant aux différentes décisions* 87
4.5 *Exemple de listes ordonnées d'une relation R par rapport à une requête Q donnée.* 89
4.6 Exemple de trois relations formant une hiérarchie relationnelle. 91
4.7 *Les étapes de l'algorithme Delphi* 93
4.8 *Exemple de tables sans redondances et avec redondances* 94
4.9 *Deux relations à intégrer* 105
4.10 Exemple d'arbre de décision pour le domaine des Musées. 106
4.11 *Intelliclean : un exemple de règle de réconciliation* 108

5.1 Exemple de graphe RDF 118
5.2 Un extrait du schéma RDF des données de la Figure 5.1 119
5.3 The Layer Cake proposé par Tim Berners-Lee [BL04] 121
5.4 Exemple de schéma RDFS 123
5.5 *Exemple de données RDF* 126

6.1 *L2R – Exemple de données RDF du domaine des lieux culturels.* 138
6.2 Schéma RDFS des données FT_HOTEL 146
6.3 Schéma RDFS des données de Cora 147

7.1 *N2R – Exemple de données RDF du domaine des lieux culturels.* 156
7.2 Résultats obtenus par N2R sur *Cora* 184
7.3 Résultats obtenus par N2R sur *FT_HOTELS* 185

B.1 Un extrait de l'ontologie du domaine du risque alimentaire(Sym'Previus) 206

C.1 Un exemple de document XTab 207
C.2 La DTD des documents XTab 208

D.1 La DTD complète des documents SML traitant du domaine du risque alimentaire 210

E.1 La requête XQuery *Aliment/effet du facteur analysé* 211
E.2 la requête XQuery *Charge microbienne* 212

F.1 La description WSDL du service Web XTab2SML 214

Liste des tableaux

1.1 *Schéma S : Houses* . . . 12
1.2 *Schéma S : Agents* . . . 12
1.3 *Schéma T : Listings* . . . 12
1.4 *Le contexte formel (G,M1,I1)* . . . 16
1.5 *Le contexte formel (G,M2,I2)* . . . 17
1.6 *Exemple de table* . . . 18
1.7 *approximative pH of some food products* . . . 26

2.1 *Nutritional Composition of some food products* . . . 33
2.2 *Doubling times of Listeria monocytogenes in foodstuffs* . . . 36
2.3 *approximative pH of some food products* . . . 39

3.1 Exemple de tableau où la relation reconnue est incorrecte . . . 60
3.2 Exemple de tableau où la constante reconnue est incorrecte . . . 61
3.3 Exemple de tableau : pertinence des attributs supplémentaires . . . 61
3.4 Résultat de l'évaluation de la découverte de relations sémantiques . . . 65
3.5 Intérêt des relations génériques . . . 65

4.1 Tableau comparatif des différentes méthodes de réconciliation de références Description/Connaissances . . . 116

5.1 Sémantique des constructeurs RDFS en logique du premier ordre . . . 119

6.1 *Résultats de L2R sur les données FT_HOTELS* . . . 148
6.2 *Résultats de L2R sur les données Cora (classes Article et Conference)* . . . 149

7.1 *N2R-Exemple de description commune à des références* . . . 160
7.2 *Illustration du calcul – Exemple de description commune à des références* . . . 173
7.3 *Illustration du calcul – itération du calcul de similarité.* . . . 175

Annexe A

Preuve de convergence du calcul de similarité

Pour prouver la convergence du calcul de similarité, nous devons prouver que les différentes propriétés suivantes sont vérifiées :

1. $\forall i$ et $\forall k$ la valeur de x_i^k calculée par le système d'équations est bornée supérieurement par 1 : $\forall\ i, \forall k,\ x_i^k\ \leq\ 1$
2. $\forall$ i, la fonction f_i est croissante :

$$\forall\ (x_1, x_2, \ldots, x_n)\ \in\ R^n\ et\ \forall\ (y_1, y_2, \ldots, y_n)\ \in\ R^n,$$

$$(x_1 \leq y_1), (x_2 \leq y_2), \ldots, (x_n \leq y_n)\ \Longrightarrow\ f_i(x_1, x_2, \ldots, x_n)\ \leq\ f_i(y_1, y_2, \ldots, y_n)$$

3. après K itérations le calcul de similarité admet un point-fixe. Si l'on montre que pour tout i, (x_i^k) converge vers une limite $l_i \leq 1$, on a alors par définition :

$$\forall i, \exists K_i,\ \forall k,\ tq\ k \geq K_i,\ |x_i^{k+1} - x_i^k| \leq \epsilon$$

Si l'on choisit $K = \max\{K_1, \ldots, K_n\}$, on a alors :

$$\forall k,\ tq\ k \geq K,\ \forall i,\ |x_i^{k+1} - x_i^k| \leq \epsilon$$

Afin de rendre plus compacte l'expression de notre calcul de similarité, nous déclarons la constante c_i exprimant la moyenne pondérée des scores de similarité des paires et des ensembles de valeurs de base qui est définie comme suit :

$$c_i = \sum_{l=0}^{l=|NDF_A(<ref,ref'>)|} (\lambda_{il} *\ b_{il-ndf}) + \sum_{l=0}^{l=|NDF_A*(<ref,ref'>)|} (\lambda_{il} *\ BS_{il-ndf})$$

Tout d'abord posons les hypothèses suivantes :

⋆ Les scores de similarité concernant les paires de valeurs de base sont calculés par des mesures de similarité normalisées. En d'autres termes, $b_{il-ndf} \in [0..1]$, $b_{il-df} \in [0..1]$ et $b_{il-dfm} \in [0..1]$ .....(H1)

⋆ Les pondérations associées aux différents attributs et aux différentes relations considérés dans le calcul de la moyenne sont fixées ou calculées de manière à garantir que le moyenne pondérée de scores de similarité **normalisés** envoie une valeur **normalisée**, ce qui est exprimé dans l'inégalité suivante :

$$\forall\ l,\ \lambda_{il}\ \geq 0\ et\ \forall\ j,\ \lambda_{ij}\ \geq 0\ et$$

$$\Big(\sum_{l=0}^{l=|NDF_A(<ref,ref'>|}\lambda_{il}+\sum_{l=0}^{l=|NDF_A*(<ref,ref'>|}\lambda_{il}+\sum_{j=0}^{j=|NDF_R(<ref,ref'>|}\lambda_{ij}+\sum_{j=0}^{j=|NDF_R*(<ref,ref'>|}\lambda_{ij}\Big) \leq 1$$

$$.....(H2)$$

⋆ nous admettons que la fonction $SSim_r$ de calcul de similarité entre des ensembles de références renvoie des scores de similarité normalisés entre ensembles de références ..... (H3.1)
et que la fonction $SSim_r$ est une fonction **croissante** .....(H3.2).

A.1 Scores de similarité des paires de références bornés par 1

Proposition 1 : les valeurs des x_i calculées par le système, à toute itération k, sont bornées par la valeur 1,

$$\forall k,\ \forall i,\ \ x_i^k\ \leq\ 1\ \ .....\ (P1)$$

Preuve. Nous allons procéder par récurrence sur k.
Soit k=0. A l'initialisation le vecteur X^0 du système d'équations est initialisé à 0, en posant $\forall\ i,\ x_i^0 = 0$, alors $\forall\ i,\ x_i^0 \leq 1$. Par conséquent, pour tout i, x_i^0 est borné par 1.

Supposons que pour k=p > 0, pour tout i, $x_i^p \leq 1$. Maintenant, il faut prouver que cette affirmation reste vraie pour k=p+1. Pour cela, il suffit de montrer pour un i arbitraire que $x_i^{p+1} \leq 1$.

$$f_i(X) = \max\Big(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|}(b_{il-df}), moy\Big(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm})\Big),$$

$$\bigcup_{j=0}^{j=|DF_R(<ref,ref'>)|}(x_{ij-df}^p), moy\Big(\bigcup_{j=0}^{j=|DF_R^M(<ref,ref'>)|}(x_{ij-dfm}^p)\Big)\Big)$$

$$\Big(c_i + \sum_{j=0}^{j=|NDF_R(<ref,ref'>)|}(\lambda_{ij} *\ x_{ij-ndf}^p) + \sum_{j=0}^{j=|NDF_R*(<ref,ref'>)|}(\lambda_{ij} *\ XS_{ij-ndf}^p)\Big)\Big)$$

Pour cela, nous devons montrer que,

1. la valeur calculée par la fonction $f_{i-df}(X)$ est bornée supérieurement par 1 pour $k = p+1$:

$$\max(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|}(b_{il-df}), moy(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm})),$$
$$\bigcup_{j=0}^{j=|DF_R(<ref,ref'>)|}(x^p_{ij-df}), moy(\bigcup_{j=0}^{j=|DF_R^M(<ref,ref'>)|}(x^p_{ij-dfm})))$$

Par hypothèse (H1) $b_{il-dfm} \leq 1$, par hypothèse d'induction $x^p_{j-dfm} \leq 1$ et nous avons par définition la fonction *moy* qui est monotone nous avons donc les inégalités suivantes qui sont vérifiées :

$$moy(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm})) \leq 1$$

$$moy(\bigcup_{j=0}^{j=|DF_R^M(<ref,ref'>)|}(x^p_{ij-dfm})) \leq 1$$

Par l'hypothèse (H1) $b_{il-df} \leq 1$ et par l'hypothèse d'induction $x^p_{j-df} \leq 1$ et par le fait que le maximum d'un ensemble de valeurs bornées par 1 est aussi borné par 1, nous avons prouvé l'inéquation suivante :

$$\max\Big(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|}(b_{il-df}), moy(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm})),$$
$$\bigcup_{j=0}^{j=|DF_R(<ref,ref'>)|}(x^p_{ij-df}), moy(\bigcup_{j=0}^{j=|DF_R^M(<ref,ref'>)|}(x^p_{ij-dfm}))\Big) \leq 1 \tag{A.1}$$

2. la valeur calculée par la fonction $f_{i-ndf}(X)$ est bornée supérieurement par 1 pour $k = p+1$

$$(c_i + \sum_{j=0}^{j=|NDF_R(<ref,ref'>)|}(\lambda_{ij} * x^p_{ij-ndf}) + \sum_{j=0}^{j=|NDF_{R*}(<ref,ref'>)|}(\lambda_{ij} * XS^p_{ij-ndf}))$$

Montrons tout d'abord que c_i est aussi bornée par 1.
Par définition

$$c_i = \sum_{l=0}^{l=|NDF_A(<ref,ref'>)|}(\lambda_{il} * b_{il-ndf}) + \sum_{l=0}^{l=|NDF_{A*}(<ref,ref'>)|}(\lambda_{il} * BS_{il-ndf})$$

D'après l'hypothèse (H1), nous avons $\forall j,\ b_{ij-ndf} \leq 1$.
Nous avons par définition :

$$BS_{il-ndf} = SSim_v(S1, S2, \theta) = \frac{|CLOSE_v(S1, S2, \theta)|}{\max(|S1| + |S2|)},$$

avec S1 et S2 deux ensembles de valeurs de base. Nous avons également par définition $|CLOSE_v(S1, S2, \theta)| \leq \max(|S1| + |S2|)$. Il est évident d'affirmer que : $BS_{il-ndf} \leq 1$
D'après l'hypothèse de normalité des valeurs de λ_{il} et de λ_{ij} (H2), nous obtenons alors l'inégalité suivante :

$$\left(\sum_{l=0}^{l=|NDF_A(<ref,ref'>)|} (\lambda_{il} *\ b_{il-ndf}) + \sum_{l=0}^{l=|NDF_A*(<ref,ref'>)|} (\lambda_{il} *\ BS_{il-ndf}) \right) \leq 1 \tag{A.2}$$

Nous déduisons donc que $c_i \leq 1$.

D'après l'inéquation 2 nous avons $c_i \leq 1$, et d'après l'hypothèse de récurrence nous avons $x^p_{j-ndf} \leq 1$. Nous rappelons que la valeur de XS^p_{ij} est calculée par la fonction $SSim_r$. Nous avons fait l'hypothèse (H3.1) que la fonction $SSim_r$ renvoie des valeurs de similarité normalisées , $XS^p_{ij} \in [0..1]$, donc $XS^p_{ij} \leq 1$.
D'après l'hypothèse de normalité des valeurs de λ_{il} et de λ_{ij} (H2), nous obtenons alors l'inégalité suivante :

$$\left(\sum_{l=0}^{l=|NDF_A(<ref,ref'>)|} (\lambda_{il} *\ b_{il-ndf}) + \sum_{l=0}^{l=|NDF_A*(<ref,ref'>)|} (\lambda_{il} *\ BS_{il-ndf}) + \right.$$
$$\left. \sum_{j=0}^{j=|NDF_R(<ref,ref'>)|} (\lambda_{ij} *\ x_{ij-ndf}) + \sum_{j=0}^{j=|NDF_R*(<ref,ref'>)|} (\lambda_{ij} *\ XS_{ij-ndf}) \right) \leq 1 \tag{A.3}$$

Par conséquent, d'après 1 et 2 le maximum de deux valeurs bornées par 1 est aussi borné par 1. Nous avons donc montré que $x_i^{p+1} \leq 1$. Alors $\forall k,\ \forall i, x_i^k \leq 1$, nous avons donc pu prouver l'énoncé de la proposition P1 :

$$\forall k,\ \forall i,\ x_i^k \ \leq\ 1$$

A.2 Fonction monotone de calcul de similarité

Proposition 2 : pour tout i, la fonction f_i de calcul de similarité entre références est croissante.

$$\forall\ (x_1, x_2, \ldots, x_n)\ \in\ R^n\ et\ \forall\ (y_1, y_2, \ldots, y_n)\ \in\ R^n, (x_1 \leq y_1), (x_2 \leq y_2), \ldots, (x_n \leq y_n)$$
$$\Longrightarrow\ f_i((x_1, x_2, \ldots, x_n) \leq f_i(y_1, y_2, \ldots, y_n) \quad \ldots\ldots\ldots\ (P2)$$

Preuve. Nous allons prouver cette proposition par construction. On suppose que

$$\forall(x_1, x_2, \ldots, x_n) \in R^n \; et \; \forall \; (y_1, y_2, \ldots, y_n) \in R^n, \; (x_1 \leq y_1), (x_2 \leq y_2), \ldots, (x_n \leq y_n)$$

Soit $X = (x_1, x_2, \ldots, x_n)$ et $Y = (y_1, y_2, \ldots, y_n)$ deux vecteurs de valeurs réelles. Tout d'abord, nous allons décomposer le vecteur X en quatre sous vecteurs X_1, X_2, X_3 et X_4 et le vecteur Y en quatre sous vecteurs Y_1, Y_2, Y_3 et Y_4, tels que :

- les vecteurs X_1 et Y_1 représentent les variables de type *ndf* et correspondant à des paires de références ;
- les vecteurs X_2 et Y_2 représentent les variables de type *ndf* et correspondant à des ensembles de références ;
- les vecteurs X_3 et Y_3 représentent les variables de type dfm ;
- les vecteurs X_4 et Y_4 représentent les variables de type *df* ;

Soient u, v, w, z quatre réels représentant le nombre de composantes des vecteurs définis précédemment, tels que $u = |X_1| = |Y_1|$, $v = |X_2| = |Y_2|$, $w = |X_3| = |Y_3|$, $z = |X_4| = |Y_4|$ et $u + v + w + z = |X| = |Y|$

Posons $X_1 = (x_{11}, x_{12}, \ldots\ldots, x_{1u})$ et $Y_1 = (y_{11}, y_{12}, \ldots\ldots, y_{1u})$, $X_2 = (x_{21}, x_{22}, \ldots\ldots, x_{2v})$ et $Y_2 = (y_{21}, y_{22}, \ldots\ldots, y_{2v})$, $X_3 = (x_{31}, x_{32}, \ldots\ldots x_{3w})$ et $Y_3 = (y_{31}, y_{32}, \ldots\ldots, y_{3w})$ et $X_4 = (x_{41}, x_{42}, \ldots\ldots x_{4w})$ et $Y_4 = (y_{41}, y_{42}, \ldots\ldots, y_{4w})$.

Les inégalités suivantes sont vérifiées par hypothèse :

1. $(x_{11} \leq y_{11}), (x_{12} \leq y_{12}), \ldots, (x_{1u} \leq y_{1u})$;
2. $(x_{21} \leq y_{21}), (x_{22} \leq y_{22}), \ldots, (x_{2v} \leq y_{2v})$;
3. $(x_{31} \leq y_{31}), (x_{32} \leq y_{32}), \ldots, (x_{3w} \leq y_{3w})$
4. $(x_{41} \leq y_{41}), (x_{42} \leq y_{42}), \ldots, (x_{4z} \leq y_{4z})$

La somme des éléments du vecteur X_1, la somme des éléments du vecteur Y_1 et leur multiplication par une constante réelle positive $\lambda_{ij} \in [0..1]$ nous permet d'obtenir l'inégalité suivante :

$$\sum_{j=0}^{j=u}(\lambda_{ij} * x_{1j}) \leq \sum_{j=0}^{j=u}(\lambda_{ij} * y_{1j})$$

De manière analogue nous obtenons l'inégalité suivante :

$$\sum_{j=0}^{j=v}(\lambda_{ij} * x_{2j}) \leq \sum_{j=0}^{j=v}(\lambda_{ij} * y_{2j})$$

L'addition des deux parties des deux dernières inégalités nous permet d'obtenir l'inégalité suivante :

$$\left(\sum_{j=0}^{j=u}(\lambda_{ij} * x_{1j}) + \sum_{j=0}^{j=v}(\lambda_{ij} * x_{2j})\right) \leq \left(\sum_{j=0}^{j=u}(\lambda_{ij} * y_{1j}) + \sum_{j=0}^{j=v}(\lambda_{ij} * y_{2j})\right)$$

L'ajout d'une constante c_i donne l'inégalité suivante :

$$c_i + \Big(\sum_{j=0}^{j=u} (\lambda_{ij} * x_{1j}) + \sum_{j=0}^{j=v} (\lambda_{ij} * x_{2j}) \Big) \leq c_i + \Big(\sum_{j=0}^{j=u} (\lambda_{ij} * y_{1j}) + \sum_{j=0}^{j=v} (\lambda_{ij} * y_{2j}) \Big)$$

(A.4)

L'application de la fonction *moy* qui calcule la moyenne d'un ensemble de valeurs, sur les éléments du vecteur X_3 et sur les éléments du vecteur Y_3 nous obtenons donc :

$$moy(\bigcup_{j=0}^{j=w} (x_{3j})) \ \leq \ moy(\bigcup_{j=0}^{j=w} (y_{3j}))$$

L'application du maximum sur les éléments du vecteur X_4 et sur les éléments du vecteur Y_4 nous obtenons donc :

$$\max(\bigcup_{j=0}^{j=z} (x_{4j})\Big) \ \leq \ \max(\bigcup_{j=0}^{j=z} (y_{4j})\Big)$$

L'application de la fonction maximum sur les deux parties des deux dernières inéquations donne l'inéquation suivante :

$$\max\Big(\bigcup_{j=0}^{j=z} (x_{4j}), moy(\bigcup_{j=0}^{j=w} (x_{3j}))\Big) \ \leq \ \max\Big(\bigcup_{j=0}^{j=z} (y_{4j}), moy(\bigcup_{j=0}^{j=w} (y_{3j}))\Big)$$

Soit B un vecteur de valeurs réelles positives. Si l'on ajoute l'ensemble des éléments de B à celles des vecteurs X_3 et Y_3 alors nous vérifions l'inégalité suivante :

$$\max\Big(B, \bigcup_{j=0}^{j=z} (x_{4j}), moy(\bigcup_{j=0}^{j=w} (x_{3j}))\Big) \ \leq \ \max\Big(B, \bigcup_{j=0}^{j=z} (y_{4j}), moy(\bigcup_{j=0}^{j=w} (y_{3j}))\Big)$$

(A.5)

L'application de la fonction maximum sur les deux inégalités A.2 et A.5 permet d'obtenir l'inéquation suivante :

$$\max\Big(\max\Big(B, \bigcup_{j=0}^{j=z} (x_{4j}), moy(\bigcup_{j=0}^{j=w} (x_{3j}))\Big), c_i + \Big(\sum_{j=0}^{j=u} (\lambda_j * x_{1j}) + \sum_{j=0}^{j=v} (\lambda_j * x_{2j})\Big)\Big) \ \leq$$

$$\max\Big(\max\Big(B, \bigcup_{j=0}^{j=z} (y_{4j}), moy(\bigcup_{j=0}^{j=w} (y_{3j}))\Big), c_i + \Big(\sum_{j=0}^{j=u} (\lambda_j * y_{1j}) + \sum_{j=0}^{j=v} (\lambda_j * y_{2j})\Big)\Big)$$

(A.6)

Nous avons donc pu prouver que pour tout i, la fonction de calcul de similarité est une fonction croissante :

$$\forall i, f_i(x_1^k, x_2^k, \ldots, x_n^k) \leq f_i(x_1^{k+1}, x_2^{k+1}, \ldots, x_n^{k+1})$$

Ce qui correspond à l'énoncé de la proposition P2.

A.3 Convergence vers un point fixe

Propriété 1. Si une suite (u) est croissante et admet un majorant M alors cette suite est convergente et sa limite L est inférieure ou égale à M.

$$\forall \epsilon \geq 0, \exists K,\ tq\ \forall\ k > K,\ alors\ \ \parallel U_k - L \parallel \leq \epsilon$$

Propriété 2. Soit $(u_k)_{k \in N}$ définie de la façon suivante :

$$\begin{cases} u_0 = 0 \\ u_{k+1} = f(u_k), \forall\ k \geq 0 \end{cases}$$

Si f est croissante et $u_1 \geq u_0$, alors (u_n) est croissante. On a alors, $\forall k, f(u_{k+1}) \geq f(u_k)$.

Nous avons montré (c.f. proposition 2) que pour tout i, la fonction f_i de calcul de similarité pour les paires de références est une fonction croissante.

$$\forall\ (x_1, x_2, \ldots, x_n)\ \in\ R^n\ et\ \forall\ (y_1, y_2, \ldots, y_n)\ \in\ R^n, (x_1 \leq y_1), (x_2 \leq y_2), \ldots, (x_n \leq y_n)$$

$$\Longrightarrow\ f_i((x_1, x_2, \ldots, x_n) \leq f_i(x_1, x_2, \ldots, x_n)$$

Maintenant montrons que la propriété 2 est applicable à l'ensemble de nos fonctions f_i. En d'autres termes, pour tout i, pour tout k, $f_i(x_i^k) \geq f_i(x_i^{k-1})$ ce qui est équivalent à : $\forall i, \forall k,\ x_i^{k+1} \geq x_i^k$

Nous allons montrer ceci par récurrence sur k :

⋆ **k=0 :** pour tout i, $x_i^1 \geq 0$.

Nous notons que $\forall i, x_i^0 = 0$ et par définition

$$x_i^1 = \max\left(\max(\bigcup_{l=0}^{l=|DF_A(<ref,ref'>)|}(b_{il-df}), moy(\bigcup_{l=0}^{l=|DF_A^M(<ref,ref'>)|}(b_{il-dfm}))), c_i\right)$$

D'après l'hypothèse H1 nous avons toutes les constantes $b_{il-df} \in [0..1]$, toutes les constantes $b_{il-dfm} \in [0..1]$ et toutes les constantes intervenant dans le calcul de c_i sont positives. Le

maximum de valeurs positives renvoie une valeur positive. Nous affirmons donc que la valeur de x_i^1 est positive. Nous en déduisons alors que $\forall i, x_i^1 \geq 0$, d'où la conclusion que lorsque $k = 0$, pour tout i, $x_i^1 \geq x_i^0$. Donc la fonction fi est croissante pour $k = 0$.

⋆ **Hypothèse de récurrence :** supposons que pour tout i et pour tout $k = p$ nous avons $x_i^{p+1} \geq x_i^p$. Maintenant, montrons le pour $k = p + 1$.

⋆ **k=p+1 :** montrer que $x_i^{p+2} \geq x_i^{p+1}$. Cela revient à montrer que pour tout i, $f_i(x_1^{p+1}, \ldots, x_n^{p+1}) \geq f_i(x_1^p, \ldots, x_n^p)$.

D'après l'hypothèse de récurrence nous avons pour tout i, $x_i^{p+1} \geq x_i^p$ ce qui nous permet d'avoir également les inégalités suivantes : $x_1^p \leq x_1^{p+1}, x_2^p \leq x_2^{p+1}, \ldots, x_n^p \leq x_n^{p+1}$.
Par application de la proposition 2, nous obtenons :
$f_i(x_1^p, \ldots, x_n^p) \leq f_i(x_1^{p+1}, \ldots, x_n^{p+1})$, ce qui correspond à $x_i^{p+1} \leq x_i^{p+2}$. Nous avons donc montré que pour pour tout i, pour tout k, $x_i^{k+1} \geq x_i^k$.

Ainsi, nous pouvons appliquer la propriété 2 sur les fonctions f_i pour toute valeur de $i \in [0..n]$, avec n nombre de paires de références considérées.
Nous avons pour tout i, la fonction $f_i(X)$ de calcul de score de similarité qui est croissante et majorée alors nous déduisons qu'elle admet un point-fixe, ce qui s'exprime par : pour tout i, (x_i^k) converge vers une limite $l_i \leq 1$:

$$\forall i \ \exists K_i, \ tq \ \ \forall k, \ si \ \ k \geq K_i, \ alors \quad |x_i^{k+1} - x_i^k| \leq \epsilon$$

Soit $K = \max\{K_1, \ldots, K_n\}$, alors :

$$\forall k, \ tq \ k \geq K, \ \forall i, \ |x_i^{k+1} - x_i^k| \leq \epsilon$$

Annexe B

L'ontologie Sym'Previus

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE EDot:Ontologie SYSTEM "OntologyXMLOutput.dtd">
<!--Ici commencent les donné es XML --> <ontologie>
    <taxonomie>
        <specialisation>
            <terme> Top </terme>
            <sous-terme>
                <terme> Food </terme>
                <terme> Microorganism </terme>
                <terme> Factor </terme>
            </sous-terme>
        </specialisation>
        <specialisation>
            <terme> Food </terme>
                <sous-terme>
                    <terme> Food products </terme>
                    <terme> Environment and others </terme>
                    <terme> Culture medium </terme>
                    <terme> New products to be included </terme>
                </sous-terme>
        </specialisation>
        <specialisation>
            <terme> Food products </terme>
                <sous-terme>
                    <terme> Drinks </terme>
                    <terme> Animal products </terme>
                    <terme> Vegetable products </terme>
                    <terme> Pre-cooked dishes </terme>
                </sous-terme>
        </specialisation>
        ...
        <specialisation>
                    <terme> Animal products </terme>
                        <sous-terme>
                            <terme> Milk and dairy porducts </terme>
                            <terme> Egg and egg products </terme>
                            <terme> Sea food and fresh water products </terme>
                            <terme> Meat and meat products </terme>
                        </sous-terme>
                </specialisation>
        ...
        </taxonomie>
        <schema-rel>
                    <!-- Les relations de la vue MIEL++ charge microbienne  -->
                    <signature>
                        <relation>foodPH</relation>
                        <attribut>Food</attribut>
                        <attribut>pH</attribut>
                    </signature>
                    <signature>
                        <relation>foodMicroorganismConcentration</relation>
                        <attribut>Food</attribut>
                        <attribut>Microorganism</attribut>
                        <attribut>Concentration</attribut>
            </signature>
            ...
        </schema-rel>
</ontologie>
```

FIG. B.1 – Un extrait de l'ontologie du domaine du risque alimentaire(Sym'Previus)

Annexe C

Document XTab

C.1 Exemple de document XTab

```
<?xml version="1.0" encoding="UTF-8"?> <table>
    <id_xtab>844</id_xtab>
    <id>1402</id>
    <title>
        <table-title>Meat and poultry pathogens of concern and their
                     water activity growth limits.</table-title>
        <column-title>Microorganism </column-title>
        <column-title>Water activity (aw) growth limit value </column-title>
    </title>
    <nb-col>2</nb-col>
    <content>
        <line>
            <cell>Escherichia coli </cell>
            <cell>0.95 </cell>
        </line>
        <line>
            <cell>Salmonella  spp. </cell>
            <cell>0.95 </cell>
        </line>
        <line>
            <cell>Listeria  monoctyogenes </cell>
            <cell>0.92 </cell>
        </line>
        <line>
            <cell>Stahphylococcus aureus  </cell>
            <cell>0.86 </cell>
        </line>
    </content>
</table>
```

FIG. C.1 – Un exemple de document XTab

C.2 La DTD XTab

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT table (title, nb-col, content)>
<!ELEMENT title (table-title?, column-title+)>
<!ELEMENT nb-col ANY>
<!ELEMENT content (line+)>
<!ELEMENT table-title ANY>
<!ELEMENT column-title ANY>
<!ELEMENT line (cell+)> <!ELEMENT cell ANY>
```

FIG. C.2 – La DTD des documents XTab

Annexe D

La DTD SML (Semantic Markup Language)

```
<!ELEMENT table (id_xtab, id, title, nb-col, content)> <!ELEMENT
id_xtab ANY> <!ELEMENT id ANY> <!ELEMENT title (table-title?,
column-title+)> <!ELEMENT nb-col ANY> <!ELEMENT table-title ANY>
<!ELEMENT column-title ANY> <!ELEMENT content (rowRel+)>

<!-- Déclaration de lensemble des relations sémantiques pouvant
être instanciéees dans un document SML -->

<!ELEMENT rowRel ( foodph*, foodaw*,foodstate*,foodstructure*,
foodmicroorganismconcentration*, foodfactormicroorganism*,
foodfactor*, foodmicroorganism*, foodcalorie*, foodprotein*,
foodglucid*, foodlipid*, foodacid*,foodvitamine*,
foodamountcalorie*, foodamountprotein*, foodamountglucid*,
foodamountlipid*, foodamountacid*,  foodcarbon*, foodoxygen*,
foodazot*, foodstate*, relation?)>

<!ATTLIST rowRel additionalAttr (yes | no) #REQUIRED> <!ELEMENT
foodph (foodList,phList,attribute*)> <!ELEMENT foodaw
(foodList,awList,attribute*)> ... <!ELEMENT
foodmicroorganismconcentration (foodList,microorganismList,
concentrationList,attribute*)> <!ELEMENT foodfactormicroorganism
(foodList, factorList, microorganismList, attribute*)> <!ELEMENT
foodfactor (foodList, factorList, attribute*)> <!ELEMENT
foodmicroorganism (foodList,microorganismList, attribute*)>
<!ELEMENT relation (attribute*)>

<!-- Déclaration des attributes XML des tags des relations
sémantiques -->

<!ATTLIST foodph  relationType (total | partialNull | partialConst |
partialConstNull | generic) #REQUIRED>  ...

 <!-- Déclaration des attributes des tags des relations sémantiques-->

<!ELEMENT foodList (food+)> <!ELEMENT food (ontoVal*, originalVal)>
... <!-- Déclaration des tags  XML ontoVal et originalVal  -->

<!ELEMENT ontoVal (#PCDATA)> <!ELEMENT originalVal (#PCDATA)>
<!ATTLIST ontoVal   indMap CDATA #IMPLIED >

<!-- Déclaration des attributes XML des attributes des relations
sémantiques -->

<!ATTLIST attribute attrMatch CDATA #REQUIRED attrType (generic) >
<!ATTLIST food attrMatch CDATA #REQUIRED attrType (Const | Null |
Normal) #IMPLIED>  ...
```

FIG. D.1 – La DTD complète des documents SML traitant du domaine du risque alimentaire

Annexe E

Requêtes d'interrogation de MIEL++

E.1 Requête sur la vue *Aliment/effet du facteur analysé*

L'utilisateur saisit : *food* **: cabbage, ph : 5,** *factor* **: Temperature.**
Cette requête est exprimée en langage XQuery de la manière suivante :

```
Let $food:="cabbage"
Let $microorganism :="Listeria M."
Let $ph:=5
For $content IN table
    Return
    <table>
    <title>$content/ancestor()//table-title</title>
    {
    For $line IN $content/rowRel
        Let $foodData := $line/food/MicroorganismConcetration/food/ontoVal
        Let $microorganismData := $line/food/MicroorganismConcetration/microorganism/ontoVal
        Let $phData :=$line/foodPh/ph/ontoVal
        Where(min,  ad($food, foodData),
                ad($microorganism, $microorganismData),
                ad($ph, $phData))>0)

            Return
            <line>
                $line
                <ad>
                min(ad($food, foodData),ad($ph, $phData))
                </ad>
            </line>

    }
    </table>
```

FIG. E.1 – La requête XQuery *Aliment/effet du facteur analysé*

E.2 Requête sur la vue *charge microbienne*

L'utilisateur saisit : *food* **: cabbage,** *Ph* **: 5,** *Microorganism* **: Listeria M.**
Cette requête est exprimée en langage XQuery de la manière suivante :

```
Let $food:="cabbage" Let $microorganism :="Listeria M."

For $content IN table
    Return
    <table>
    <title>$content/ancestor()//table-title</title>
    {
    For $line IN $content/rowRel
    if($line/@additionnalAttr ="yes") then
        if($line//attribute/@attrMatch !="attribute") then
            Let AttrATerm :=$line//attribute/@attrMatch
            Let ATermIdentified :="yes"
        else
            Let ATermIdentified :="no"
        Let $data := $line//attribute
        Let $resType := "generic"
    else
        Let data :=()
        Let $resType := "notGeneric"

        Let $foodData := $line/food/MicroorganismConcetration/food
        Let $microorganismData := $line/food/MicroorganismConcetration/microorganism/ontoVal
        Let $phData :=$line/foodPh/ph
        Where(min,  ad($food, foodData),
                ad($microorganism, $microorganismData))>0)
            Return
            <line>
                $line
                if($resType ="generic") then
                    <additionalInfo> $data </additionalInfo>
                if($ATermIdentified = "yes") then
                    <ATerm> $AttrATerm </ATerm>
            </line>

    }
    </table>
```

FIG. E.2 – la requête XQuery *Charge microbienne*

Annexe F

La description WSDL du service Web XTab2SML

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
targetNamespace="http://edot.lri.fr:8080/axis/services/XTabToSML"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://edot.lri.fr:8080/axis/services/XTabToSML"
xmlns:intf="http://edot.lri.fr:8080/axis/services/XTabToSML"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:message name="launchXTabToSMLRequest">
    <wsdl:part name="xTabContent" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="launchXTabToSMLResponse">
    <wsdl:part name="launchXTabToSMLReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:portType name="CallXTabToSML">
    <wsdl:operation name="launchXTabToSML" parameterOrder="xTabContent">
      <wsdl:input message="impl:launchXTabToSMLRequest" name="launchXTabToSMLRequest"/>
      <wsdl:output message="impl:launchXTabToSMLResponse" name="launchXTabToSMLResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="XTabToSMLSoapBinding" type="impl:CallXTabToSML">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="launchXTabToSML">
      <wsdlsoap:operation soapAction=""/>
      <wsdl:input name="launchXTabToSMLRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://xTabToSML" use="encoded"/>
      </wsdl:input>
      <wsdl:output name="launchXTabToSMLResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://edot.lri.fr:8080/axis/services/XTabToSML" use="encoded"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="CallXTabToSMLService">
    <wsdl:port binding="impl:XTabToSMLSoapBinding" name="XTabToSML">
      <wsdlsoap:address location="http://edot.lri.fr:8080/axis/services/XTabToSML"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Fig. F.1 – La description WSDL du service Web XTab2SML

Bibliographie

[ACG02] Rohit Ananthakrishna, Surajit Chaudhuri, and Venkatesh Ganti. Eliminating fuzzy duplicates in data warehouses. In *VLDB '2002 : Proceedings of the 28th international conference on Very Large Data Bases*, pages 586–597. VLDB Endowment, 2002.

[ACG+06] Philippe Adjiman, Philippe Chatalic, François Goasdoué, Marie-Christine Rousset, and Laurent Simon. Distributed reasoning in a peer-to-peer setting : Application to the semantic web. *J. Artif. Intell. Res. (JAIR)*, 25 :269–314, 2006.

[AGM+90] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J Mol Biol*, 215(3) :403–410, October 1990.

[Aiz03] Akiko Aizawa. An information-theoretic perspective of tfidf measures. *Inf. Process. Manage.*, 39(1) :45–65, 2003.

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows : Theory, Algorithms, and Applications*. Prentice Hall, February 1993.

[AS02] R. Agrawal and R. Srikant. Searching with numbers, 2002.

[Baa03] Franz Baader. *The Description Logic Handbook : Theory, Implementation and Applications*. Cambridge University Press, January 2003.

[Bac06] Thanh-Le Bach. *Web sémantique multi points de vue*. PhD thesis, université de Nice - Sophia Antipolis, 2006.

[BDBHH04] Patrice Buche, Juliette Dibie-Barthélemy, Ollivier Haemmerlé, and Mounir Houhou. Towards flexible querying of xml imprecise data in a dataware house opened on the web. In *FQAS*, pages 28–40, 2004.

[BDBHT06] Patrice Buche, Juliette Dibie-Barthélemy, Ollivier Haemmerlé, and Rallou Thomopoulos. The miel++ architecture when rdb, cgs and xml meet for the sake of risk assessment in food products. In *ICCS*, pages 158–171, 2006.

[BG04] Indrajit Bhattacharya and Lise Getoor. Iterative record linkage for cleaning and integration. In *DMKD*, pages 11–18, 2004.

[BG06] Indrajit Bhattacharya and Lise Getoor. *Entity Resolution in Graphs*, chapter Mining Graph Data. Wiley, 2006.

[Bis92] Gilles Bisson. Learning in fol with a similarity measure. In *AAAI*, pages 82–87, 1992.

[BL04] Tim Berners-Lee. W3c semantic web layer cake, 2004.

[BM03a] Mikhail Bilenko and Raymond J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *KDD*, pages 39–48, 2003.

[BM03b] Misha Bilenko and Ray Mooney. On evaluation and training-set construction for duplicate detection. In *proceedings KDD-03 wokshop. on Data Cleaning, Record Linkage, and Object Consolidation*, 2003.

[BMC+03] Mikhail Bilenko, Raymond J. Mooney, William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5) :16–23, 2003.

[BN05] Alexander Bilke and Felix Naumann. Schema matching using duplicates. In *ICDE*, pages 69–80, 2005.

[CAL94] David A. Cohn, Les Atlas, and Richard E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2) :201–221, 1994.

[CHS04] Philipp Cimiano, Siegfried Handschuh, and Steffen Staab. Towards the self-annotating web. In *WWW '04 : Proceedings of the 13th international conference on World Wide Web*, pages 462–471. ACM Press, 2004.

[CL97] Chin-Liang Chang and Richard Char-Tung Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, Inc., Orlando, FL, USA, 1997.

[CM92] Michel Chein and Marie-Laure Mugnier. Conceptual graphs : Fundamental notions. *Revue d'intelligence artificielle*, 6(4) :365–406, 1992.

[Coh98a] William W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *SIGMOD Conference*, pages 201–212, 1998.

[Coh98b] William W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *SIGMOD Conference*, pages 201–212, 1998.

[Coh00] William W. Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems*, 18(3) :288–321, 2000.

[Coh03] William Cohen. http ://secondstring.sourceforge.net, 2003.

[CRF] W. Cohen, P. Ravikumar, and S. Fienberg.

[Dea01] Mike Dean. The rule markup initiative (http ://www.ruleml.org/), 2001.

[DHM05] Xin Dong, Alon Halevy, and Jayant Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD '05 : Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 85–96, New York, NY, USA, 2005. ACM Press.

[DLD+04] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Halevy, and Pedro Domingos. imap : discovering complex semantic matches between database schemas. In *SIGMOD '04 : Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 383–394, New York, NY, USA, 2004. ACM Press.

[DLLH03] AnHai Doan, Ying Lu, Yoonkyong Lee, and Jiawei Han. Object matching for information integration : A profiler-based approach. In *IIWeb*, pages 53–58, 2003.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *the Royal Statistical Society*, pages 1–38, 1977.

[DOM] http ://www.w3c.org/dom.

[DSD98a] Debabrata Dey, Sumit Sarkar, and Prabuddha De. Entity matching in heterogeneous databases : A distance based decision model. In *HICSS '98 : Proceedings of the Thirty-First Annual Hawaii International Conference on System Sciences-Volume 7*, page 305, Washington, DC, USA, 1998. IEEE Computer Society.

[DSD98b] Debabrata Dey, Sumit Sarkar, and Prabuddha De. A probabilistic decision model for entity matching in heterogeneous databases. *Manage. Sci.*, 44(10) :1379–1395, 1998.

[Dun46] Halbert L. Dunn. Record linkage. *American Journal of Public Health 36*, pages 1412–1416, 1946.

[ECJ+99] D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, D. W. Lonsdale, Y.-K. Ng, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data Knowl. Eng.*, 31(3) :227–251, 1999.

[e.d04] e.dot. Report of the e.dot project. http ://www-rocq.inria.fr/ amann/edot/, 2004.

[Euz04] Jérôme Euzenat. An api for ontology alignment. In *International Semantic Web Conference*, pages 698–712, 2004.

[EV04] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in owl-lite. In *ECAI*, pages 333–337, 2004.

[FdK93] Kenneth D. Forbus and Johan de Kleer. *Building problem solvers.* MIT Press, Cambridge, MA, USA, 1993.

[FH99] Ernest J. Friedman-Hill. Jess, the java expert system shell, available at url : http ://-herzberg.ca.sandia.gov/jess/, 1999.

[Fis73] Susan Fischer. Two processes of reduplication in the american sign language. *Foundations of Language*, 9 :460–480, 1973.

[FS69] Ivan P. Fellegi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328) :1183–1210, 1969.

[GBL98] Lee C. Giles, Kurt Bollacker, and Steve Lawrence. Citeseer : An automatic citation indexing system. In Ian Witten, Rob Akscyn, and Frank M. Shipman, editors, *Digital Libraries 98 - The Third ACM Conference on Digital Libraries*, pages 89–98, Pittsburgh, PA, June FebruaryMarch–FebruaryJune 1998. ACM Press.

[GC] Google and Code. http ://code.google.com/.

[GFS01] Helena Galhardas, Daniela Florescu, and Dennis Shasha. Declarative data cleaning : Language, model, and algorithms, 2001.

[GFSS00] Helena Galhardas, Daniela Florescu, Dennis Shasha, and Eric Simon. Ajax : an extensible data cleaning tool. In *SIGMOD '00 : Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, page 590, New York, NY, USA, 2000. ACM Press.

[GHM+05] Hélène Gagliardi, Ollivier Haemmerlé, Damiano Migliori, Nathalie Pernelle, Marie-Christine Rousset, and Fatiha Saïs. Enriching a relational data warehouse by integrating xml data : report on the e.dot project applied to microbiology. In *Second Franco Japanese Workshop on Information Search Integration and Personnalization (ISIP)*, 2005.

[GHPS05a] Hélène Gagliardi, Ollivier Haemmerlé, Nathalie Pernelle, and Fatiha Saïs. A semantic enrichment of data tables applied to food risk assessment. In *Discovery Science*, pages 374–376, 2005.

[GHPS05b] Hélène Gagliardi, Ollivier Haemmerlé, Nathalie Pernelle, and Fatiha Saïs. An automatic ontology-based approach to enrich tables semantically. In *Workshop AAAI on Context and Ontology*, pages 64–71, july 2005.

[GHPS06] Hélène Gagliardi, Ollivier Haemmerlé, Nathalie Pernelle, and Fatiha Saïs. Découverte de relations candidates à l'enrichissement d'entrepôt thématiques. In *Colloque sur l'Optimisation et les Systèmes d'Information (COSI)*, pages 142–154, 2006.

[GIKS03] Luis Gravano, Panagiotis G. Ipeirotis, Nick Koudas, and Divesh Srivastava. Text joins in an rdbms for web data integration. In *WWW*, pages 90–101, 2003.

[GKMS04] Sudipto Guha, Nick Koudas, Amit Marathe, and Divesh Srivastava. Merging the results of approximate match operations. In *VLDB*, pages 636–647, 2004.

[GL89] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Baltimore, MD, USA, second edition, 1989.

[goo02] http ://labs.google.com/sets, 2002.

[Gru93] Thomas R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers.

[HBDBH07] Gaëlle Hignette, Patrice Buche, Juliette Dibie-Barthélemy, and Ollivier Haemmerlé. Annotation sémantique floue de tableaux guidée par une ontologie. In *EGC*, pages 587–598, 2007.

[Hea92] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545. Association for Computational Linguistics, 1992.

[HGP+04] Katherine G. Herbert, Narain H. Gehani, William H. Piel, Jason T. L. Wang, and Cathy H. Wu. Bio-ajax : an extensible framework for biological data cleaning. *SIGMOD Rec.*, 33(2) :51–57, 2004.

[HPSB+04] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. Swrl : A semantic web rule language combining owl and ruleml (a submission), May 2004.

[HPSvH03] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From shiq and rdf to owl : The making of a web ontology language. *Journal of Web Semantics*, 1(1) :7–26, 2003.

[HS98] Mauricio A. Hernandez and Salvatore J. Stolfo. Real-world data is dirty : Data cleansing and the merge/purge problem. *Data Mining and Knowledge Discovery*, 2(1) :9–37, 1998.

[Hur01] Matthew Hurst. Layout and language : Challenges for table understanding on the web. In *In Web Document Analysis, Proceedings of the 1st International Workshop on Web Document Analysis*, pages 27–30, 2001.

[HW74] Lawrence J. Henschen and Larry Wos. Unit refutations and horn sets. *J. ACM*, 21(4) :590–605, 1974.

[Jac01] Paul Jaccard. Bulletin del la société vaudoisedes sciences naturelles. 37 :241–272, 1901.

[Jah07] Stefan Jahn. Solving linear equation systems. Technical report, June 2007.

[Jar76] Matthew A. Jaro. Unimatch : A record linkage system : User's manual, 1976.

[Jar89] Matthew A. Jaro. Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida. 84(406) :414–420, 1989.

[KAM03] Anastasios Kementsietsidis, Marcelo Arenas, and Renée J. Miller. Mapping data in peer-to-peer systems : Semantics and algorithmic issues. In *SIGMOD Conference*, pages 325–336, 2003.

[KM06] Dmitri V. Kalashnikov and Sharad Mehrotra. Domain-independent data cleaning via analysis of entity-relationship graph. *ACM Transactions on Database Systems (ACM TODS)*, 31(2) :716–767, June 2006.

[KMC05] Dmitri Kalashnikov, Sharad Mehrotra, and Zhaoki Chen. Exploiting relationships for domain-independent data cleaning. In *SIAM Data Mining '05*, 2005.

[KS05] Yannis Kalfoglou and W. Marco Schorlemmer. Ontology mapping : The state of the art. In *Semantic Interoperability and Integration*, 2005.

[Len02] Maurizio Lenzerini. Data integration : a theoretical perspective. In *PODS '02 : Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, New York, NY, USA, 2002. ACM Press.

[Lev66] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.

[LLL01] Wai Lup Low, Mong Li Lee, and Tok Wang Ling. A knowledge-based approach for duplicate elimination in data cleaning. *Inf. Syst.*, 26(8) :585–606, 2001.

[LSPR96] Ee-Peng Lim, Jaideep Srivastava, Satya Prabhakar, and James Richardson. Entity identification in database integration. *Information Sciences*, 89(1) :1–38, 1996.

[McB04] Brian McBride. The resource description framework (rdf) and its vocabulary description language rdfs. In *Handbook on Ontologies*, pages 51–66. 2004.

[ME96] Alvaro E. Monge and Charles Elkan. The field matching prob lem : Algorithms and applications. In *Knowledge Discovery and Data Mining*, pages 267–270, 1996.

[ME97] Alvaro E. Monge and Charles Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *In proceedings of the 2nd ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'97)*, pages 23–29, 1997.

[Mez05] Amar Djalil Mezaour. Filtering Web Documents for a Thematic Warehouse, case study : eDot a Food Risk Data Warehouse (extended). In *in Proceedings of New Trends in Intelligent Information Processing and Web Mining Conference (IIPWM'05), Gdansk, Poland.* Springer Verlag series–Advances in Soft Computing–, June 13-16 2005.

[Mil95] George A. Miller. Wordnet : a lexical database for english. *Commun. ACM*, 38(11) :39–41, November 1995.

[Min74] Marvin Minsky. A framework for representing knowledge. Technical Report 306, Cambridge, MA, USA, 1974.

[MvH04] Deborah L. McGuinness and Frank van Harmelen. Owl web ontology language overview, February 2004.

[New88] H. B. Newcombe. *Handbook of record linkage : methods for health and statistical studies, administration, and business.* Oxford University Press, Inc., New York, NY, USA, 1988.

[NK62] Howard B. Newcombe and James M. Kennedy. Record linkage : making maximum use of the discriminating power of identifying information. *Commun. ACM*, 5(11) :563–566, 1962.

[NKAJ59] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. *Science*, 130 :954–959, October 1959.

[NMTM00] Kamal Nigam, Andrew K. McCallum, Sebastian Thrun, and Tom M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3) :103–134, 2000.

[Noy04] Natalya Noy. Semantic integration : a survey on ontology-based approaches. *SIGMOD Record, Special Issue on Semantic Integration*, 2004.

[omn05] http ://www.nuance.fr/omnipage/, 2005.

[PCS04] Aleksander Pivk, Philipp Cimiano, and York Sure. From tables to frames. In *International Semantic Web Conference*, pages 166–181, 2004.

[PD77] R. L. Graham Persi Diaconis. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society*, 39(2) :262–268, 1977.

[pic05] http ://www.lri.fr/ sais/picsel3/, 2005.

[PKB93] Sheth Amit P., Gala Sunit K., and Navathe Shamkant B. On automatic reasoning for schema integration. *International Journal of Intelligent and Cooperative Information Systems*, 2(1) :23–50, 1993.

[PS07] Nathalie Pernelle and Fatiha Saïs. Passage à l'echelle de la réconciliation de concepts et de la réconciliation de références : quelques points de comparaisons. In *workshop DECOR : Passage à l'échelle des techinques de découverte de correspondances of EGC 2007*, pages 70–76, january 2007.

[Qui93] J. Ross Quinlan. *C4.5 : programs for machine learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[Rab90] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990.

[RB01] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4) :334–350, 2001.

[RN95] Stuart Russell and Peter Norvig. *Artificial Intelligence : A Modern Approach, 2nd Edition.* Prentice Hall Series in Artificial Intelligence. Pearson Education, 1995.

[Rob65] Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal ACM*, 12(1) :23–41, 1965.

[RS07] Chantal Reynaud and Brigitte Safar. Techniques structurelles d'alignement pour portails web. *RNTI, Revue des Nouvelles Technologies de l'I nformation*, 2007.

[Rus18] Robert Russell. Index, u.s. patent 1,261,167, available at http ://-patft.uspto.gov/netahtml/srchnum.htm, April 1918.

[RW69] Alan Robinson and Larry Wos. Paramodulation and theorem-proving in first-order theories with equality. *Machine Intelligence*, 4 :135–150, 1969.

[SD04] Parag Singla and Pedro Domingos. Multi-relational record linkage. In *KDD-2004 Workshop on Multi-Relational Data Mining*, pages 31–48, 2004.

[SD05] Parag Singla and Pedro Domingos. Object identification with attribute-mediated dependences. In *PKDD*, pages 297–308, 2005.

[SE05] Pavel Shvaiko and Jérôme Euzenat. A survey of schema-based matching approaches. pages 146–171, 2005.

[SGHP05] Fatiha Saïs, Hélène Gagliardi, Ollivier Haemmerlé, and Nathalie Pernelle. Enrichissement sémantique de documents xml représentant des tableaux. In *EGC*, pages 407–418, 2005.

[SLD05] Warren Shen, Xin Li, and AnHai Doan. Constraint-based entity matching. In *AAAI*, pages 862–867, 2005.

[SM01] Gerd Stumme and Alexander Maedche. Fca-merge : bootm up merging of ontologies. *In the proceedings of IJCAI*, pages 225–234, 2001.

[Sow84] John F. Sowa. *Conceptual Structures : Information Processing in Mind and Machine*. Addison-Wesley, 1984.

[SPR07a] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. Approche logique pour la réconciliation de références. In *EGC*, pages 623–634, 2007.

[SPR07b] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. L2R : A Logical Method for Reference Reconciliation. In *AAAI*, pages 329–334, 2007.

[SPR07c] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. Réconciliation de références : une approche adaptée aux grands volumes de données. In *Colloque sur l'Optimisation et les Systèmes d'Information (COSI)*, pages 521–532, june 2007.

[Sym99] http ://www.symprevius.net, 1999.

[Taf70] Robert L. Taft. Name search techniques, 1970.

[TBP07] Mouhamadou Thiam, Nacéra Bennacer, and Nathalie Pernelle. Webdocenrich : enrichissement sémantique flexible de documents semi-structurés. In *EGC*, pages 211–212, 2007.

[TKM01] Sheila Tejada, Craig A. Knoblock, and Steven Minton. Learning object identification rules for information integration. *Information Systems*, 26(8) :607–633, 2001.

[TRB95] Donald B. Rubin Thomas R. Belin. A method for calibrating false-match rates in record linkage. *Journal of the American Statistical Association*, 90(430) :694–707, 1995.

[Ull77] Julian R. Ullmann. A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words. *Comput. J.*, 20(2) :141–147, 1977.

[VEH00] Vassilios S. Verykios, Ahmed K. Elmagarmid, and Elias N. Houstis. Automating the approximate record-matching process. *Information Sciences*, 126(1–4) :83–98, 2000.

[VME03] V. S. Verykios, G. V. Moustakides, and M. G. Elfeky. A bayesian decision model for cost optimal record matching. *The VLDB Journal*, 12(1) :28–40, 2003.

[web06] http ://www.webcontent-project.org/, 2006.

[WEW91] Yves Thibaudeau William E. Winkler. An application of the fellegi-sunter model of record linkage to the 1990 u.s. decennial census. Technical report, 1991.

[WEW93] Yves Thibaudeau William E. Winkler. Improved decision rules in the fellegi-sunter model of record linkage. Technical report, 1993.

[Wie92] Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25(3) :38–49, 1992.

[Wil82] R. Wille. Restructuring lattice theory : An approach based on hierarchies of concepts. In *Ordered Setsand in I. Rivals (Ed.)*, volume 23, pages 445–470, 1982.

[Win89] W. Winkler. Near automatic weight computation in the fellegi-sunter model of record linkage, 1989.

[Win02] William E. Winkler. Methods for record linkage and bayesian networks, 2002.

[Win06] William E Winkler. Overview of record linkage and current research directions. Technical report, Statistical Research Division U.S. Census Bureau Washington, DC 20233, 2006.

[WM89] Y. Richard Wang and Stuart E. Madnick. The inter-database instance identification problem in integrating autonomous systems. In *Proceedings of the Fifth International Conference on Data Engineering*, pages 46–55, Washington, DC, USA, 1989. IEEE Computer Society.

[wor95] http ://wordnet.princeton.edu/, 1995.

[WSB76] Michael S. Waterman, Temple F. Smith, and William A. Beyer. Some biological sequence metrics. *Advances in Mathematics*, 20(4) :367–387, 1976.

[WSD01] http ://www.w3.org/tr/wsdl, 2001.

[XE03] Li Xu and David W. Embley. Discovering direct and indirect matches for schema elements. In *DASFAA '03 : Proceedings of the Eighth International Conference on Database Systems for Advanced Applications*, page 39, Washington, DC, USA, 2003. IEEE Computer Society.

[Yan05] William E. Yancey. Evaluating string comparator performance for record linkage. Technical report, Statistical Research report series RRS2005/05, US, Bureau of the census, Washington, 2005.

[Zad65] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3) :338–353, June 1965.