République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université Djillali Liabès de Sidi-Bel-Abbès

Faculté des Sciences Exactes
Département d'informatique

Thèse

Pour l'obtention du Diplôme de Doctorat LMD en Informatique
Option : Informatique

Présenté par :
Mlle. Khayra BENCHERIF

Enrichissement et Intégration des Données Liées

Directeur de thèse :
Pr. Mimoun MALKI
Professeur à l'école supérieure de l'informatique de Sidi-Bel-Abbès

Devant le jury composé de :

Président :	Dr. Adel Tomoh (M.C.A)	Université Djilali Liabes
Examinateurs :	Pr. Sidi Mohamed Benslimane (Professeur)	ESI-SBA
	Dr. Zohra Slama (M.C.A)	Université Djilali Liabes
	Dr. Djamal Berrabah (M.C.A)	Université Djilali Liabes
Directeur de thèse :	Pr. Mimoun MALKI (Professeur)	ESI-SBA

Remerciements

Je remercie d'abord le bon Dieu pour toute la grâce qu'il m'a accordé jusqu'à ce jour.

Je tiens à remercier chaleureusement professeur Mimoun MALKI qui a dirigé mes travaux de thèse. Je tiens à lui exprimer ma gratitude pour la qualité de ses conseils, son disponibilité ainsi que le degré de responsabilisation de son encadrement qui m'a permis de développer mon goût pour la recherche.

Je tiens à remercier tous les membres du jury qui ont accepté de juger ce travail, le président du jury Monsieur Adel Tomoh et les examinateurs Monsieur Sidi Mohamed Benslimane, Madame Zohra Slama et Monsieur Djamal Berrabah.

Mes remerciements vont aussi à tous les membres du laboratoire EEDIS (Evolutionary Engineering and Distributed Information Systems), doctorants et permanents.

J'adresse aussi mes sincères remerciements à tous les membres de ma famille : ma mère, mon père, mes soeurs et mes frères pour ses générosités et ses soutiens morals qui ont été pour moi une source de courage et de confiance.

Enfin, je souhaite exprimer ma gratitude à tous ceux qui directement ou indirectement m'ont apporté leurs aides.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction générale

Contents

Les savants sont ceux qui mettent en application ce qu'ils ont appris.

Prophète Mohammad (paix et bénédiction d'Allah soient sur lui)

1.1 Contexte Général

Le web de documents est fondé sur un ensemble de normes simples : un mécanisme d'identification unique (URI) (Berners-Lee et al., August 1998), un mécanisme d'accès universel (le protocole HTTP) (Fielding, 1999) et le langage HTML (Raggett et al., 1999) comme un format de contenu. En outre, le web de documents permet de créer des liens hypertextes entre les documents web qui peuvent résider sur différents serveurs web. Donc, les liens hypertextes sont cruciaux pour relier le contenu de différents serveurs dans un espace d'informations global. Cependant, les pages web qui existent dans le web de documents sont lisibles et compréhensibles uniquement par l'homme. En revanche, elles sont présentées dans un format inexploitable automatiquement par les machines. Donc, les machines ne peuvent pas identifier ou interpréter les informations contenues dans les pages web qui constituent le web de documents.

En 2001, l'inventeur du web Tim Berners-Lee (Berners-Lee, 2006) a proposé une nouvelle vision du web (appelé le web sémantique (Berners-Lee et al., 2001)) pour remédier aux insuffisances du web de documents (Zghal, 2010). En fait, le web sémantique est basé sur l'utilisation des ontologies pour décrire la sémantique des données contenues dans les pages web. Donc, il l'a conçu pour être interprétable par les humains et les machines (en particulier les agents logiciels) grâce à la mise en place des liaisons sémantiques entre les données qui se trouvent à l'intérieur des documents.

Le terme "Linked Data" (données liées) se réfère à un ensemble de meilleures pratiques pour la publication et l'interconnexion des données structurées sur le web. Ces meilleures pratiques ont été introduites par Tim Berners-Lee dans son architecture web et sont devenus connus comme "les principes des données liées". Comme les hyperliens relient les documents du web classique pour fournir un espace d'informations global, les données liées permettent d'établir des liaisons sémantiques entre les données qui se trouvent à l'intérieur des documents et de déréférencier les URIs pour construire un seul espace de données global, le web de données.

Le web de données permet de publier des données structurées et non structurées sur le web, non pas sous la forme de silos de données isolés les uns des autres, mais en les reliant pour constituer un réseau d'informations global. Dans ce contexte, le projet Linked Open Data Cloud (appelé LOD Cloud [1]) a été créé. Il permet de publier et de connecter des données structurées sur le web selon les principes des données liées. Linked Open Data Cloud comporte un ensemble de bases de connaissances qui ont

1. http://lod-cloud.net/

été publiées et reliées entre elles par le projet jusqu'à présent. En fait, le LOD Cloud couvre de nombreux domaines, tels que les personnes, les entreprises, les films, de la musique, les lieux, les livres et autres publications, les communautés en ligne, ainsi qu'un volume croissant de données scientifiques et gouvernementales.
Avec la croissance des sources de données disponibles dans le LOD Cloud, le problème d'hétérogénéité de données dans ces sources augmente et donc le besoin d'accès à toutes ces sources à travers une interface cohérente a été le défi de nombreuses recherches dans le domaine d'intégration des données liées. Dans cette perspective, il s'avère nécessaire de fournir une vue unifiée de différents ensembles de données liées en combinant les informations distinctes et hétérogènes. En parallèle à la liaison des entités, s'ajoute l'intégration des données liées qui exige également l'alignement des différents vocabulaires utilisés pour décrire les entités ainsi que la résolution des conflits de données entre les sources de données liées. Donc, trois étapes sont nécessaires pour obtenir un web de données intégré : la première étape consiste à appliquer un ensemble de mappings au niveau de schémas (l'alignement de schémas), la deuxième étape consiste à aligner les différentes entités qui représentent le même objet du monde réel dans les sources de données (la détection de doublons ou la résolution d'identité) et la dernière étape consiste à évaluer la qualité des données et la fusion des représentations en double pour résoudre les incohérences des données (l'évaluation de la qualité et la fusion des données).
Les deux premières étapes du processus d'intégration des données liées consistent à découvrir des liens au niveau de schémas et au niveau d'instances. Bien que les liens entre les ensembles de données liées du LOD Cloud jouent un rôle primordial dans l'intégration des données, moins de 23% de triplets RDF représentent des liens entre les ensembles de données liées. En général, la tâche de découverte des liens est devenue difficile parce que les ensembles de données liées ne partagent pas des entités communes acceptées d'une part et ne réutilisent pas les vocabulaires et les ontologies préexistantes d'autre part. La découverte des liens au niveau d'instances permet de trouver les différentes entités qui représentent le même objet du monde réel et d'établir un lien de type "owl :sameAs" entre eux. Cependant, les frameworks de découverte des liens doivent être évolutives (performantes) car elles doivent produire en un temps raisonnable des liens entre les grands ensembles de données liées qui contiennent des milliers d'instances.
La troisième étape du processus d'intégration des données liées consiste à évaluer la qualité des données et de fusionner des informations provenant de plusieurs sources de données. Selon (Jens and Felix, 2009), la qualité des données peut être évaluée à l'aide de plusieurs dimensions : la complétude, la concision, la cohérence, l'exacti-

tude, l'intelligibilité, l'objectivité, la rapidité, la crédibilité, la pertinence, la vérifiabilité, la disponibilité, etc. Ces dimensions ne sont pas indépendantes les unes des autres, et en général, seulement un sous ensemble d'entre eux est pertinent dans une situation spécifique. Pour que les applications consomment efficacement des données liées, elles doivent être évaluées et intégrées en fonction de leurs qualités. Dans le contexte d'intégration des données, la fusion des données est définie comme "le processus de combinaison de plusieurs enregistrements représentant le même objet du monde réel dans une représentation unique, cohérente et propre" (Jens and Felix, 2009). En fait, la fusion des données liées a deux objectifs principaux : accroître la complétude et la concision des données qui sont à la disposition des utilisateurs et des applications.

1.2 Problématiques abordées

Ce travail se concentre principalement sur l'intégration des ensembles de données liées du LOD Cloud en combinant des données hétérogènes pour faciliter l'interrogation des sources de données multiples. Dans cette thèse, nous avons soulevé deux problèmes principaux pour concevoir un système d'intégration de données liées :

1. **Les challenges de la découverte des liens entre les ensembles de données liées :**
   La liaison peut être généralement définie comme la connexion des choses qui sont en quelque sorte liées. Dans le contexte des données liées, l'idée de la liaison est particulièrement préoccupée par l'établissement des liens typés entre les entités (à savoir, les classes, les propriétés ou les instances) contenues dans les bases de connaissances. Au cours de ces dernières années, plusieurs approches ont été développées pour pallier le manque de liens typés entre les différentes bases de connaissances sur le web de données liées. En général, deux catégories d'approches qui visent à atteindre cet objectif peuvent être différenciées. La première catégorie met en oeuvre des techniques de matching d'ontologies et vise à établir des liens entre les ontologies sous-jacentes deux sources de données. La deuxième catégorie, appelée des approches de découverte des liens, permet de découvrir des liens entre les instances contenues dans deux sources de données.
   Principalement, trois défis majeurs se posent lors de la découverte des liens entre deux ensembles d'instances : la complexité temporelle du processus de découverte des liens, le choix d'une spécification de liens appropriée et la visualisation des données liées.

- La complexité temporelle du processus de matching peut être mesurée par le nombre de comparaisons nécessaires pour accomplir cette tâche. Plusieurs approches de découverte des liens dans les données liées ont une complexité temporelle quadratique (elles nécessitent $O(|S||T|)$ comparaisons dans le processus de matching) car elles comparent chaque instance de la base de connaissance sources S avec toutes les instances de la base de connaissance cible T. Cette complexité est impraticable quand les bases de connaissances deviennent grandes. Donc, la réduction du temps d'exécution est une exigence cruciale dans le processus de découverte des liens.
- Le deuxième défi du processus de découverte des liens réside dans la sélection d'une spécification appropriée pour l'établissement des liens. En général, la configuration des frameworks de découverte des liens est effectuée manuellement. Pourtant, le choix d'une mesure de similarité appropriée est central pour découvrir des liens satisfaisants et par conséquent améliorer la qualité des mappings. Donc, soutenir l'utilisateur pour choisir une mesure de similarité et un seuil appropriés est un problème qui doit encore être abordé par la communauté de Linked Data.

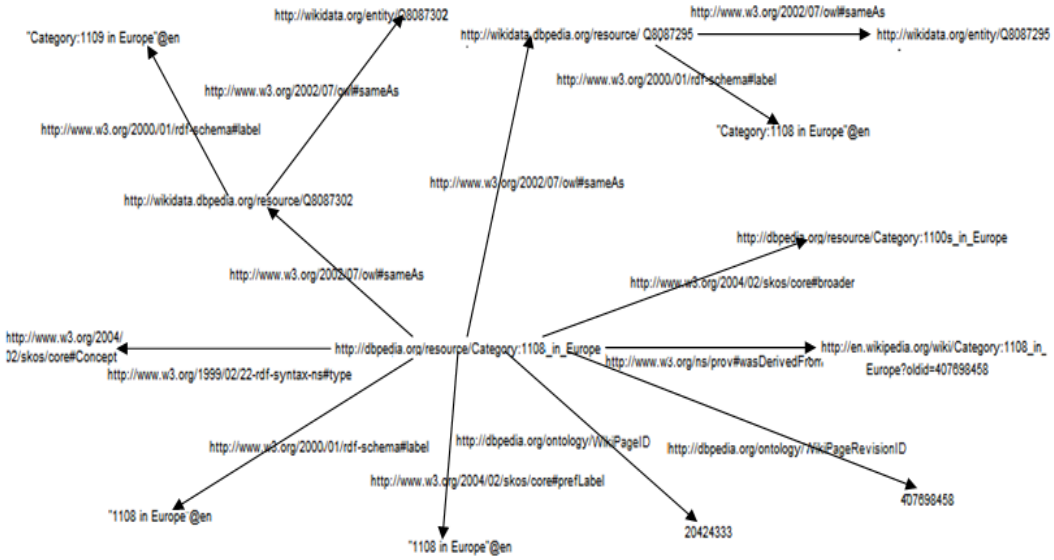

Figure 1.1 – Les instances liées par le prédicat "owl :sameAs".

- La taille des ensembles de données liées qui existent dans le LOD Cloud présente des défis lorsqu'un utilisateur (principalement des utilisateurs sans expérience de technologies du web sémantique) essaie de visualiser, de rechercher et d'utiliser des données au sein de ces ensembles de données. En particulier, les outils de visualisation des données liées ignorent la visualisation des liens

qui relient les ressources d'un ensemble de données liées avec des ressources d'un autre ensemble de données liées. Donc, la visualisation des données liées est une tâche cruciale qui permet aux utilisateurs de connaitre et de corriger les erreurs liées à la découverte des liens.

2. **Le problème de la fusion des données :**
   Actuellement, un nombre croissant d'ensembles de données disponibles dans le LOD Cloud extrait automatiquement des données à partir de plusieurs sources d'informations en utilisant différents extracteurs d'informations. La capacité de choisir et de fusionner des données hétérogènes dans le LOD Cloud d'une manière efficace est une étape clé vers le développement des applications de données liées. En fait, la fusion des ensembles de données contenant des informations conflictuelles permet aux internautes d'obtenir une vue complète et concise de toutes les données existantes, sans aucune contradiction. Les plus gros problèmes dans le contexte de la fusion des données liées sont de (1) surmonter l'effet que plusieurs ensembles de données représentent le même objet du monde réel en utilisant différentes entités, (2) choisir les bonnes solutions (entités) au cours de la tâche de fusion (améliorer l'efficacité du système de fusion des données liées), et (3) minimiser le temps d'exécution du processus de fusion (améliorer la performance du système de fusion des données liées). L'efficacité est affectée par la qualité des données à fusionner, par conséquent, le choix de la meilleure solution est un défi majeur dans la tâche de fusion des données. La performance est affectée par le grand nombre d'instances dans le LOD Cloud, donc, minimiser le temps d'exécution est un enjeu crucial dans la tâche d'intégration des données liées. La figure 1.1 illustre des instances reliées entre eux par le prédicat *"owl :sameAs"* ainsi que tous les prédicats et les objets liés avec ces instances. Comme ils contiennent des doublons, nous devons les fusionner afin que chacun ait une valeur unique.

1.3 Objectifs

L'objectif du présent travail est de proposer une approche de découverte des liens ainsi qu'une approche de fusion des données liées. Notre approche de découverte des liens permet d'améliorer la complexité temporelle et par conséquent minimiser le temps d'exécution du processus de matching. Ainsi, notre approche de fusion des données liées permet de choisir les solutions correctes et de minimiser le temps d'exécution lors du processus de fusion des données liées. Par conséquent, les objectifs de cette thèse sont énumérés dans ce qui suit :

Objectif 1 : Nous cherchons à résoudre le problème d'hétérogénéité sémantique qui existe dans les ensembles de données liées du LOD Cloud.
Objectif 2 : Dans les dernières années, le nombre de sources des données disponibles dans le LOD Cloud augmente rapidement. D'où, la nécessité de développer des méthodes pour réduire le nombre de comparaisons lors de la découverte des liens entre les grands ensembles de données liées et par conséquent optimiser la complexité temporelle et minimiser le temps d'exécution. En d'autres termes, ces méthodes peuvent déterminer un nombre maximal de liens entre les différentes entités dans un temps raisonnable.
Objectif 3 : La littérature comporte plusieurs approches de découverte des liens. Ces approches utilisent des mesures de similarité de chaînes de caractères (telles que la mesure de Levenshtein (Levenshtein, 1966), de Jaro (Jaro, 1989), de Jaccard, etc), des mesures numériques, des mesures de date, ou les coordonnées géographiques pour calculer la similarité entre les entités. Cependant, elles n'utilisent pas des sources de connaissances externes, comme les dictionnaires (Wordnet). Donc, notre objectif est d'utiliser des sources de connaissances externes afin de produire des liens de haute qualité et donc améliorer l'efficacité du système de découverte des liens.
Objectif 4 : La visualisation des liens après le processus de la découverte des liens est une tâche cruciale. Donc, la nécessité de fournir un outil de visualisation des liens pour aider les experts et les utilisateurs à détecter des erreurs dans le processus de matching d'une part et de vérifier la complétude et l'exactitude des liens, d'autre part.
Objectif 5 : La fusion des ensembles de données contenant des informations contradictoires permet aux internautes d'obtenir une vue complète et concise de toutes les données existantes, sans aucune contradiction. D'où, notre objectif est de proposer une approche de fusion des données liées pour combiner automatiquement, efficacement et rapidement les différentes informations provenant de plusieurs sources de données liées.

1.4 Principales contributions

En prenant en considération les objectifs cités auparavant, la méthodologie suivie comporte trois phases principales : une phase d'analyse, une phase de développement et une phase d'évaluation.
1) La phase d'analyse comporte une étude du domaine d'intégration des données liées. Pour cela, nous allons effectuer une recherche bibliographique sur les méthodes et les solutions qui ont été proposées dans le domaine d'intégration des données liées,

Problème	Notre contribution
La complexité temporelle dans la tâche de découverte des liens	Indexing-based link discovery in Linked Data
La visualisation des données liées	Linked Data Visualization Tool
La fusion des données liées	An effective and time-efficient approach for Linked Data fusion using genetic algorithms

Table 1.1 – Les problèmes liées à l'intégration des données liées et nos propositions

en particulier les techniques de découverte des liens au niveau de schémas et au niveau d'instances ainsi que des approches de fusion des données liées.
2) La phase de développement détaille une méthode de découverte des liens dans des ensembles de données liées, un outil de visualisation de Linked Data et une approche de fusion des données liées. Le tableau 1.1 illustre les problèmes soulevés ainsi que les approches proposées.
3) La phase d'évaluation comporte l'expérimentation et l'évaluation de nos approches ainsi qu'une comparaison avec des approches qui existent dans la littérature.
En prenant en considération les objectifs fixés et les étapes précédentes, les principales contributions de la thèse peuvent être récapitulées comme suit :

1) La découverte des liens dans Linked Data en se basant sur l'indexation :

Au cours de ces dernières années, le nombre de sources de données qui existent dans le LOD Cloud augmente rapidement. Pour faciliter la combinaison des informations provenant de différentes sources, il est nécessaire d'établir des liens typés entre les éléments de ces sources de données. En raison de la grande quantité d'informations disponibles dans le LOD Cloud, les approches de découverte des liens dans un temps efficace sont devenues indispensables pour mettre en oeuvre le quatrième principe des données liées, à savoir la fourniture des liens entre les sources de données. Actuellement, plusieurs recherches permettent de découvrir des liens typés entre les instances qui représentent le même objet du monde réel dans différentes sources de données. Cependant, la tâche de la découverte des liens entre les grands ensembles de données nécessite un grand nombre de comparaisons et par conséquent une complexité temporelle élevée.
Afin de résoudre ce problème, nous avons proposé une approche (Benchérif et al., Mai 2016) sans perte qui permet de réduire le nombre de comparaisons dans le processus de matching et par conséquent optimiser le temps d'exécution lors de mapping de grands ensembles de données. Notre approche améliore le premier algorithme de LIMES (une approche de découverte de liens dans les espaces métriques) par (1) l'utilisation du Wordnet pour trouver tous les synonymes de chaque instance source,

puis les indexer dans la base de connaissances cible en utilisant des indices inversés structurés pour obtenir rapidement des candidats de matching, et (2) la suppression des exemplaires doublons de la base de connaissances cible pour réduire le nombre de comparaisons dans le processus de matching.

2) La visualisation des données liées :

Le potentiel de la grande quantité de données liées sur le web est énorme, mais dans la plupart des cas, il est très difficile pour les utilisateurs de visualiser, d'explorer et d'utiliser ces données ; en particulier pour les utilisateurs qui n'ont pas une expérience avec les technologies du web sémantique. Donc, une visualisation claire et cohérente des données liées permet d'offrir des interfaces pour explorer, trouver des informations dans le web de données et d'aider les utilisateurs à reconnaître toutes les erreurs qui résultent après le processus de découverte des liens entre les ensembles de données pour vérifier la précision et la complétude des liens. Afin de répondre à ce besoin, nous avons développé un outil de visualisation des données liées (Bencherif and Malki, Novembre 2016). Il prend en entrée une ressource d'un ensemble de données liées et visualise toutes les ressources internes et externes liées à cette ressource. En outre, notre outil fournit des informations sur le nombre total de triplets, le nombre de liens de type "owl :sameAs", le nombre de propriétés et d'objets, etc. La visualisation des informations fournies sur l'URI permet aux utilisateurs d'obtenir une vue sur l'ensemble de données et de comprendre sa structure interne.

3) La fusion des données liées :

L'intégration des données est une étape clé vers le développement des applications des données liées. Elle facilite l'interrogation des ensembles de données en combinant les données provenant de sources multiples pour fournir une vue unifiée de ces sources. Le processus de la fusion des données visent à combiner, agréger et résoudre des conflits de valeurs provenant de différentes sources pour obtenir une vue unifiée entre elles. Il suit le processus de la résolution d'identité ou la détection des doublons qui définit des valeurs différentes pour représenter les mêmes objets du monde réel. Automatiser le processus de la fusion des données liées a été le défi de nombreux chercheurs dans le domaine d'intégration des données liées. Il existe différents problèmes de fusion parmi lesquels : le choix des solutions correctes et adéquates et la réduction du temps d'exécution lors de la fusion de grands ensembles de données liées.

Afin de résoudre ces problèmes, nous avons proposé une nouvelle approche (Bencherif and Malki, 2016) qui permet de fusionner automatiquement et efficacement des données liées du LOD Cloud en utilisant un algorithme génétique.

Les raisons pour lesquelles nous avons utilisé les algorithmes génétiques pour la fusion des données liées sont les suivantes :
a) Comme nous le savons, la fusion des données permet de combiner des termes redondants en un seul en conservant le meilleur d'entre eux. À son tour, un algorithme génétique évalue les meilleurs individus d'une population en utilisant une fonction d'évaluation et puis il les combine en utilisant les opérateurs de croisement et de mutation.
b) Bien que les approches de fusion des données liées utilisent différentes méthodes pour améliorer l'efficacité, la complexité est relativement élevée. Comme les algorithmes génétiques ont été utilisé pour optimiser la complexité temporelle, ils sont les plus appropriés pour améliorer la performance des systèmes de fusion des données liées.

1.5 Plan de la thèse

Ce mémoire de thèse regroupe six chapitres ; à savoir :
Dans le deuxième chapitre, les concepts de bases de Linked Data sont présentés. En outre, il illustre les étapes nécessaires pour l'intégration des ensembles de données liées. Ce chapitre introduit aussi le modèle de données RDF, RDFS, OWL et le langage d'interrogation SPARQL.
Le troisième chapitre détaille les travaux connexes de la découverte des liens dans Linked Data, l'alignement des ontologies LOD et la fusion des données liées. De plus, nous comparons les approches de la découverte des liens, les techniques d'alignement d'ontologies ainsi que les approches de la fusion des données liées en utilisant des tables avec des aspects importants pour discuter leurs limitations.
Le quatrième chapitre est dédié à la présentation de notre approche de découverte des liens à base d'indexation et notre outil de visualisation des données liées qui permettent de résoudre des problèmes liées à la découverte des liens dans les grands ensembles de données liées. Dans une première étape, nous détaillons le problème de la découverte des liens et la recherche d'informations dans les espaces métriques. Dans une seconde étape, nous présentons en détail notre approche de découverte des liens et notre outil de visualisation. Dans une troisième étape, nous présentons l'expérimentation et l'évaluation de l'approche proposée ainsi qu'une étude comparative de la méthode proposée par rapport aux méthodes qui existent dans la littérature.
Dans le cinquième chapitre, l'approche proposée pour la fusion des données liées est présentée. Dans un premier temps, nous détaillons le problème de la fusion des données et les types des conflits d'informations. Ensuite, nous présentons en détail

l'approche proposée pour la fusion des données liées qui repose sur l'utilisation d'un algorithme génétique pour améliorer les résultats de la fusion. Enfin, nous présentons l'expérimentation et l'évaluation de l'approche proposée sur deux volets : l'évaluation de l'efficacité et l'évaluation de la performance. En outre, nous présentons une comparaison entre l'approche proposée et des approches de fusion des données liées. Le sixième chapitre présente une conclusion de notre travail de recherche qui permet d'intégrer des ensembles de données liées du LOD Cloud en découvrant des liens entre ces ensembles et en fusionnant ces données. Enfin, nous proposons des travaux futurs pour concrétiser notre vision.

Chapitre 2

Préliminaires sur les données liées

Contents

Celui qui emprunte un chemin menant à l'apprentissage d'une science, Allah lui facilite l'accès au paradis.

Prophète Mohammad (paix et bénédiction d'Allah soient sur lui)

2.1 Introduction

Le web de documents est basé sur un ensemble de normes simples : un mécanisme d'identification unique (URI), un mécanisme d'accès universel (le protocole HTTP) et le langage HTML comme un format de contenu. En outre, le web de documents permet de créer des liens hypertextes entre les documents web qui peuvent résider sur différents serveurs web. Donc, les liens hypertextes sont cruciaux pour relier le contenu de différents serveurs pour founir un seul espace d'informations. Comme les hyperliens dans le web classique relient les documents web, les données liées permettent d'établir des liens entre les éléments des différentes sources de données et de déréférencier les URIs pour construire un seul espace de données global : le web de données. Ce web de données (Heath et al., 2009), aussi appelé le web sémantique (Berners-Lee et al., 2001), couvre de nombreux domaines, tels que des personnes, des entreprises, des films, de la musique, des lieux, des communautés en ligne, des données scientifiques et gouvernementaux, des livres et autres publications.
Le reste du chapitre est organisé comme suit : Nous allons citer brièvement l'évolution du web dans la section 2. La section 3 décrit les principes de base des données liées. Le modèle de données RDF, RDFS, OWL et le langage d'interrogation SPARQL sont détaillés dans les sections 5, 6, 7 et 8, respectivement. Dans la section 9, nous détaillons le quatrième principe des données liées pour décrire les types de liaisons. Nous allons citer les ensembles de données liées qui se trouvent dans le projet Linked Open Data Cloud dans la section 10. La section 11 présente les étapes d'intégration des données liées.

2.2 Evolution du web

Le web est une technologie informatique qui permet de consulter des pages regroupées sur des sites ; il a connu une évolution incroyable en 30 ans. En effet, cette évolution comporte quatre phases essentielles. La Figure 2.1 présente les phases d'évolution du web.

Le Web 1.0 (le web de documents) est un web statique qui est basé sur un ensemble de normes simples : des URIs (Identifiants de ressource uniformes) comme un mécanisme d'identification unique au monde, le protocole HTTP (Hypertext Transfer) comme un mécanisme d'accès universel et le langage HTML (Hypertext Markup Language) comme un format de contenu largement utilisé. Le langage HTML a été conçu pour écrire des pages web riches en textes et créer des liens hypertextes entre les documents web qui peuvent résider sur différents serveurs web. Mais sa tolérance aux erreurs pose plusieurs problèmes aux développeurs. Le web de documents per-

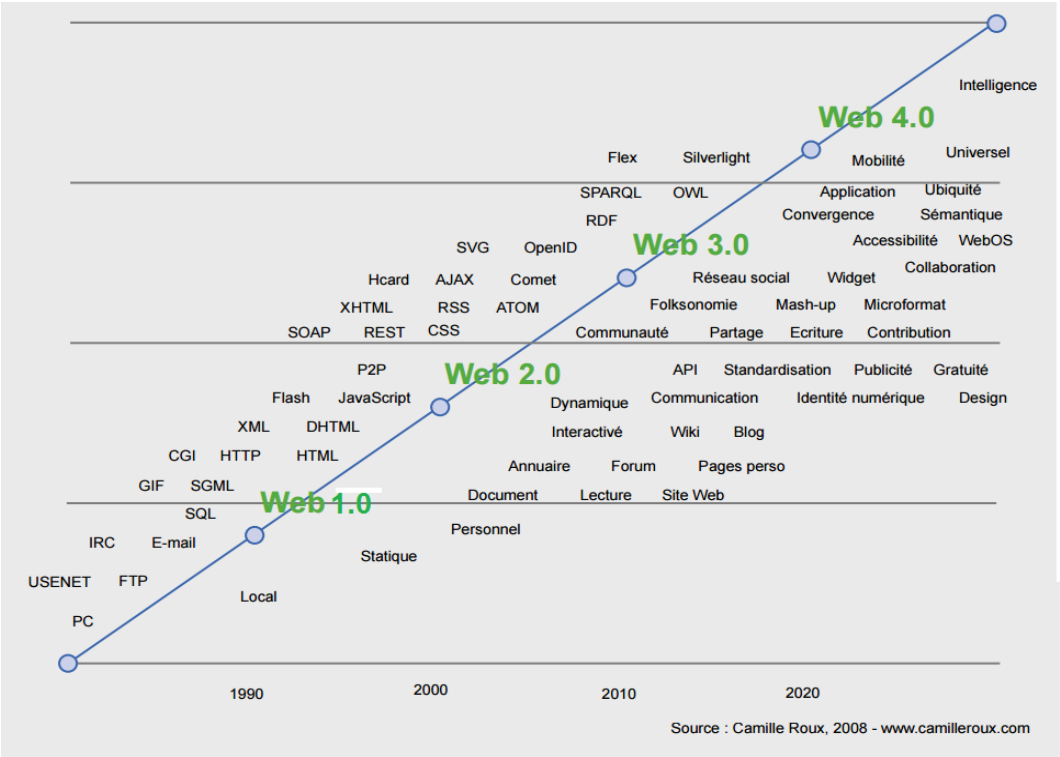

Figure 2.1 – Les phases d'évolution du web.

met de transférer des informations d'une manière unidirectionnelle (c'est-à-dire des développeurs vers les internautes). Les applications les plus utilisées dans le web 1.0 sont : les portails de contenus, les courriers électroniques et un peu de commerce en ligne.

Le Web 2.0 (le web social) privilégie la dimension de partage et d'échange d'informations et de contenus (textes, images, vidéos, etc). Ce web a été démarré avec l'apparition des scripts qui permettent aux personnes, qui ne connaissent aucun langage de programmation, de gérer les sites. Ces scripts offrent la possibilité d'insérer des modules dans les pages (heures et dates, e-mailling-list, flux RSS, etc). Ensuite, le web 2.0 a vu l'émergence des services de blog (Skyblog [1] , etc.), l'hébergement des photos et des vidéos (Picasa [2] , FlickR [3] , YouTube [4] , DailyMotion [5], etc.), les mashups et les réseaux sociaux (Facebook [6] , LinkedIn [7] , etc.). Enfin, le concept UGC (User Generated Content) est apparu. Il a été utilisé pour le travail collaboratif entre

1. http://www.skyrock.com/blog/
2. http://picasa.google.com/
3. http://www.flickr.com/
4. http://www.youtube.com/
5. http://www.dailymotion.com/
6. http://www.facebook.com/
7. http://www.linkedin.com/

les utilisateurs. Généralement, tous les utilisateurs (visiteurs) peuvent modifier ou créer un contenu (par exemple : Wikipédia [8]).

Le Web 3.0 (le web sémantique) est un espace de données unique mondialement distribué, il a été conçu pour être interprétable par les humains et les machines grâce à la mise en place des liaisons sémantiques entre les données qui se trouvent à l'intérieur des documents. Le web sémantique permet d'organiser les données en fonction du contexte et des besoins des utilisateurs et de donner un sens aux données. De plus, il ajoute des informations cachées (métadonnées) destinées à être utilisées par des moteurs de recherches, des applications, etc. Ces informations sont présentées à l'aide du formalisme RDF (basé sur le langage XML) qui permet de structurer les données sous forme de triplets <sujet, prédicat, objet>.

Le Web 4.0 (Web intelligent) vise à immerger l'individu dans un environnement web très solide qu'il fonctionnera grâce à des agents intelligents. C'est un terrain d'expérimentation qu'il n'est pas encore exploré.

2.3 Ressources, URIs et espaces de noms

Une ressource est un composant de base du web, elle peut être : un fragment d'un document XML, une page web, un identificateur pour une entité, une propriété, un concept, etc. Au début, ce terme a désigné le référent d'une URL (une page web). Ensuite, cette définition a été généralisée à tous les référents des URIs.

Une URI (Uniform Resource Identifier) est une chaîne de caractères qui identifie une ressource abstraite ou physique sur un réseau, donc tous les hyperliens du web sont exprimés sous forme d'URIs. Par exemple, la ressource "Organisation" dans l'ontologie FOAF est représentée par l'URI : http ://xmlns.com/foaf/0.1/Organization.

Pour éviter l'utilisation des URIs longues, nous pouvons utiliser des espaces de noms (en anglais namespace). En général, il est préférable de définir à l'avance les différents espaces de noms qui vont être utilisés et d'attribuer à chaque espace de nom un nom local, beaucoup plus court, afin de réduire l'écriture des triplets et de les rendre plus lisibles.

Par exemple, on peut définir *foaf* comme un espace de nom de l'ontologie FOAF. Donc, la ressource "Organisation" de l'ontologie FOAF devient *foaf :Organization*, ou *foaf* a comme URI : <http ://xmlns.com/foaf/0.1/> . Donc, *foaf :Organization* est une abréviation de l'URI actuelle pour la ressource "Organisation" dans l'ontologie FOAF.

Les espaces de noms les plus utilisés dans les ensembles de données liées sont :

8. https://www.wikipedia.org

rdf : Un espace de nom pour RDF, son URI est : http ://www.w3.org/1999/02/22-rdf-syntax-ns#.
rdfs : Un espace de nom pour RDFS, son URI est : http ://www.w3.org/2000/01/rdf-schema#.
owl : Un espace de nom pour OWL, son URI est : http ://www.w3.org/2002/07/owl#.
dc : Un espace de nom pour l'Initiative Dublin Core, son URI est : http ://dublincore.org/documents/dcmi-namespace/.
foaf : Un espace de nom pour FOAF, son URI est : http ://xmlns.com/foaf/0.1/.
sioc : Un espace de nom pour SIOC, son URI est : http ://rdfs.org/sioc/ns#.

2.4 Principes de données liées

En 2006, le terme Linked Data [9] a été bien défini par l'inventeur du web "Tim Berners Lee" comme un ensemble de meilleures pratiques pour exposer, partager et connecter des données structurées sur le web en utilisant des URIs et le modèle de données RDF. L'idée de base de Linked Data est d'appliquer l'architecture générale du World Wide Web (Jacobs and Walsh, 2004) pour partager des données structurées à l'échelle mondiale. Ensuite, ces pratiques sont devenues connus comme "les principes de données liées". Ces principes sont les suivants (Heath and Bizer, 2011) :

1. L'utilisation des URIs pour nommer les données ;
2. L'utilisation des URIs HTTP afin que les gens peuvent regarder ces noms ;
3. Fournir des informations utiles (en utilisant RDF et SPARQL) quand quelqu'un visite une URI ;
4. Inclure des liens vers d'autres URIs, afin qu'ils puissent découvrir plus de choses.

Le premier principe supporte l'utilisation des URIs pour identifier des documents web, du contenu numérique ainsi que des objets du monde réel et des concepts abstraits, c'est-à-dire, il englobe n'importe quel objet ou concept dans le monde. Autrement dit, ces URIs peuvent inclure des choses concrètes comme des lieux, des gens et des voitures, ou celles qui sont plus abstraites, telles que la couleur d'une voiture, l'ensemble des voitures marron dans le monde, etc.
Dans le web traditionnel, les URIs HTTP sont utilisées pour combiner une identification unique au monde (URI) qui identifie des documents HTML classiques avec un mécanisme de récupération simple (HTTP) afin qu'un client HTTP peut récupérer une description d'une ressource. Alors, le deuxième principe des données liées

9. http://linkeddata.org/

préconise l'utilisation des URIs HTTP pour identifier des objets et des concepts abstraits, permettant à ces URIs d'être déréférencées sur le protocole HTTP en une description de l'objet ou du concept identifié.
Dans le web classique, le langage HTML est largement utilisé pour traiter le contenu du web. Le troisième principe des données liées consiste à utiliser RDF comme un modèle de données unique pour la publication des données structurées sur le web. RDF sera expliqué plus en détails à la section suivante.
Le dernier principe des données liées préconise l'utilisation des hyperliens pour relier les données, les documents web et tout type de choses ; ces liens sont appelés des liens RDF (RDF links). Par exemple, on peut établir un lien entre une personne et un lieu ou une compagnie. Contrairement au web traditionnel ou les hyperliens entre les documents sont en grande partie non typés, les hyperliens qui relient les données à l'intérieur des documents ont des types qui décrivent la relation entre les choses. Par exemple, un lien de type "knows" peut être établis entre deux personnes. Nous verrons par la suite les différents types de liens RDF.

2.5 Resource Description Framework (RDF)

RDF (Klyne and Carroll, 2004) est un modèle de données basé sur la théorie des graphes, visant à représenter les informations sous forme de graphes dirigés avec des noeuds et des arcs étiquetés. Ce modèle de données est désigné pour faciliter l'intégration des sources de données hétérogènes et d'échanger des données sur le web de données. En RDF, une ressource est représentée comme un ensemble de triplets. Généralement, un triplet prend la structure d'une phrase simple, il se compose de trois parties : un sujet, un prédicat et un objet.
Par exemple : Bruxelles est la capitale de la Belgique.

Sujet	Prédicat	Objet

Le sujet est une ressource qui représente une information en utilisant une URI [10].
Une ressource est une chose abstraite ou physique, elle représente une personne, une ville, un concept, une compagnie, etc.
L'objet peut être soit un littéral (*rdf :literal*) (comme une chaîne de caractères, une date ou un nombre) ou une URI d'une autre ressource qui est reliée avec le sujet.
Le prédicat est une URI qui représente le type de la relation qui existe entre le sujet et l'objet, par exemple, le nom ou la date de naissance (dans le cas d'un littéral), ou le lieu de la naissance d'une personne (dans le cas d'une autre ressource).
Un exemple d'un triplet (en pseudo-code, ou l'objet est un littéral) pourrait être :

10. Uniform Resource Identifier, identifiant la ressource en question.

```
prefix db:<http://dbpedia.org/resource/>.
db:Tim_Berners-Lee <http://dbpedia.org/ontology/birthDate> 1955-06-08.
db:Tim_Berners-Lee <http://dbpedia.org/property/birthName> "Timothy John Berners-
Lee"@en.
```

De même, un exemple de triplet qui utilise une URI comme objet pourrait être :

```
prefix db:<http://dbpedia.org/resource/>.
db:Tim_Berners-Lee <http://dbpedia.org/ontology/birthPlace> db:England.
db:Tim_Berners-Lee <http://dbpedia.org/property/parents> db:Conway_Berners-Lee.
```

En RDF, on peut distinguer entre les individus (objets) et les propriétés (relations) en utilisant deux mots-clés : *rdf : type* et *rdf :Property*.
En effet, les triplets RDF peuvent être représentés soit d'une façon tabulaire (une table de triplet, voir le tableau 2.1) ou comme un graphe RDF (la figure 2.2 illustre une visualisation graphique des triplets donnés en exemple ci -dessus).

Sujet	Prédicat	Objet
http ://dbpedia.org/resource /Tim_Berners-Lee	http ://dbpedia.org/ontology /birthDate	1955-06-08
http ://dbpedia.org/resource /Tim_Berners-Lee	http ://dbpedia.org/property /birthName	“Timothy John Berners-Lee”
http ://dbpedia.org/resource /Tim_Berners-Lee	http ://dbpedia.org/ontology /birthPlace	http ://dbpedia.org/resource /England
http ://dbpedia.org/resource /Tim_Berners-Lee	http ://dbpedia.org/property /parents	http ://dbpedia.org/resource /Conway_Berners-Lee

Table 2.1 – Visualisation des triplets sous forme d'un tableau

Représenter un ensemble de triplets comme un graphe dirigé est très pratique pour visualiser toutes les informations qui sont reliées avec un individu. Dans un tel graphe, chaque triplet est représenté comme une arête de son sujet à son objet.
Dans un graphe RDF, on peut utiliser des noeuds vides pour capturer une certaine forme d'individus inconnus. Un noeud blanc (ou une ressource anonyme) est un sujet ou un objet dans un triplet RDF qui n'est pas ni un littéral ni une URI. L'exemple suivant présente un graphe RDF avec des noeuds anonymes.
:Pierre foaf :knows _ :p
_ :p foaf :name "John Smith"

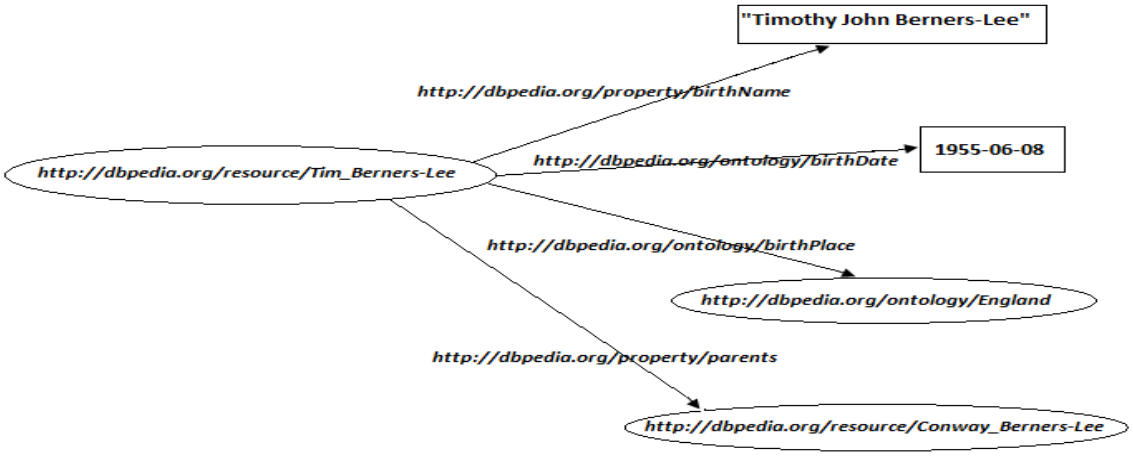

Figure 2.2 – Représentation graphique d'un ensemble de triplets.

_ :p wrote _ :b
_ :b dc :title "Introduction to Java"

Les prédicats *foaf :knows* et *foaf :name* appartiennent au vocabulaire FOAF. Le prédicat *dc :title* appartient au vocabulaire Dublin Core [11].

Pour publier un graphe RDF sur le web, il doit être sérialisé en utilisant une syntaxe RDF. La sérialisation est le processus d'encodage d'une information sous forme d'octets ou de bits, permettant ainsi un stockage sur un disque ou un transfert sur le réseau. Le W3C a formalisé deux formats RDF : RDF/XML et RDFa.

RDF/XML (Beckett, 2004) est la syntaxe la plus utilisée pour publier les données liées sur le web. Cependant, elle est très difficile à lire ou écrire pour les humains mais facilement parsable par une machine, car elle est basée sur le langage XML. Par conséquent, il est recommandé d'utiliser d'autres sérialisations pour la gestion des données. Le pseudo-code ci-dessous montre la sérialisation RDF/XML de deux triplets RDF.

Le premier triplet affirme qu'il existe une chose, identifiée par l'URI <http ://dbpedia. org/resource/Tim_Berners-Lee> de type personne. Le deuxième triplet déclare que cette chose a le nom "Timothy John Berners-Lee".

RDFa (Adida and Birbeck, 2008) est un format de sérialisation qui intègre des triplets RDF dans des pages HTML. Les données RDF ne sont pas intégrées dans les commentaires dans un document HTML, comme ce fut le cas avec quelques premières tentatives pour mélanger RDF et HTML. Donc, si on utilise RDFa pour publier des données liées sur le web, il est important de distinguer entre les objets du monde réel qui sont décrites par les données et le document HTML + RDFa qui

11. http://dublincore.org

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">
<rdf:Description rdf:about="http://dbpedia.org/resource/Tim_Berners-Lee">
<rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
<foaf:name> Timothy John Berners-Lee </foaf:name>
</rdf:Description>
</rdf:RDF>
```

incarne ces descriptions.
Le pseudo-code ci-dessous montre la sérialisation RDFa qui permet d'afficher les triplets RDF décrites dans l'exemple précédent dans une page HTML.

```
<!DOCTYPE html PUBLIC " -//W3C//DTD XHTML+RDFa 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">
<head>
<meta http-equiv="Content-Type" content="application/xhtml+xml ;
charset=UTF-8"/>
<title>Profile Page for Tim Berners-Lee</title>
</head>
<body>
<div about="http://dbpedia.org/resource/Tim_Berners-Lee" typeof="foaf:Person">
<span property="foaf:name"> Timothy John Berners-Lee </span>
</div>
</body>
</html>
```

Turtle (Beckett and Berners-Lee, 2008) est un format de texte brut pour la sérialisation des données RDF, elle est beaucoup plus facile à écrire et interpréter par un humain que RDF/XML parce qu'elle s'appuis sur les préfixes des espaces de noms. Le pseudo-code ci-dessous montre la sérialisation Turtle de deux triplets RDF.

N-Triples est un sous-ensemble de la syntaxe Turtle, mais elle a moins de fonctionnalités telles que les préfixes des espaces de noms. Par conséquent, un fichier RDF en cette sérialisation peut avoir beaucoup de redondance, parce que toutes les

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
prefix foaf: <http://xmlns.com/foaf/0.1/>.
<http://dbpedia.org/resource/Tim_Berners-Lee>
foaf:type foaf:Person;
foaf:name "Timothy John Berners-Lee".
```

URIs doivent être spécifiées en plein dans chaque triplets. Le pseudo-code ci-dessous montre un exemple d'une sérialisation N-Triples de deux triplets RDF.

```
<http://dbpedia.org/resource/Tim_Berners-Lee>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://xmlns.com/foaf/0.1/Person>.
<http://dbpedia.org/resource/Tim_Berners-Lee> <<http://xmlns.com/foaf/0.1/name>
"Timothy John Berners-Lee".
```

RDF/JSON est une sérialisation hautement souhaitable, parce qu'un nombre croissant de langages de programmation fournissent un soutien JSON natif, y compris des langages de programmation web tels que JavaScript et PHP. Le pseudo-code ci-dessous illustre un exemple d'une sérialisation RDF/JSON.

```
{
<http://dbpedia.org/resource/Tim_Berners-Lee> :{
<http://xmlns.com/foaf/0.1/name>: [{"value": "Timothy John Berners-Lee",
                                    "type": "literal"}]
```

2.6 RDF Schema (RDFS)

Généralement, les ontologies sont utilisées pour définir la sémantique des classes et des prédicats. En effet, cette sémantique permet de créer des règles d'inférences et par conséquent générer un grand nombre de triplets et d'information qui peuvent augmenter la capacité de raisonnement par les machines.
Dans ce contexte, RDF Schema (Brickley and Guha, 2004) est un langage qui vise à représenter des vocabulaires RDF simples sur le web de données liées. Les déclarations RDFS peuvent être exprimées comme des triplets RDF en utilisant le mot

clé RDFS.
RDFS permet de spécifier un certain nombre de contraintes utiles sur les individus et les relations utilisées dans les triplets RDF. En particulier, il permet de déclarer des objets et des sujets comme des instances de certaines classes.
En outre, RDFS peut déclarer des relations d'inclusions entre les classes et les propriétés. Il fournit un nouveau prédicat *rdfs : subClassOf* pour spécifier qu'une classe est une sous-classe d'une autre. Les relations de subsomptions entre les classes peuvent être déclarées comme un ensemble de triplets de la forme *<A rdfs : subClassOf B>*. Le pseudo-code ci-dessous présente un exemple d'une relation de subsomption.

```
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.
<http://xmlns.com/foaf/0.1/Organization> rdfs:subClassOf
<http://xmlns.com/foaf/0.1/Agent>.
```

Cela veut dire que la classe "Organisation" est une sous classe de "Agent". Ainsi, si x est de type A et A est une sous classe de B, alors x est une sous classe de B.
De même, RDFS fournit le prédicat *rdfs :subPropertyOf* pour exprimer les relations structurelles entre les propriétés. Nous pourrions dire par l'exemple en dessous : que la relation *deathPlace* est plus spécifique que la relation *hasLocation*.

```
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.
<http://dbpedia.org/ontology/deathPlace> rdfs:subPropertyOf
<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#hasLocation>.
```

Enfin, RDFS peut exprimer des relations sémantiques entre le "domaine" et la "portée" d'une propriété à certaines classes. Une propriété P (entre un sujet et un ou plusieurs objets) peut être considérée comme une fonction qui relie un sujet s avec un ensemble d'objets liés à s via P. Donc, l'ensemble de sujets d'une propriété P est appelé : son domaine (Domain), et l'ensemble d'objets de P est appelé : son Co-domaine (Range). La restriction du domaine et du Co-domaine d'une propriété est également possible dans RDFS en utilisant deux autres prédicats : *rdfs :domain* et *rdfs :range*. Ces triplets sont de la forme : *<P rdfs :domain d>* et *<P rdfs :range r>*.
Le pseudo-code ci-dessous illustre un exemple de déclaration des prédicats *rdfs :domain* et *rdfs :range*.

```
prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.
<http://xmlns.com/foaf/0.1/skypeID> rdfs:domain <http://xmlns.com/foaf/0.1/Agent>.
<http://xmlns.com/foaf/0.1/skypeID> rdfs:range rdf:literal.
```

Les déclarations RDFS peuvent être interprétées par des formules de la logique du premier ordre FOL (Abiteboul et al., 2011). Pour rappel, la logique du premier ordre FOL (aussi appelé la logique des prédicats) est la base formelle du langage d'ontologies OWL. En particulier, elle est appropriée pour la représentation des connaissances et de raisonnement.
Le tableau 2.2 donne la sémantique logique des contraintes RDF et RDFS en donnant leurs traductions FOL et DL (logique de description) correspondantes.

Contraintes RDF et RDFS	FOL	DL
i rdf :type C	$C(i)$	$I : C$ ou $C(i)$
$I\ P\ j$	$P(i,j)$	$I\ P\ j$ ou $P(i,j)$
C_1 rdfs :subclassOf C_2	$\forall x(C_1(x) \Rightarrow C_2(x))$	$C_1 \subseteq C_2$
P_1 rdfs :subpropertyOf P_2	$\forall x, \forall y(P_1(x,y) \Rightarrow P_2(x,y))$	$P_1 \subseteq P_2$
P rdfs :domain D	$\forall x, \forall y(P(x,y) \Rightarrow D(x))$	$\exists P \subseteq D$
P rdfs :range R	$\forall x, \forall y(P(x,y) \Rightarrow R(x))$	$\exists P \subseteq R$

Table 2.2 – La sémantique logique de RDF et RDFS (Abiteboul et al., 2011)

Ces contraintes sont très utiles dans la pratique et très adaptées pour faire des inférences. Intuitivement, chaque règle peut générer de nouveaux faits : à partir de la partie gauche de la flèche, nous pouvons déduire la partie droite. Ainsi, nous pouvons construire un ensemble de patterns à partir de ces inférences. Par exemple, le pattern *skypeID(X,Y)* ou X et Y sont des variables. Une évaluation v relie X au *John* et Y au *2439*. Donc, il transforme le pattern *skypeID(X,Y)* au fait *skypeID(John, 2439)*. Les contraintes RDFS sont exploitées pour saturer les triplets RDF en ajoutant les triplets qui peuvent être déduites par des règles. Ensuite, l'ensemble de triplets RDF résultant peut être interrogé avec le langage d'interrogation SPARQL. Plus d'informations sur le langage SPARQL seront données à la section 8.

2.7 Web Ontology Language (OWL)

Comme indiqué précédemment, le vocabulaire RDFS permet d'ajouter une sémantique aux ressources. En effet, il existe des vocabulaires qui offrent plus de

Contraintes OWL	FOL	DL
C_1 owl :disjointWith C_2	$\forall x(C_1(x) \Rightarrow \neg C_2(x))$	$C_1 \subseteq \neg C_2$
P rdf :type owl :FunctionalProperty	$\forall x, \forall y, \forall z(P(x,y) \wedge P(x,z) \Rightarrow y = z$	$FunctP$ ou $\exists P \subseteq (\leq 1P)$
P rdf :type owl :InverseFunctional-Property	$\forall x, \forall y, \forall z(P(x,y) \wedge P(x,z) \Rightarrow x = z$	$FunctP^-$ ou $\exists P^- \subseteq (\leq 1P^-)$
P owl :inverseOf Q	$\forall x, \forall y(P(x,y) \Leftrightarrow Q(x,y))$	$P \equiv Q^-$
P rdf :type owl :SymmetricProperty	$\forall x, \forall y(P(x,y) \Leftrightarrow P(y,x))$	$P \subseteq P^-$
_ :a owl :onProperty P _ :a owl :allValuesFrom C	$\forall y(P(x,y) \Rightarrow C(y))$	$\forall P.C$
_ :a owl :onProperty P _ :a owl :someValuesFrom C	$\exists y(P(x,y) \wedge C(y))$	$\exists P.C$
C_1 owl :intersectionOf C_2	$C_1(x) \wedge C_2(x)$	$C_1 \cap C_2$
C_1 owl :unionOf C_2	$C_1(x) \vee C_2(x)$	$C_1 \cup C_2$

Table 2.3 – La sémantique logique des contraintes OWL (Group, 2009)

possibilités d'inférence.

OWL (Group, 2009) est un langage d'ontologie plus riche que RDFS et largement utilisé pour ajouter une sémantique aux données. Comme RDFS, les contraintes OWL peuvent être exprimées en triplets RDF en utilisant le mot clé OWL. OWL offre un certain nombre de contraintes sémantiques riches, à savoir la disjonction des classes, la restriction fonctionnelle, la définition de la classe intentionnelle, l'union et l'intersection des classes, etc. Le tableau 2.3 présente la sémantique logique des contraintes OWL et leurs traductions FOL et DL correspondantes.

OWL fournit le prédicat *owl : disjointWith* pour exprimer que deux classes C_1 et C_2 sont disjointes en utilisant le triplet : $<C_1$ *owl :disjointWith* $C_2>$, cette contrainte ne peut pas être exprimée en RDFS. Par exemple, nous pouvons déclarer le triplet ci-dessous pour dire que les classes "document" et "organisation" sont disjointes.

```
prefix owl:< http://www.w3.org/2002/07/owl#>.
prefix foaf: <http://xmlns.com/foaf/0.1/>.
foaf :Document owl:disjointWith foaf :Organization.
```

Le prédicat *owl :disjointWith* ne peut pas être utilisé pour produire de nouveaux faits, mais pour exclure les ressources incompatibles avec ce que nous savons du domaine d'intérêt.

En effet, OWL permet de déclarer une relation entre A et B qui est en fait une

fonction de A à B. Il permet également d'indiquer qu'une propriété est l'inverse d'une autre, ou qu'elle est symétrique.
Le prédicat *owl :FunctionalProperty* exprime que le premier attribut de la propriété P est une clé, tandis que *owl :InverseFunctionalProperty* exprime que le deuxième attribut est une clé. On note qu'une propriété peut être à la fois un *owl :FunctionalProperty* et *owl :InverseFunctionalProperty*.
La définition intentionnelle de nouvelles classes est une caractéristique principale du langage OWL, elle permet d'exprimer des contraintes complexes. Prenant comme exemple : un département dispose d'un gestionnaire unique, qui est professeur. Le mot clé *owl :Restriction* est associé à une classe de noeud anonyme et quelques propriétés de restrictions spécifiques (*owl :someValuesFrom*, *owl :allValuesFrom*, *owl :minCardinality*, *owl : maxCardinality*) qui sont utilisées pour la définition de la nouvelle classe. Le noeud anonyme est nécessaire parce que l'expression de chaque restriction exige un ensemble de triplets qui sont tous liés à la même description de classe.
Les restrictions *owl :intersectionOf* et *owl :union* permettent de combiner les classes. L'intersection de classes désigne les individus qui appartiennent aux deux classes ; alors que l'union représente les individus qui appartiennent au moins à une de deux classes.
La restriction *owl :equivalentClass* indique que deux classes sont équivalentes, c'est à dire, il existe des inclusions dans les deux sens. De même, *owl :equivalentProperty* indique que deux propriétés sont équivalentes. En fait, $<C_1$ *owl :equivalentClass* $C_2>$ peut être exprimée en RDFS par les deux triplets : $<C_1$ *rdfs :subClassOf* $C_2>$ et $<C_2$ *rdfs :subClassOf* $C_1>$.
La déclaration *owl :sameAs* indique que deux références URIs se réfèrent à la même chose, c'est-à-dire, les individus ont la même "identité". Par exemple, les individus *William_Jefferson_Clinton* et *BillClinton* se réfèrent à la même personne.

2.8 SPARQL

Stocker de grandes quantités de données liées n'aurait aucune utilité si l'accès à ces données est impossible. Donc, il est nécessaire d'avoir un langage d'interrogation pour accéder aux sources de données RDF. En 2008, le groupe de travail DAWG (RDF Data Access Working Group) du W3C (Consortium World Wide Web) proposait le langage d'interrogation SPARQL (Prud'hommeaux and Seaborne, 2008) qui est largement utilisé pour l'interrogation des données RDF. SPARQL est un acronyme récursif pour Simple Protocol And RDF Query Language.

Un BGP (Basic Graph Pattern) est le composant de base d'une requête SPARQL, il est composé d'un ensemble de TP (Triple Pattern). Ce dernier est adapté à la structure d'un triplet RDF ; il peut contenir un ou plusieurs sujets, prédicats ou objets (Hartig and Langegger, 2010). Dans une requête SPARQL, chaque variable commence par un point d'interrogation *?*.
En effet, nous pouvons exprimer des requêtes constructives ou interrogatives :
Une requête constructive permet d'engendrer un nouveau graphe à partir d'un graphe RDF qui existe déjà en utilisant le mot clé *CONSTRUCT*. Par exemple, nous pouvons ajouter des relations frère-soeur à un graphe RDF qui contient des informations généalogiques.
Dans une requête interrogative, nous utilisons le mot clé *SELECT* pour extraire un sous graphe (un ensemble de ressources) à partir d'un graphe RDF en vérifiant les contraintes qui sont définies dans la clause *WHERE*. Le pseudo-code ci-dessous présente un exemple d'une requête SPARQL.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?nom
WHERE {
        ?personne rdf:tvpe foaf:Person.
        ?personne foaf:name ?nom.
}
```

Les espaces de noms sont déclarés au début de la requête. Les variables *nom* et *personne* sont précédées par un point d'interrogation *?*. La ligne *SELECT* permet de sélectionner l'ensemble de tuples correspondants aux conditions de la clause *WHERE*.
Le premier TP de la requête est : *?personne rdf :type foaf :Person*. Il introduit une première variable *?personne* qui est de type *foaf :Person*. Le deuxième TP est *?personne foaf :name ?nom* introduisant une nouvelle variable *?nom* qui représente l'objet du triplet. Finalement, le résultat de cette requête est la liste de noms des personnes d'une base de données RDF qui utilise le vocabulaire FOAF. Pour rappel, FOAF est le vocabulaire le plus utilisé pour décrire les personnes et les liens entre elles. Le résultat de cette requête SPARQL peut être exprimé sous forme d'un tableau, ou sous forme d'un sous graphe RDF.
Un point d'accès SPARQL (SPARQL endpoint) est un service web d'interrogation basé sur le protocole HTTP (Gewelt, 2012). Il fournit une interface qui prend en

charge l'exécution d'une requête SPARQL et renvoi les résultats à l'utilisateur (soit un humain ou autre). Les SPARQL endpoint les plus connus sont : DBpedia [12] et OpenLink Virtuoso [13] .

2.9 Types de liens de Linked Data

Comme décrit ci-dessus, le quatrième principe des données liées permet de la mise en place des liens RDF pointant vers d'autres sources de données sur le web. En effet, un lien RDF décrit la relation entre deux ressources. Une distinction utile peut être faite entre les liens RDF internes et externes. Un lien RDF interne relie les ressources dans un seul ensemble de données liées. Donc, les espaces de noms des URIs du sujet et d'objet sont les mêmes.
Techniquement, un lien RDF externe est un triplet RDF dans lequel le sujet du triplet est une référence URI dans l'espace de nom d'un ensemble de données, tandis que le prédicat ou l'objet du triplet sont des références URIs pointant vers des espaces de noms d'autres ensembles de données.
Le déréférencement des URIs donne une description d'une ressource donnée. Cette description contiendra habituellement des liens RDF supplémentaires qui pointent vers d'autres URIs qui peuvent également être déréférencées et ainsi de suite. Les liens externes sont cruciaux parce qu'elles relient les données qui existent dans plusieurs ensembles de données liées et par conséquent elles rendent le web comme un espace de données global.
En réalité, il existe trois types de liens RDF : des liens relationnels, des liens d'identités et des liens de vocabulaires.

2.9.1 Liens relationnels

Le web de données contient des informations sur une multitude de choses allant des personnes, des entreprises et des lieux, vers des films, de la musique, des livres, des gènes et autres types de données (Heath and Bizer, 2011). Les liens relationnels permettent de relier des données à l'intérieur d'un ou de plusieurs ensembles de données liées. Ce type de liens pointe vers des choses connexes à une connaissance dans d'autres sources de données. Cela peut construire un réseau de données potentiellement infini qui peut être utilisé par des applications clientes.

Le pseudo-code ci-dessous montre comment les liens RDF peuvent connecter des entités pour enrichir les données publiées à propos de l'inventeur du web *Tim*

12. https://dbpedia.org/sparql
13. http://lod.openlinksw.com/sparql/

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix foaf: <http://xmlns.com/foaf/0.1/>
<http://dbpedia.org/resource/Tim_Berners-Lee>
rdf:type   foaf:Person
<http://dbpedia.org/property/birthPlace>  <http://dbpedia.org/resource/England>
<http://dbpedia.org/ontology/birthDate> 1955-06-08
<http://dbpedia.org/property/birthName> "Timothy John Berners-Lee"
<http://dbpedia.org/ontology/wikiPageExternalLink>
<http://news.bbc.co.uk/2/hi/technology/4132752.stm>
```

Berners-Lee. Cet exemple contient un lien RDF indiquant que *l'Angleterre* est le lieu de naissance de *Tim*. De plus, il contient un lien externe <http ://dbpedia.org/ontology/wikiPageExternalLink> pointant vers la ressource <http ://news.bbc.co.uk/2/hi/technology/4132752.stm> de l'ensemble de données BBC news [14] . Ainsi, les applications des données liées qui voient cette URI peuvent récupérer une autre description de *Tim* à partir de BBC news qu'il peut contenir, à son tour, d'autres liens RDF qui sont en relation avec <http ://news.bbc.co.uk/2/hi/technology/4132752.stm>.

2.9.2 Liens d'identité

Dans le web de données, plusieurs fournisseurs de données parlent des mêmes entités. Comme ils utilisent leurs propres URIs pour désigner une personne ou un lieu, le résultat sera plusieurs et différentes URIs identifiant la même entité. Ces URIs sont appelés "alias URIs" (Heath and Bizer, 2011). En fait, les éditeurs des données liées utilisent le lien typé http ://www.w3.org/2002/07/owl#sameAs pour déclarer que deux alias URIs se rapportent à la même ressource.

```
prefix owl: < http://www.w3.org/2002/07/owl#>
<http://dbpedia.org/resource/Tim_Berners-Lee> owl:sameAs <http://yago-
knowledge.org/resource/Tim_Berners-Lee>
```

Le pseudo-code ci-dessus présente deux URIs qui se réfèrent à la même ressource. Donc, *Tim* peut ajouter un lien http ://www.w3.org/2002/07/owl#sameAs à sa page d'ac-

14. www.bbc.co.uk

cueil dans l'ensemble de données DBpedia. Ce lien indique que l'URI utilisée pour faire référence à *Tim* dans DBpedia <http ://dbpedia.org/resource/Tim_Berners-Lee> et l'URI utilisée par YAGO <http ://yago-knowledge.org/resource/Tim_Berners-Lee> se réfèrent au même objet du monde réel.
Actuellement, l'établissement des liens http ://www.w3.org/2002/07/owl#sameAs est indispensable pour rendre le web de données comme un système social pour les raisons suivantes :

— Les alias URIs sont déréférencés aux descriptions de la même ressource fournies par différents éditeurs de données et permettent d'exprimer différents opinions et points de vue.

— L'utilisation de différents URIs permet aux consommateurs des données liées de connaitre ce qu'un éditeur veut dire à propos d'une entité spécifique.

— Si toutes les choses dans le monde devaient avoir une seule URI, cela implique la création et le fonctionnement d'une autorité de nommage centralisé pour attribuer les URIs. En outre, cela créera un obstacle majeur à la croissance du web de données.

Le web de données liées repose sur la résolution du problème de la déduplication des entités d'une façon évolutive (une quantité énorme de liens *owl :sameAs* peut être ajoutée au cours du temps) et distribuée (Comme les liens *owl :sameAs* sont publiés par différents fournisseurs de données, l'effort global pour la création de ces liens peut être partagé entre les différentes parties).
En plus du lien *owl :sameAs*, il existe plusieurs prédicats qui permettent d'exprimer des liens d'identité entre les entités du web de données qui se réfèrent au même objet du monde réel (*rdfs :seeAlso*, *skos :closeMatch*, *skos :exactMatch*, *skos :relatedMatch*) (Gandon, 2015). Aujourd'hui, *owl :sameAs* est largement utilisé dans le contexte de données liées et des centaines de millions de liens *owl :sameAs* sont publiés sur le web.

2.9.3 Liens de vocabulaires

Le web de données permet aux applications clientes de découvrir de nouvelles sources de données en suivant les liens RDF. En outre, il aide aussi à intégrer les données provenant de ces sources. L'intégration des données vise à fournir une vue unifiée des schémas qui sont utilisés par différentes sources de données pour publier leurs données. Dans le contexte des données liées, le terme schéma signifie le mélange des termes distincts de plusieurs vocabulaires RDF qui sont utilisés par une source de données pour publier des données sur le web. En effet, il existe deux façons pour

traiter l'hétérogénéité des données dans le web de données. La première consiste à réutiliser des termes des vocabulaires largement déployés (comme : FOAF, SIOC, Dublin Core, etc). Ainsi, cela permet d'éviter l'hétérogénéité des ontologies de données liées.
La deuxième façon consiste à rendre les données auto-descriptives (Mendelsohn, 2009), ce qui signifie qu'une application Linked Data (qui vise à découvrir des données ou le vocabulaire est inconnu) devrait être en mesure de trouver toutes les méta-informations dont elle a besoin pour traduire les données en une représentation compréhensible et traitable.
Techniquement, cela est réalisable : 1) en faisant les URIs (qui identifient les termes de vocabulaires) déréferenceables de sorte que les applications clientes peuvent identifier la définition des termes RDFS et OWL (Berrueta and Phipps, 2008), ou 2) en publiant des mappings entre les termes de différents vocabulaires sous forme de liens RDF (Mendelsohn, 2009).
Par conséquent, les éditeurs des données liées devraient adopter la procédure suivante : dans un premier temps, rechercher des termes de vocabulaires largement utilisés qui pourraient être réutilisés pour représenter les données. Si ces vocabulaires ne fournissent pas tous les termes qui sont nécessaires pour publier le contenu complet d'un ensemble de données liées, les conditions requises doivent être définies comme un vocabulaire propriétaire et utilisées en plus de termes de vocabulaires largement déployés.
À un moment donné, si un éditeur de données découvre qu'un autre vocabulaire utilise le même terme que le vocabulaire propriétaire, un lien RDF devrait être établi entre les URIs identifiant les deux termes des vocabulaires, indiquant que ces URIs se réfèrent au même concept. OWL (McGuinness and Harmelen, 2004), RDFS (Brickley and Guha, 2004) et Skos (Miles and Bechhofer, 2009) ont définis des liens RDF qui peuvent être utilisés pour représenter ces mappings. Par exemple, nous pouvons utiliser *owl :equivalentClass* et *owl :equivalentProperty* pour déclarer que deux différents termes sont équivalents. De plus, les termes de différents vocabulaires peuvent être reliés par d'autres relations comme : *rdfs :subClassOf*, *rdfs :subPropertyOf*, *SKOS : broadMatch*, et *skos :narrowMatch*.
Le pseudo-code ci-dessous illustre des relations entre des termes du vocabulaire FOAF et d'autres vocabulaires.

Comme il est possible de définir des liens *owl :sameAs* entre les alias URIs, nous pouvons également définir des liens entre les termes des vocabulaires. La déclaration de beaucoup de liens entre les termes des vocabulaires permet de faire une meilleure intégration des données et donc faciliter l'accès à ces données par des applications

```
prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.
prefix owl:< http://www.w3.org/2002/07/owl#>.
<http://xmlns.com/foaf/0.1/Agent>
rdfs:label "Agent";
rdfs:subClassOf  <http://xmlns.com/wordnet/1.6/Agent-3>;
owl:equivalentClass <http://purl.org/dc/terms/Agent>.
```

clientes.
Comme conclusion ces trois types de liens sont indispensables pour construire un espace de données global- qui est le web de données.

2.10 Projet Linked Open Data

Un nombre important de personnes et d'organisations ont adopté les données liées comme un moyen pour publier leurs données sur le web. Linked Open Data Cloud (nommé LOD Cloud) est un projet qui permet de publier une quantité énorme de données structurées sur le web de données, selon les principes des données liées (Alexander et al., 2009), (Bizer et al., 2007b). Il est constitué de milliards de triplets RDF à partir de nombreuses sources couvrant toutes sortes de sujets, à savoir : des emplacements géographiques, des personnes, des entreprises, des livres, des publications scientifiques, des films, de la musique, des programmes de télévision et de radio, des gènes, des protéines, des médicaments et des essais cliniques, des données statistiques, des communautés en ligne,etc. Ce projet vise à établir des connections entre les ensembles de données afin d'obtenir un web de données qui est grand et unique. En effet, les sources de données qui existent dans le LOD Cloud sont classées dans les domaines thématiques suivants : géographique, gouvernement, médias, bibliothèques, sciences de la vie, commerce, le contenu généré par l'utilisateur et des ensembles de données inter-domaines.
Par exemple, toutes les sources qui contiennent des médias peuvent être regroupées en un cluster propre à ce thème, toutes les sources qui contiennent des informations géographiques peuvent être regroupées dans un cluster propre à ce thème, etc. Dans la figure 2.4, nous remarquons que les sources qui appartiennent au même cluster ont la même couleur du fond.

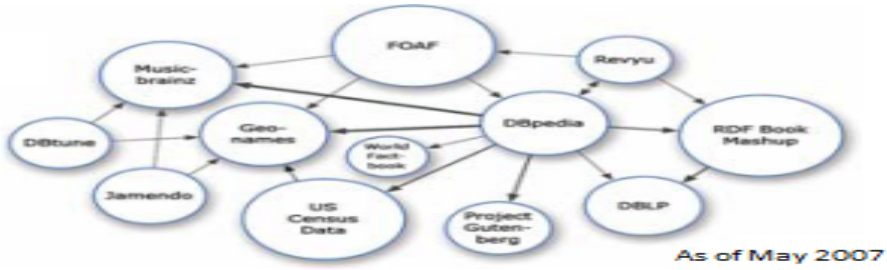

Figure 2.3 – L'état du LOD Cloud en 2007.

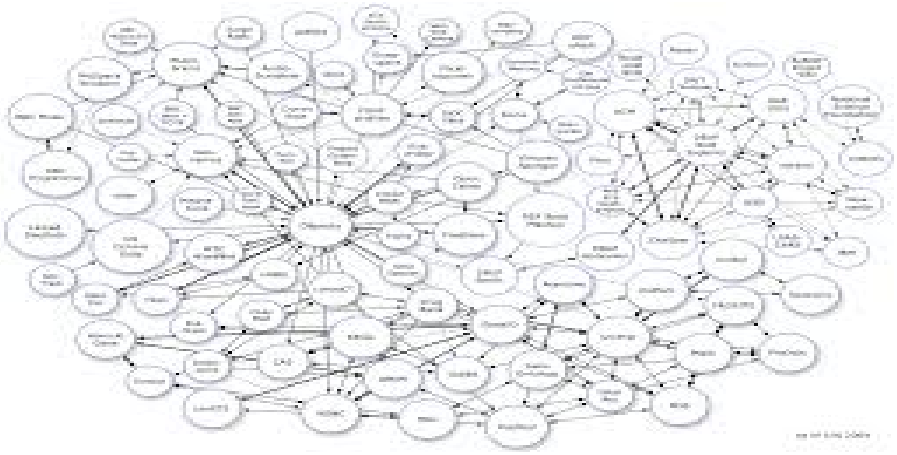

Figure 2.4 – L'état du LOD Cloud en 2009.

Le projet LOD Cloud a connu une croissance exponentielle du nombre de sources de données inter-connectées. Les figures 2.3, 2.4, 2.5, 2.6 illustrent les sources de données dans le LOD Cloud en 2007, 2009, 2011 et 2016, respectivement. En 2007, le LOD Cloud a contenu 12 ensembles de données, plus d'un milliard de triplets RDF et environ de 120.000 liens RDF entre les sources de données (Heath and Bizer, 2011). Ensuite, il a évolué à 295 ensembles de données, plus de 31 milliards de triplets RDF et environ de 504 millions liens RDF entre les sources de données [15] en 2011. Le nombre d'ensembles de données dans le LOD Cloud est devenu 9960 en 2016, plus de 176 milliards de triplets RDF et environ de 41 milliards liens RDF entre les sources de données [16].

Les ensembles de données de domaines multiples (Cross-domain) :

Les premiers ensembles de données liées qui sont apparus dans le LOD Cloud n'ont pas un sujet spécifique, ils couvrent plusieurs domaines. DBpedia [17] (Bizer

15. http://lod-cloud.net/state/
16. http://stats.lod2.eu/stats
17. http://dbpedia.org/

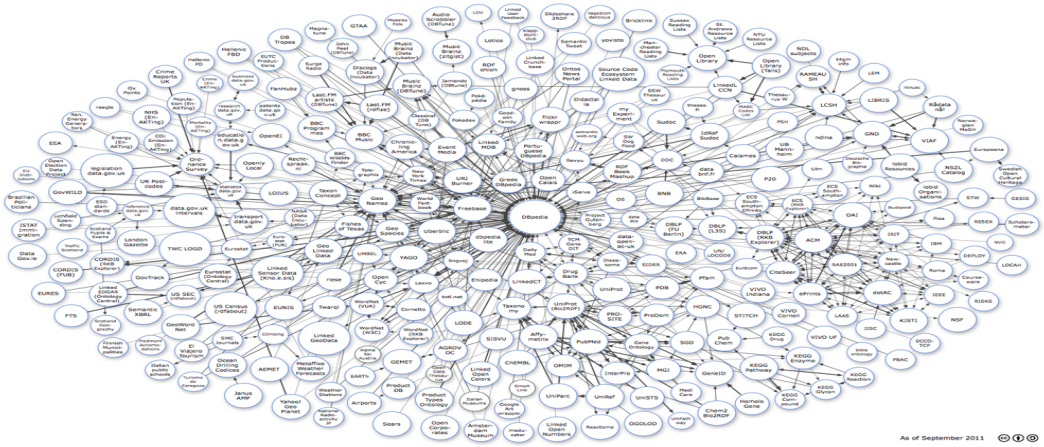

Figure 2.5 – L'état du LOD Cloud en 2011.

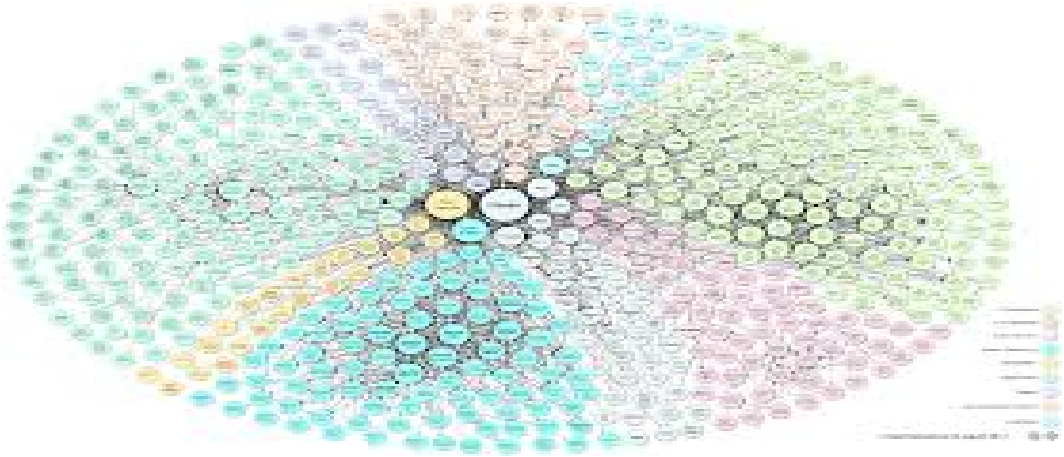

Figure 2.6 – L'état du LOD Cloud en 2016.

et al., 2009) est l'ensemble de données liées le plus connu qui couvre plusieurs domaines. Les données de DBpedia sont extraites automatiquement à partir des dumps publiques de Wikipédia. Par exemple, l'article Wikipédia qui a comme URI <https ://fr.wikipedia.org/wiki/Tim_Berners-Lee> est transformé en une URI dans DBpedia <http ://dbpedia.org/resource/Tim_Berners-Lee> qui n'est pas une URI d'une page web, mais une URI qui identifie la personne *Tim_Berners-Lee* elle-même. DBpedia est considéré comme une source de données centrale dans le LOD Cloud grâce à sa large couverture de différents domaines.

Freebase [18] est la deuxième source importante de données liées de cross-domain. Cet ensemble de données liées importe des données provenant de plusieurs sources telles que Wikipedia et Geonames. Freebase fournit des descriptions RDF des éléments

18. http://www.freebase.com

qui sont liés à des éléments dans DBpedia avec des liens entrants et sortants.
D'autres ensembles de données inter-domaines comprennent UMBEL [19] , YAGO (M.Suchanek et al., 2007) et OpenCyc liées [20] sont liées à DBpedia pour faciliter l'intégration des données.

Les ensembles de données géographiques :

La géographie est un autre facteur qui permet de connecter des informations à partir de plusieurs domaines. L'ensemble de données Geoname [21] est considéré comme un hub des ensembles de données qui ont des informations géographiques. Il a publié des données liées d'environ 8 millions emplacements.
LinkedGeoData (Hellmann et al., 2009) est le second ensemble de données liées dans le domaine géographique. Il a fourni plus de 350 millions informations spatiales. Les emplacements de Geoname et LinkedGeoData sont reliés avec des emplacements correspondants dans DBpedia pour fournir un espace de données inter-connectées.

Les ensembles de données médias :

BBC (British Broadcasting Corporation) est une grande organisation qui adopte des principes et des technologies pour publier des données liées et gérer leurs contenus. En 2008, BBC avait mis en ligne deux grands sites qui combinent des données liées et des pages web classiques. Le premier site [22] fournit une URI et une description RDF pour chaque épisode de chaque programme de télévision ou de radio diffusé à travers différents canaux de BBC (Kobilarov et al., 2009). Le deuxième site [23] publie des données liées à propos de tous les artistes dont la musique a été jouée sur les stations de radio de BBC, y compris les liens entrants de l'épisode du programme spécifique au cours de diffusion. L'ensemble de données liées BBC est relié avec DBpedia et il reçoit des liens entrants à partir des sources de données liées à la musique.

Les ensembles de données gouvernementaux :

Les organismes gouvernementaux et les organisations du secteur public produisent une multitude de données, allant des statistiques économiques, aux registres des sociétés et la propriété des terres, des rapports sur la performance des écoles, des

19. http://umbel.org/
20. http://sw.opencyc.org/
21. http://www.geonames.org/
22. http://www.bbc.co.uk/programmes
23. http://www.bbc.co.uk/music

statistiques sur la criminalité et les dossiers de vote des représentants élus. Les initiatives data.gov.uk [24] et data.gov [25] ont publié des volumes importants de données en RDF pour faciliter l'accès aux données du gouvernement. La première initiative a mis l'accent sur la création de l'infrastructure de base au niveau de données pour la publication des données liées, telles que des URIs stables auxquelles des quantités croissantes de données peuvent être connectées (Sheridan and Tennison, 2010). La deuxième initiative a converti de très gros volumes de données pour les publier comme des données liées.

Les ensembles de données éducationnelles :

Les bibliothèques sont des compléments naturels aux données liées parce qu'elles fournissent des données structurées de haute qualité. Ce domaine a connu des développements importants qui visent à intégrer les catalogues des bibliothèques à l'échelle mondiale, l'interconnexion du contenu de plusieurs catalogues de bibliothèques et l'interconnexion des catalogues de bibliothèques avec d'autres informations (comme des photos et vidéos, ou des bases de connaissances comme DBpedia) pour faciliter l'accès à ces données. Par exemple, la Bibliothèque du Congrès américain [26] et la Bibliothèque nationale allemande d'économie (Neubert, 2009) permettent de publier des taxonomies suivant les principes des données liées.
Les articles de revues et de conférences sont également représentés dans le web des données grâce à des efforts de publications telles que DBLP [27], RKBexplorer [28] et le web sémantique Dogfood Server [29] (Möller et al., 2007).

Les ensembles de données de science de la vie :

Dans le projet LOD Cloud, il existe plusieurs ensembles de données liées qui permettent de connecter des données qui sont utilisées par des chercheurs dans le domaine de la science de la vie. En particulier, le projet Bio2RDF (Belleau et al., 2008) est en relation avec plus de 30 ensembles de données largement utilisées, y compris UniProt (Universal Protein Resource), KEGG (Kyoto Encyclopedia of Genes and Genomes), CAS (Chemical Abstracts Service), PubMed et l'ontologie Gene.
Dans le cadre du projet Linking Open Data Drug de W3C [30], plusieurs sociétés

24. http://data.gov.uk/linked-data
25. http://www.data.gov/semantic
26. http://id.loc.gov/authorities/about.html
27. http://dblp.l3s.de/
28. http://www.rkbexplorer.com/data/
29. http://data.semanticweb.org/
30. http://esw.w3.org/HCLSIG/LODD

pharmaceutiques (Eli Lilly, AstraZeneca et Johnson & Johnson) ont fait un effort de coopération pour connecter des données sur les médicaments et les essais cliniques, afin de faciliter la découverte de médicaments (Jentzsch et al., 2009).

Les ensembles de données commerciales :

Un exemple de la publication des données liées dans le domaine du commerce est RDF Book Mashup (Bizer et al., 2007a). Cet ensemble de données utilise le vocabulaire Simple Commerce pour représenter et republier des données sur des livres extraites à partir les APIs Web Amazon.com et Google Base.
Good Relations (Hepp, 2008) est une ontologie très riche qui décrit plusieurs aspects de e-commerce, tels que les entreprises, les produits et les services, les offres, les heures d'ouverture et les prix.
L'ensemble de données ProductDB [31] rassemble et connecte des données sur des produits de différentes sources pour démontrer le potentiel d'intégration dans ce domaine.

Les ensembles de données sociaux :

Semantic MediaWiki [32] et Ontowiki [33] sont des systèmes Wiki qui publient des contenus structurés générés par les utilisateurs sous forme de Linked Data. Il y a plusieurs installations de Semantic Media Wiki qui publient leurs contenus sur le web de données.
Dernièrement, les principes et les technologies des données liées ont été adoptés par des acteurs majeurs dans le contenu généré par l'utilisateur et des sphères de médias sociaux comme Facebook de l'Open Graph Protocol [34] qui utilise RDFa pour décrire des informations dans leurs pages web.
Le projet LOD Cloud a beaucoup de propriétés telles que :
1. Tout le monde peut publier tout type de données sur le LOD Cloud.
2. Il peut exister des informations conflictuelles sur une entité dans le LOD Cloud.
3. Les entités sont reliées par des liens RDF pour créer un espace de données qui couvre des ensembles de données et découvrir de nouvelles sources de données en suivant les liens RDF.
4. Si une application Linked Data rencontre des données décrites avec un vocabulaire inconnu, l'application peut déréférencer les URIs qui identifient les termes des

31. http://productdb.org/
32. http://semantic-mediawiki.org/wiki/Semantic_MediaWiki
33. http://ontowiki.net/Projects/OntoWiki
34. http://opengraphprotocol.org/

vocabulaires afin de trouver leurs définitions.
5. Les éditeurs des données ne sont pas limités dans leur choix de vocabulaires pour la représentation des données.
6. L'utilisation du protocole HTTP comme un mécanisme d'accès aux données normalisé et RDF comme un modèle de données standardisé normalisé permet de simplifier l'accès aux données par les applications clientes.

2.11 Processus d'intégration des données liées

Un système d'intégration des données vise à fournir à l'utilisateur une vue unifiée en combinant des informations à partir de différentes sources de données et offrant une interface cohérente. Avec la croissance de sources de données disponibles dans le LOD Cloud, le problème d'hétérogénéité de données dans ces sources augmente et donc le besoin d'accès à toutes ces sources à travers une interface cohérente est le défi de nombreuses recherches dans le domaine d'intégration des données liées. Les fournisseurs des données facilitent l'intégration des données liées en créant des liens RDF entre les sources de données.
Parallèlement à la liaison des entités, l'intégration des données liées exige également l'alignement de différents vocabulaires qui sont utilisés pour décrire les entités ainsi que la résolution des conflits de données entre les sources de données liées.

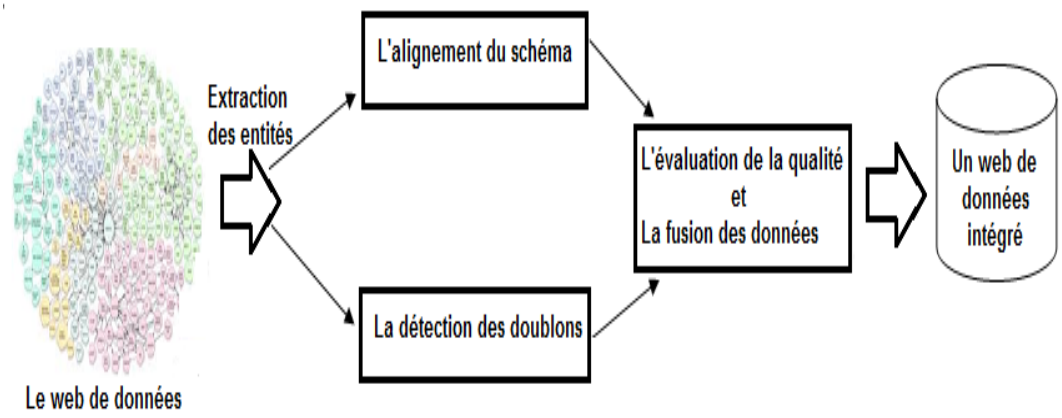

Figure 2.7 – Le processus d'intégration des données liées Bryl et al. (2014).

La figure 2.7 illustre le processus d'intégration des données liées. Trois étapes sont nécessaires pour obtenir un web de données intégré : la première étape consiste à appliquer un ensemble de mappings au niveau de schémas (alignement de schémas), la deuxième étape consiste à aligner les différentes entités qui représentent le

même objet du monde réel dans les sources de données (détection de doublons ou la résolution d'identité) et la dernière étape consiste à évaluer la qualité des données et la fusion des représentations en double pour résoudre les incohérences des données (évaluation de la qualité et la fusion des données) (Bryl et al., 2014).

a) L'alignement de schémas ou le mapping de vocabulaires : est inévitable quand les fournisseurs du LOD Cloud utilisent différents vocabulaires pour représenter le même type d'informations. Le mapping au niveau de schémas permet de générer un ensemble de connections ou d'équivalences entre les classes et les propriétés des ontologies LOD. Par conséquent, le mapping de vocabulaires est utilisé pour ajouter des relations entre les classes et les propriétés qui sont sémantiquement liées. Donc, les outils qui traduisent des termes de différents vocabulaires en un schéma cible et unique sont nécessaires.

b) La détection des doublons : Cette tâche est liée à l'identification d'objet (object identification), le couplage d'enregistrements (record linkage) et la réconciliation de référence (reference reconciliation). Elle permet de générer des liens *owl :sameAs* au niveau d'instances pour identifier et connecter différentes entités qui se réfèrent au même objet du monde réel. En fait, il existe des mesures de similarité syntaxiques et sémantiques (voir l'annexe B) qui peuvent être utilisées pour comparer les instances et detecter les doublons.

c) L'évaluation de la qualité et la fusion des données : Cette étape permet d'évaluer l'exactitude des données devant être fusionnées et mesurer la fiabilité des ensembles de données liées. Selon (Jens and Felix, 2009), la qualité des données peut être évaluée à l'aide de plusieurs dimensions : la complétude, la concision, la cohérence, l'exactitude, l'intelligibilité, l'objectivité, la rapidité, la crédibilité, la pertinence, la vérifiabilité, la disponibilité, etc. Ces dimensions ne sont pas indépendantes les unes des autres et en général seulement un sous-ensemble d'entre eux est pertinent dans une situation spécifique. Pour que les applications consomment efficacement des données liées, elles doivent être évaluées et intégrées en fonction de leurs qualités. Dans le contexte d'intégration des données, la fusion des données est définie comme "le processus de combinaison de plusieurs enregistrements représentant le même objet du monde réel dans une représentation unique, cohérente et propre" (Jens and Felix, 2009). La fusion des données est généralement considérée comme une troisième étape qui suit l'alignement de schémas et la résolution d'identité. La fusion des données a deux objectifs principaux : accroître la complétude et

la concision des données qui sont à la disposition des utilisateurs et des applications (Jens and Felix, 2009).

2.12 Conclusion

Dans ce chapitre, nous avons présenté les notions de base sur Linked Data. En outre, nous avons détaillé les différents éléments qui constituent les données liées. Nous avons aussi présenté en détail les ensembles de données liées qui se trouvent dans le projet Linked Open Data Cloud.
Pratiquement, l'intégration est un processus qui permet de réduire l'hétérogénéité des données et de fournir une vue unifiée des différentes sources de données. Dans la littérature, il existe plusieurs travaux qui visent à intégrer des ensembles de données liées du LOD Cloud. Le chapitre suivant introduit en détail les différentes techniques d'intégration des données liées.

.

Chapitre 3

Etat de l'art

Contents

Allah aime, lorsqu'une personne effectue une chose, qu'elle le fasse avec soin.

Prophète Mohammad (paix et bénédiction d'Allah soient sur lui)

3.1 Introduction

Le but du mouvement Linked Data est d'étendre le web avec un espace de données global en publiant des ensembles de données selon un ensemble de meilleures pratiques et en établissant des liens RDF entre les sources de données. Actuellement, la quantité de données liées disponibles dans le LOD Cloud est énorme en évoluánt de 12 ensembles de données en 2007 à 9960 ensembles en 2016. Cependant, la qualité des sources de données dans le LOD Cloud varie considérablement selon les domaines et les ensembles de données (Bryl et al., 2014), ce qui a créé des challenges lors de l'utilisation des données.
Comme indiqué précédemment, il existe trois étapes nécessaires pour intégrer les données liées : l'alignement de schémas, la résolution d'identité ainsi que l'évaluation de la qualité et la fusion des données. La concision des données est l'un des problèmes liés à la qualité des données liées parce que plusieurs sources de données utilisent différentes entités pour représenter le même objet du monde réel.
Le présent chapitre est organisé en trois sections. Section 2 présente les différentes approches d'intégration des données liées, y compris les techniques de découverte de liens, les approches d'alignement d'ontologies LOD et les méthodes de fusion des données liées. En outre, nous proposons des études comparatives des principales méthodes d'intégration des données liées. Dans la section 3, nous présentons quelques problèmes non élucidés dans les travaux connexes qui peuvent être résolus par nos approches et nous concluons ce chapitre.

3.2 Intégration des données liées

Avec la croissance des sources de données disponibles dans le LOD Cloud, le désir d'accéder à toutes ces sources à travers une interface cohérente a été le défi de nombreuses recherches dans le domaine d'intégration des données. Un système d'intégration des données vise à fournir à l'utilisateur une vue unifiée en combinant le résultat à partir de différentes sources de données et offrant une interface cohérente (Jens and Felix, 2009). Dans cette section, nous présentons les différentes approches d'intégration des données liées qui sont utilisées pour résoudre les problèmes d'hétérogénéité des données.

3.2.1 Découverte des liens

Actuellement, le web de données liées connait une croissance explosive de sources de données. Malgré que le nombre de triplets RDF disponibles dans le LOD Cloud

a dépassé 187 millions [1], moins de 23% de ces triplets représentent des liens entre les ensembles de données. Pourtant, les liens entre les ensembles de données liées jouent un rôle clé dans plusieurs tâches importantes telles que l'intégration des données (Ben-David et al., 2010), les inférences à une grande échelle (Urbani et al., 2010) et l'interrogation des données (Lopez et al., 2009). Avec la grande quantité d'informations disponibles dans le LOD Cloud, les frameworks de découverte des liens dans un temps efficace sont devenus indispensables pour mettre en oeuvre le quatrième principe des données liées, à savoir l'établissement des liens entre les sources de données (Volz et al., 2009), (Ngonga and Auer, 2011). Au cours de ces dernières années, plusieurs approches ont été développées pour découvrir des liens typés entre les différents ensembles de données liées. Elles peuvent être classées en deux catégories fondamentales : générale et spécifique.

A) Domaine spécifique

Les approches de découverte des liens de cette catégorie permettent de découvrir des liens entre des ensembles de données liées dans un domaine spécifique.
RKB Knowledge base (RKBCRS) (Glaser et al., 2009) permet de découvrir des liens entre des entités dans des ensembles de données liées dans le domaine de la recherche scientifique (c'est-à-dire les universités et les conférences). Pour chaque nouveau mapping, ce framework écrit un nouveau programme, dans lequel les ensembles de données source et cible ainsi que la fonction de mapping doivent être déclarés.
Le projet RKBExplorer [2] extrait des méta-données RDF à partir des sources de données hétérogènes pour remplir ses bases de connaissances avec des instances selon l'ontologie AKT [3]. En particulier, les instances des personnes, des publications et des institutions ont été extraites à partir de grands sites de métadonnées tels que DBLP et ACM. L'établissement des liens a été mis en oeuvre par le service CRS (Consistent Reference Service) qui relie les entités équivalentes en comparant les propriétés, y compris leurs types et leurs étiquettes. Jusqu'à présent, le CRS est limité à lier des objets dans les bases de connaissances sous-jacentes de RKBExplorer et ne peut pas être utilisé pour d'autres tâches, sans autres implémentations.
GNAT (Raimond et al., 2008) est un outil qui a été développé pour le domaine de la musique. Il met en oeuvre plusieurs algorithmes de matching d'instances. OGMA (Online Graph Matching Algorithm) est l'algorithme d'appariement de graphes en ligne le plus sophistiqué qui applique une approche de propagation de similarité pour découvrir des ressources équivalentes. L'approche de base implémentée par OGMA

1. http://stats.lod2.eu/stats
2. www.rkbexplorer.com
3. http://www.aktors.org/publications/ontology/

commence avec une seule ressource $s \in S$. Ensuite, elle récupère les ressources candidates pour le matching $t \in T$ en comparant les propriétés. Dans le cas d'une ambiguïté, les ressources liées à s et t dans leurs bases de connaissances respectives sont comparées et leurs valeurs de similarités sont cumulées pour recalculer $\sigma(s, t)$. Dans (Luis and Cervantes, 2013), les auteurs ont décrit l'utilisation des données liées pour générer un ensemble de données financières qui fait partie du projet FLORA (Financial Linked Open Data Reasoning and Management for Web Science). Cet ensemble de données stocke les triplets RDF générés par l'extraction de 409 374 informations financières, qui ont été publiées en format XBRL (b52, June 19, 2016) par différentes compagnies américaines à travers le système EDGAR (b53, June 19, 2016). En outre, ils décrivent un processus pour découvrir d'autres liens entre l'ensemble de données FLORA et d'autres ensembles de données du LOD Cloud afin de fournir une base de connaissances financières. En particulier, ils ont utilisé le framework SILK pour connecter les ressources de FLORA avec des ressources des ensembles de données externes comme : DBpedia, Semantic XBRL et la base de connaissances SEC.

B) Générale

Ces approches sont conçues pour effectuer des mappings indépendamment des domaines des bases de connaissances source et cible. RDF-AI (Scharffe et al., 2009) est un framework qui vise à intégrer des ensembles de données RDF. Il implémente une approche de cinq étapes qui comprend : le module du prétraitement prépare les graphes d'entrée afin d'adapter les ensembles de données à une même version d'ontologie, traduit les propriétés sélectionnées en utilisant l'API Google translate et harmonise les noms. Le module de matching matche les ensembles de données liées pour produire des alignements en utilisánt le Wordnet et des mesures de similarité syntaxiques pour calculer la similarité entre deux ressources. RDF-AI utilise un fichier de configuration qui permet à l'utilisateur de sélectionner les propriétés les plus pertinentes pour le mapping des ressources, la stratégie de la fusion (sélectionner l'ensemble de données source et cible), le seuil, etc. L'utilisation de ce framework dans le processus de mapping est très couteuse en termes du temps.
SILK (Volz et al., 2009) est un autre framework de découverte des liens dans Linked Data. Il combine des mesures de similarité pour calculer une valeur de similarité totale pour une paire d'entités. Il met en oeuvre plusieurs approches pour minimiser le temps d'exécution nécessaire pour le mapping au niveau d'instances. En plus de la mise en oeuvre d'un index rugueux pour atteindre une complexité temporelle quasi-linéaire, SILK met également en oeuvre un algorithme de blocage efficace

appelé MultiBlock (Isele et al., 2011) pour réduire son temps d'exécution global. MultiBlock utilise un indice multidimensionnel dans lequel les objets similaires sont situés à proximité les uns des autres. Dans chaque dimension, les entités sont indexées par une autre propriété en augmentant l'efficacité de l'indice d'une manière significative.
Le framework LIMES (Ngonga and Auer, 2011) utilise les caractéristiques mathématiques des espaces métriques afin de réduire le nombre de calcul qui doit être effectué par le système. En particulier, il utilise l'inégalité triangulaire dans les espaces métriques pour calculer les estimations pessimistes des similarités entre les instances. En se basant sur ces approximations, LIMES peut filtrer un grand nombre de paires d'instances qui ne peuvent pas suffire la condition de matching définie par l'utilisateur. Ensuite, il calcule la similarité entre les paires d'instances restantes et retourne le résultat de matching à l'utilisateur.
La nouvelle version de LIMES étend et intègre l'algorithme PPJoin+ (Chuan et al., 2011) dans sa version précédente (Ngonga and Auer, 2011) afin qu'il permette de faire face aux configurations complexes. En outre, LIMES intègre l'algorithme HYPPO (Ngomo, 2011), et HR3 (Ngomo, 2012) qui reposent sur la technique "Space Tiling" pour optimiser le temps d'exécution. Cette technique divise l'espace de recherche en petits morceaux ou blocs indépendantes à travers les dimensions du problème.
KnoFuss (Andriy et al., 2012) met également en oeuvre des techniques de blocage pour atteindre des temps d'exécutions acceptables lors du mapping de grands ensembles de données. Il repose sur la découverte non supervisée des paramètres de similarités requis. Au lieu d'utiliser les données étiquetées, cette méthode prend en compte plusieurs propriétés souhaitées dont la répartition des valeurs de similarité de la sortie doivent satisfaire l'utilisateur. Elle ajoute ces caractéristiques dans une fonction d'évaluation pour établir les paramètres de similarités qui maximisent la qualité des liens résultants selon les propriétés considérées.
Dans (Dreßler and Ngomo, 2014), les auteurs abordent l'efficacité de l'exécution des mesures de Jaro-Winkler, qui sont connus pour être efficace lorsqu'on compare les noms des personnes. Pour cela, ils dérivent des équations qui permettent de débarrasser un grand nombre de calculs lors de l'exécution des comparaisons Jaro-Winkler bornées avec des seuils élevés. Dans ce contexte, l'algorithme fait un filtre à base de langueurs de chaînes de caractères (c'est-à-dire il filtre les chaînes de caractères qui ont une grande différence en termes du nombre de caractères). En outre, les auteurs présentent un filtre à base de caractères qui permet de détecter si deux chaînes s et t sont similaires selon la mesure Jaro-Winkler.

	Knofuss (Andriy et al., 2012)	SILK (Robert et al., 2011)	LIMES (Kevin and Axel-Cyrille, 2014)
Entrées	RDF, SPARQL	RDF, SPARQL, CSV	RDF, SPARQL, CSV
Sorties	owl :sameAs	owl :sameAs et autres liens spécifiées par l'utilisateur	owl :sameAs et autres liens spécifiés par l'utilisateur
Configuration	Manuelle	Manuelle	Manuelle
Types de mesures de similarité	Des similarités de chaînes de caractères	Des similarités de chaînes de caractères, égalité, numérique et date	Des similarités de chaînes de caractères, égalité, numérique et date
Utilisation des sources de connaissances externes	Non	Non	Non
Méthode d'optimisation du temps d'exécution	L'indexation	multidimensionnel	espace carrelage (Space tiling)

Table 3.1 – Comparaison des techniques de découverte des liens dans un temps efficace

C) Synthèse

Le tableau 3.1 illustre une comparaison entre les approches de découverte des liens qui permettent d'optimiser la complexité temporelle du processus de matching, cette comparaison est faite en utilisant trois critères. Le premier critère permet de comparer ces approches en se basant sur leurs entrées et leurs sorties. Le deuxième critère étudie les différentes mesures de similarité ainsi que les sources de connaissances externes qui sont utilisées dans le processus de matching. Le dernier critère permet de comparer les approches de découverte des liens en se basant sur les méthodes d'optimisation du temps d'exécution.

Toutes ces approches acceptent en entrée des fichiers RDF ou des requêtes SPARQL pour récupérer des données à partir des SPARQL endpoint. Par ailleurs, SILK et LIMES peuvent découvrir des liens entre des fichiers CSV. Ainsi, ces framework génèrent en sortie des liens *owl :sameAs*. Les frameworks LIMES et SILK peuvent générer d'autres types de liens qui doivent être spécifiés par l'utilisateur. Ces approches utilisent principalement des méthodes d'indexation pour réduire la complexité temporelle du processus de matching. Par conséquent, toutes les paires qui ne correspondent pas aux conditions de matching sont éliminées. Dans le processus de matching, toutes ces approches utilisent des mesures de similarité de chaînes de caractères telles que Levenshtein, Jaro, Jaccard, etc (Nentwig et al., 2015). Cepen-

dant, quelques approches (SILK, LIMES) soutiennent des mesures numériques, de date, ou des coordonnées géographiques pour calculer la similarité entre les entités. Enfin, toutes ces approches utilisent une configuration manuelle pour combiner les valeurs de différentes matchers, c'est-à-dire, l'utilisateur doit spécifier les conditions pour connecter deux entités. Néanmoins, ces approches n'utilisent pas de sources de connaissances externes, comme des thésaurus.

3.2.2 Alignement des ontologies LOD

Dans le contexte d'intégration des données, il est important d'établir des liens corrects entre les données de différentes sources. Le problème d'établissement des liens entre les ensembles de données est étroitement lié au problème de matching d'ontologies utilisées dans le web sémantique et les bases de données (Cruz et al., 2013). La publication de grands ensembles de données liées dans le LOD Cloud a besoin de stratégies efficaces de matching d'ontologies. Dans ce contexte, les outils d'extraction d'entités à partir des documents non structurés considèrent un seul ensemble de données et ne peuvent pas lier les données extraites avec d'autres ontologies ou ensembles de données. Le problème majeur est que les méthodes de matching d'ontologies qui existent déjà ne répondent pas aux besoins d'efficacité et de précision. Un autre problème des systèmes classiques de matching d'ontologies est de trouver d'autres relations entre les concepts, telles que des relations de subsomptions, disjonctions, partonomiques, etc. Actuellement, plusieurs approches ont été développées pour aligner les ontologies du LOD Cloud. Cette section présente un état de l'art sur l'alignement d'ontologies LOD ainsi qu'une comparaison entre ces méthodes (Bencherif and Malki, March 2016).

A) Méthodes d'alignement d'ontologies LOD

Généralement, le problème d'hétérogénéité d'ontologies est résolu par des méthodes de matching d'ontologies, qui visent à trouver des correspondances entre des entités (des classes, des propriétés ou des individus) sémantiquement liées. On peut présenter ces correspondances en utilisant des relations d'équivalences et d'autres relations, telles que : des relations de subsomptions et de disjonctions. Au cours des dernières années, plusieurs approches d'alignement d'ontologies LOD sont développées. Dans cette section, nous présentons les différentes techniques d'alignement d'ontologies LOD ainsi qu'une étude comparative pour discuter leurs limitations.
BLOOMS (Jain et al., 2010) est un système d'alignement d'ontologies du LOD Cloud, il permet d'ajouter des liens d'équivalences et de subsomptions au niveau de

schémas (entre les classes des ontologies). BLOOMS utilise Wikipédia et une API d'alignement (Euzenat, 2004a) pour trouver des liens entre les concepts des ontologies LOD. D'abord, il procède un prétraitement des ontologies d'entrée pour en extraire des mots simples. Pour chaque classe candidate de matching, BLOOMS utilise Wikipédia pour construire un ensemble d'arbres. Ensuite, il compare les arbres construits après qu'il supprime les noeuds qui se répètent dans les arbres source et cible. Finalement, BLOOMS analyse le résultat en utilisant l'API d'alignement.
BLOOMS+ (Damova et al., 2010), la version étendue de l'approche BLOOMS, a été développée pour aligner des ontologies LOD avec une ontologie de haut niveau (Proton)(Cruz et al., 2013). BLOOMS+ étend BLOOMS en utilisant une mesure plus sophistiquée pour trouver les classes qui doivent être alignées et en prenant en compte des informations contextuelles pour supporter ou rejeter un alignement.
Dans (Cruz et al., 2009), l'outil AgreementMaker a été développé pour aligner les ontologies LOD dans plusieurs domaines, y compris : le géospatiale (Cruz et al., 2002), l'environnement (Cruz et al., 2007) et le biomédicale (Sunna and Cruz, 2007), qui peuvent être constituées des centaines ou même des milliers de concepts. Cet outil a été évolué pour tenir compte : 1) des besoins des utilisateurs 2) d'un ensemble d'ontologies en entrée et d'un ensemble de fichiers en sortie 3) du choix des méthodes de matching selon les différentes caractéristiques prises en compte dans la comparaison (conceptuels ou structurels), de l'intervention des utilisateurs (manuelles ou automatiques), du type de matching (au niveau d'instances ou de schémas), etc. Dans le processus de matching, AgreementMaker produit deux matrices de similarités. La première permet de calculer la similarité entre les classes et la deuxième vise à calculer la similarité entre les propriétés. Il utilise des méthodes syntaxiques (par exemple, edit distance, Jaro-Winkler) et sémantiques (le Wordnet) pour calculer la similarité entre les concepts.
Par la suite, AgreementMaker a été étendu pour améliorer la précision de son système (Cruz et al., 2013). Pour cela, quatre méthodes ont été développées pour comparer des ontologies sources et cibles :
la méthode d'extension de mapping d'équivalence utilise une mesure de similarité pour découvrir un ensemble de mappings d'équivalence et déduire un ensemble de mappings de sous classes et de superclasses.
La méthode d'analyse des noms composés analyse les noms composés pour découvrir des mappings de subsomptions (sous classes et superclasses). La méthode de comparaison lexicale polysémique à base de distance compare l'annotation lexicale des concepts de l'ontologie pour découvrir des mappings de subsomptions. La méthode de matching global utilise des ontologies externes pour découvrir des mappings de

subsomptions.
HCM (Holistic Concept Mapping) (Grütze et al., 2012) extrait une représentation de connaissances pour tous les concepts disponibles. Cette représentation peut être une simple chaîne descriptive, un vecteur de caractéristiques, ou une structure de données plus complexe. Pour cela, il propose les méthodes TFIDFn-extractor et ConceptID-extractor pour déterminer les mots clés des concepts et traiter leurs descriptions. En outre, HCM utilise la méthode WCF (Wikipedia Category Forests) (Jain et al., 2010) pour effectuer une recherche Wikipedia à l'aide d'une requête de recherche qui est construite par la concaténation de tous les mots-clés. Ensuite, il applique un regroupement thématique afin de créer des petits ensembles de concepts. Au sein de ces ensembles, HCM crée des alignements en identifiant des représentations de connaissances similaires et un raisonnement entre eux.
Afin de trouver des relations partonomiques entre les concepts des ontologies LOD, l'outil PLATO (Joshi et al., 2012) a été développé. Tout d'abord, il utilise les pages Wikipédia pour créer tous les couples possibles entre les ressources d'un ensemble de données. Ensuite, il produit les hypothèses possibles pour chaque couple en utilisant la taxonomie de Winston. Ensuite, PLATO utilise l'API Bing Search2.0 [4] pour évaluer les patterns construits. Enfin, il calcule le nombre total de pages qui contiennent chaque pattern et il utilise ce nombre pour spécifier le type de relation entre les couples.
N2R-Partie (Pernelle et al., 2013) est une autre approche d'alignement d'ontologies, elle permet de calculer les scores de similarités entre les propriétés des ontologies LOD. Elle développe une nouvelle mesure pour calculer la similarité entre les entités en se basant sur les propriétés non mappées. N2R est basée sur un ensemble d'équations non linéaires pour exprimer les influences entre les similarités. Elle exploite la sémantique des clés qui sont déclarées et identifiées comme communes dans les ontologies pour déduire des liens d'identité entre les instances et les classes.

B) Synthèse

Le tableau 3.2 présente une comparaison entre les approches d'alignement d'ontologies LOD, cette comparaison est faite en utilisant plusieurs critères. Le premier critère permet de comparer les types des liens générés en sortie. Le second critère étudie l'utilisation des sources de connaissances externes ainsi que les algorithmes qui sont utilisés dans le processus de matching. Le dernier critère indique les ensembles de données liées qui sont utilisés dans l'expérimentation. Ces approches permettent de trouver plusieurs types de liens, par exemple, les liens d'équivalence

4. http://msdn.microsoft.com/en-us/library/dd251056.aspx

	Types de liens	Utilisation des sources de connaissances externes	Algorithmes de matching	Ontologies utilisées pour l'alignement
BLOOMS (Jain et al., 2010)	Equivalence, subsomption	Wikipédia	similarité contextuelle sur le super concept, la fonction Tree Overlap en utilisant la profondeur du noeud	DBpedia, Geonames, BBC Program, Music, FOAF Profiles, SIOC, AKT Reference et Semantic Web Conference
BLOOMS+ (Mariana et al., 2010)	Equivalence, subsomption	Wikipédia et l'ontologie Proton	La fonction Tree Overlap et Similarité Contextuelle	DBpedia, Geonames et Freebase
Agreement Maker (Isabel et al., 2013)	Equivalence, subsomption	Wordnet, autres ontologies LOD	ASM (Advanced Similarity Matcher)	AKT portal, BBC program, DBpedia, FOAF, GeoNames, Music, Semantic web conference, SIOC
HCM (Toni et al., 2012)	Equivalence, disjonction	Wikipédia	Les mesures TF/IDF et Jaccard	DBpedia, YAGO
PLATO (Krishna et al., 2012)	Partonomie	Wikipédia, Wordnet, l'API Bing Search et la taxonomie de Winston	L'API Mediawiki	DBpedia, Freebase
N2R (Pernelle et al., 2013)	Equivalence	Non	Les algorithmes Falcon-AOV-Doc I-Sub	Création de nouvelles ontologies

Table 3.2 – Comparaison entre les méthodes d'alignement d'ontologies LOD

(presque toutes ces approches ajoutent ce type de liens entre les ontologies LOD), les liens de subsomptions, les liens partonomiques, etc.

Cependant, PLATO est la seule approche qui vise à ajouter des liens partonomiques entre les classes. La plupart de ces approches utilisent Wikipédia comme une source de connaissances externe, et autres approches utilisent le Wordnet pour calculer la similarité sémantique entre les classes de différentes ontologies. Ainsi, autres sources de connaissances, telles que la taxonomie de Winston et autres ontologies LOD, sont utilisées dans le processus d'alignement d'ontologies. Dans le processus de matching, les approches d'alignement d'ontologies LOD utilisent plusieurs algorithmes de mapping comme Falcon-AOV-Doc et I-Sub et différentes mesures de similarité, par exemple : ASM, TF/IDF et Jaccard pour calculer le score de similarité entre les classes. Pour l'expérimentation, ces approches essayent d'aligner plusieurs

ontologies de différents domaines, tels que DBpedia, Geonames, BBC, Program, Music, FOAF Profiles, SIOC, AKT Reference, Semantic Web Conference, YAGO et Freebase.

3.2.3 Fusion des données liées

Les sources de données disponibles dans le LOD Cloud contiennent différentes informations sur les entités. Comme indiqué précédemment, un système d'intégration de données vise à fournir une vue unifiée de toutes les données existantes (Jens and Felix, 2009). Le processus de la fusion des données vise à combiner, agréger et résoudre des conflits de valeurs provenant de différentes sources. Il suit le processus de la résolution d'identité qui définit des valeurs différentes pour représenter les mêmes objets du monde réel. En outre, un système de fusion de données vise à augmenter la complétude et la concision des données qui sont disponibles pour les utilisateurs et les applications (Jens and Felix, 2009). La complétude signifie qu'aucun objet n'est oublié dans le résultat de la fusion, la concision signifie qu'aucun objet est représenté deux fois et les données présentées à l'utilisateur ne sont pas contradictoires. Donc, le résultat du processus de la fusion est utilisé pour faciliter l'interrogation de plusieurs sources de données qui contiennent des données hétérogènes et des informations contradictoires. Au cours de ces dernières années, quelques approches de fusion des données liées ont été proposées. Dans cette section, nous présentons les différentes méthodes de fusion des données liées ainsi qu'une étude comparative pour en discuter leurs limitations.
Sieve (Mendes et al., 2012) est un composant intégré dans le framework LDIF (Linked Data Integration Framework) (Schultz et al., 2010). Il comprend deux modules : un module d'évaluation de la qualité et un module de la fusion des données.
D'une part, le module d'évaluation de la qualité utilise une liste de fonctions d'évaluation telles que : la fonction "TimeCloseness" qui calcule la distance entre la date d'entrée et la date actuelle. Par conséquent, les données les plus récentes reçoivent un score proche de 1. La fonction "Preference" affecte les scores avec de plus petites valeurs à chaque graphe d'URIs donné dans un fichier de configuration.
D'autre part, le module de la fusion utilise certaines fonctions qui suppriment des valeurs d'entrées selon certaines mesures de qualité (Mendes et al., 2012), par exemple : la fonction de filtrage "Filter" supprime les valeurs où la mesure de qualité est inférieure à un seuil donné. La fonction "KeepSingleValueByQualityScore" ne conserve que la valeur où l'évaluation de la qualité est plus élevée. En outre, la fonction "Moyenne" choisit le taux, les fonctions "Max" et "Min" choisissent le maximum et le minimum des valeurs numériques. La fonction "PickMostFrequent" choisit les valeurs

les plus apparues à partir d'un ensemble de valeurs contradictoires. Les fonctions "First", "Last" et "Random" : prend le premier, le dernier ou l'élément à une position aléatoire pour une propriété donnée.
Dans (Michelfeit and Mynarz, 2014), les auteurs abordent deux problèmes liés à la fusion de Linked Data. Le premier problème est que chaque propriété dans des ensembles de données liées est résolue d'une manière autonome. Afin de résoudre ce problème, ils introduisent la dépendance entre les propriétés pour trouver le résultat correct au cours de la fusion. Le second problème est que les ressources qui représentent des valeurs structurées sont fusionnées ensemble. Ainsi, ils proposent une méthode pour fusionner des valeurs structurées séparément en intégrant les ressources dépendantes via les propriétés annotées.
Dans (Zhao and Ichise, 2013), les auteurs présentent une approche qui vise à intégrer les ontologies hétérogènes disponibles dans le LOD Cloud. Tout d'abord, cette approche rassemble les instances liées par le prédicat *owl :sameAs* ainsi que leurs prédicats et leurs objets. Ensuite, elle classifie les prédicats et les objets rassemblés en cinq types différents : classe, String, Date, nombre et URI. Ensuite, elle regroupe les classes et les propriétés liées. Enfin, cette approche agrège les classes et les propriétés intégrées de tous les patterns graphiques pour construire une ontologie intégrée.
Un autre travail lié à la fusion des connaissances est celle présentée par (Dong et al., 2014). Il adapte les techniques de fusion des données existantes pour construire des bases de connaissances. Tout d'abord, il extrait des informations à partir des sources de données, puis il fusionne et détermine le degré d'exactitude (correcteness) des informations extraites. En outre, le travail proposé présente plusieurs améliorations aux méthodes existantes pour la fusion des données pour améliorer la qualité de connaissances collectées en calculant la probabilité de chaque triplet. Finalement, les auteurs évaluent la qualité des connaissances fusionnées en utilisant l'ensemble de données Freebase parce que ses triplets sont validées manuellement.

Synthèse

Le tableau 3.3 illustre une comparaison entre les approches de fusion des données liées, cette comparaison est faite en utilisant trois critères. Le premier critère permet de comparer les méthodes de découverte des liens qui sont utilisées par les approches de fusion des données liées. Le deuxième critère vérifie si ces approches évaluent la qualité des données avant la fusion. Le dernier critère présente les différentes fonctions de fusion qui sont utilisées par ces approches. Ces approches utilisent diverses méthodes pour découvrir des liens entre les ensembles de données liées. Par

	Méthode de découverte des liens	Évaluation de la qualité	Fonctions de fusion
(Pablo et al., 2012)	Les frameworks R2R et SILK	Oui	Filter, KeepSingleValueByQualityScore,Average, Max, Min, First, Last, Random et PickMostFrequent
(Michelfeit and Mynarz, 2014)	Des mesures de similarité syntaxiques	Oui	Fusion des valeurs structurées, la résolution de chaque propriété dépendances
(Zhao and Ichise, 2013)	Des mesures de similarité syntaxiques et le Wordnet	Non	PickMostFrequent
(Luna et al., 2014)	Probabilité et le Wordnet	Oui	Améliore les méthodes de fusion des données suivantes : VOTE, ACC et POPESCU

Table 3.3 – Comparaison entre les approches de fusion des données liées.

exemple, le framework Sieve suit le processus d'intégration des données liées présenté dans (Bryl et al., 2014). Donc, il utilise le framework R2R (Bizer and Schultz, 2010) pour le mapping au niveau de schémas et le framework SILK (Volz et al., 2009) pour le mapping au niveau d'instances. Ensuite, LDIF utilise Sieve pour évaluer la qualité et fusionner les données. Les autres approches de fusion des données liées utilisent les probabilités, le Wordnet pour calculer la similarité sémantique entre les entités et les mesures de similarité pour calculer la similarité syntaxique entre deux chaînes de caractères. Par ailleurs, on remarque que la plupart des approches présentées dans le tableau ci-dessus utilisent des fonctions d'évaluation de la qualité avant la fusion des données liées. Enfin, ces approches utilisent diverses fonctions de fusion des données. Quelques approches améliorent des fonctions de fusion déjà existantes, alors que d'autres approches développent de nouvelles méthodes pour fusionner les données liées.

3.3 Discussion et conclusion

Le problème d'intégration des données liées est l'un des problèmes les plus difficiles qui affectent l'interopérabilité du Linked Open Data. Afin d'atteindre l'interopérabilité sémantique pour divers ensembles de données liées provenant de plusieurs domaines, plusieurs approches de découverte des liens, d'alignement d'ontologies LOD et de fusion des données liées ont été développées pour résoudre le problème d'intégration des données liées. Dans ce chapitre, nous avons présenté les différentes méthodes d'intégration des données liées, y compris la découverte des liens, l'ali-

gnement des ontologies et la fusion des données liées. Nous avons aussi comparé ces méthodes en se basant sur différents critères.
La première partie de ce chapitre concerne les approches qui sont proposées pour découvrir des liens typés au niveau d'instances. Ces approches peuvent prendre en charge en entrée : des fichiers RDF, des requêtes SPARQL ou des fichiers CSV. Les sorties de ces approches sont principalement des liens de type *owl :sameAs* ainsi qu'autres types de liens spécifiés par l'utilisateur. En outre, les approches de découverte des liens utilisent plusieurs mesures de similarité pour déterminer les entités similaires. Cependant, ces approches ont besoin d'améliorer leurs efficacités et leurs complexités temporelles (Nentwig et al., 2015). D'une part, l'amélioration de l'efficacité permet aux outils de découverte des liens de générer des mappings de hautes qualités (en termes de précision, rappel et F-mesure). Par conséquent, les résultats doivent être précis (c'est-à-dire, les liens générés doivent être correctes : une bonne précision). Un outil de découverte des liens devrait également générer autant que possible des liens pour assurer la complétude. En outre, la définition manuelle des spécifications de liaisons pour le processus de matching est difficile à réaliser dans de nombreux cas, même pour les experts des domaines. Par conséquent, il est souhaitable d'automatiser certaines décisions telles que le choix des mesures de similarité et le seuil. Un grand nombre de frameworks de découverte des liens dans les ensembles de données liées n'utilisent pas de sources de connaissances externes, comme les dictionnaires ou des liens et des mappings déjà connus.
D'autre part, l'amélioration de la complexité temporelle permet aux outils de découverte des liens d'être plus rapides et évolutives quand on veut matcher les grands ensembles de données liées (c'est-à-dire, des centaines de milliers ou des millions de ressources). Une approche non évolutive peut avoir une complexité quadratique parce qu'elle évalue toutes les paires possibles de ressources de deux ensembles de données liées. Par conséquent, le développement de frameworks performants est une exigence dans le processus de découverte des liens dans les grands ensembles de données liées.
Un autre besoin de ces frameworks est la visualisation des liens résultants. Donc, les frameworks de découverte des liens ont besoin de visualiser les liens résultants pour aider les utilisateurs à détecter des erreurs dans le processus d'appariement d'une part et de vérifier la complétude et l'exactitude des liens, d'autre part.
Contrairement aux approches de découverte de liens, notre approche améliore l'efficacité en utilisant une source de connaissances externe. En outre, elle est plus performante parce qu'elle permet de découvrir des liens entre de grands ensembles de données liées en un temps raisonnable. La deuxième partie de ce chapitre concerne

les approches qui ont été proposées pour ajouter des liens au niveau de schémas (c'est-à-dire, l'alignement des ontologies LOD). Ces approches prennent en charge en entrée des ontologies LOD. Les sorties de ces approches sont principalement des liens d'équivalence ainsi qu'autres types de liens tels que les liens de subsomptions, les liens de disjonctions, les liens partonomiques, etc. De plus, les approches d'alignement d'ontologies LOD utilisent plusieurs mesures de similarité ainsi que plusieurs sources de connaissances externes comme le Wordnet et Wikipédia pour calculer des liens entre les classes et les propriétés. Cependant, les approches d'alignement d'ontologies LOD sont limitées par l'établissement de liens d'équivalence et de subsomption. Donc, le développement d'autres méthodes pour ajouter d'autres types de liens (par exemple : des liens de disjonction, de partonomie, etc) entre les classes est une exigence parce qu'elles nous aident à détecter les inconsistances dans les ontologies LOD. Finalement, comme les ontologies LOD contiennent un grand nombre de classes et de propriétés, il faut minimiser le nombre de comparaisons entre les concepts des ontologies sources et cibles.

La dernière partie de ce chapitre concerne les approches qui permettent de fusionner les données liées. Ces approches extraient des entités à partir de plusieurs ensembles de données liées, puis combiner les données dupliquées pour fournir une seule vue à l'utilisateur. De plus, les approches de fusion des données liées utilisent plusieurs méthodes pour évaluer la qualité des données ainsi que plusieurs fonctions de fusion des données. Cependant, les problèmes soulevés dans ce contexte sont de (1) surmonter l'effet que plusieurs ensembles de données liées représentent le même objet du monde réel en utilisant différentes entités, (2) choisir les bonnes solutions (entités) au cours de la tâche de fusion et enfin (3) minimiser le temps d'exécution du processus de fusion.

Notre approche de fusion des données liées est complémentaire aux autres approches de fusion en minimisant le temps d'exécution lors de la fusion de grands ensemble de données liées d'une part, et en améliorant la qualité des donnés aprés la fusion d'autre part.

Chapitre 4

Amélioration des téchniques de découverte des liens dans Linked Data

Contents

Allah m'a révélé que vous devez vous montrer humbles, nul ne doit se vanter aux autres et nul ne doit opprimer les autres.

Prophète Mohammad (paix et bénédiction d'Allah soient sur lui)

4.1 Introduction

Le projet Linked Open Data Cloud a été évolué de 870 ensembles de données en janvier 2014 à 9960 ensembles de données en juin 2016, c'est-à-dire, en deux ans. Bien que le nombre de triplets RDF disponibles dans le LOD Cloud a maintenant dépassé 187 milliards [1], moins de 23% de ces triplets représentent des liens entre les ensembles de données. Récemment, des études dans le mouvement Linked Open Data montrent que 44% d'ensembles de données disponibles dans le LOD ne sont pas connectés à d'autres ensembles de données (Schmachtenberg et al., 2014). Pourtant, les liens entre les ensembles de données dans le LOD Cloud jouent un rôle primordial dans l'intégration des données (Bryl et al., 2014). Généralement, ces liens sont générés par le calcul de similarité entre les entités provenant de différentes sources de données (Heath and Bizer, 2011). Néanmoins, l'établissement des liens entre les ensembles de données représente encore l'un des défis les plus importants pour réaliser la vision du web de données (Andriy et al., 2012). Au cours de ces dernières années, les frameworks de découverte des liens ont été développés pour faciliter l'établissement des liens entre les instances de différentes bases de connaissances. En général, ces frameworks visent à découvrir des instances dans des ensembles de données liées qui doivent être connectés via le lien typé *owl :sameAs*. En effet, la tâche de découverte des liens est devenue difficile parce que les ensembles de données liées ne partagent pas les entités communes acceptées d'une part et ne réutilisent pas les vocabulaires et les ontologies préexistantes d'autre part.
Le reste du chapitre est organisé comme suit : dans la section 2, nous donnons un exemple illustratif de notre approche. Les sections 3, 4 et 5 décrivent la problématique de la découverte des liens, le workflow de la découverte des liens et les espaces métriques, respectivement. Dans la section 6, nous décrivons en détail notre approche de découverte des liens. La section 7 présente une expérimentation de notre approche ainsi qu'une comparaison entre notre approche et d'autres approches existantes.

4.2 Exemple illustratif

Dans cette section, nous présentons un exemple illustratif afin d'expliquer notre contribution. Nous choisissons Drugbank [2] comme un ensemble de données source et DBpedia comme un ensemble de données cible. Notre but est de découvrir les entités similaires qui se trouvent dans les deux ensembles de données et générer des liens de type *owl :sameAs* entre eux.

1. http://stats.lod2.eu/stats
2. http://www4.wiwiss.fu-berlin.de/drugbank/sparql

Nous utilisons la base de données Wordnet [3] afin de trouver tous les synonymes de chaque instance source (c'est-à-dire dans les instances qui se trouvent dans l'ensemble de données Drugbank). Par exemple, les synonymes de la ressource "people" : <http ://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/people> dans le Wordnet sont : *people, citizenry, multitude, masses, hoi polloi et the great unwashed*. Ensuite, nous indexons ces synonymes dans l'ensemble de données cible (DBpedia) pour récupérer toutes les instances candidates pour le matching et par conséquent réduire l'espace de recherche. Deux indices sont créés : un indice pour l'entité "people" et l'autre pour leurs synonymes. La liste ci-dessous représente le résultat d'indexation de la ressource <http ://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/people> dans l'ensemble de données cible DBpedia :

<http ://dbpedia.org/resource/The_People>, <http ://dbpedia.org/resource/The_people>,
<http ://dbpedia.org/ resource/People%21>, <http ://dbpedia.org/resource/People>,
<http ://dbpedia.org/resource/These_People>, <http ://dbpedia.org /resource/For_the_People>,
<http ://dbpedia.org/resource/For_The_People>, <http ://dbpedia.org/resource/By_the_People>
, <http ://dbpedia.org/resource/People_For>, <http ://dbpedia.org/resource/Will_of_the_People>,
<http ://dbpedia. org/resource/For_the _people>, <http ://dbpedia.org/resource/Of_The_People>,
<http ://dbpedia.org/resource/By_The _People> , <http ://dbpedia.org/resource/By_the_people>,
<http ://dbpedia.org/resource/By_the_People%2C_for_the_People> ,
<http ://dbpedia.org/resource/People_Are_People>, <http ://dbpedia.org/resource/People_are_People>,
<http ://dbpedia.org/resource/People_are_people>, <http ://dbpedia.org/resource/People_to_People>,
<http ://dbpedia.org/resource/People%2C_People>.

Ensuite, nous calculons l'ensemble d'exemplaires à partir de la liste d'instances ci-dessus et nous les supprimons de la liste d'indexation pour donner le résultat final de matching. La tableau 4.1 représente l'ensemble d'exemplaires de la ressource <http ://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/people> et le résultat de matching. Par conséquent, les instances similaires de la ressource <http ://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/people> de Drugbank dans l'ensemble de données cible DBpedia sont :

<http ://dbpedia.org/resource/The_People>, <http ://dbpedia.org/resource/People>,
<http ://dbpedia.org/resource/Citizenry>, <http ://dbpedia.org/resource/Multitude>,
<http ://dbpedia.org/resource/Hoi_polloi>, <http ://dbpedia.org/resource/The_hoi_polloi>,
<http ://dbpedia.org/resource/The_Great_Unwashed>, <http ://dbpedia.org/resource/Great_Unwashed>.

3. https://WordNet.princeton.edu/WordNet/download

L'entité	L'ensemble d'exemplaires	Le reste des ressources de T'	Résultat de matching
people	The_People, People_For, People_Are_People, For_the_people	The_people, People%21, People, These_People, For_the_People, For_The_People, By_the_People, Will_of_the_People, Of_The_People, By_The_People, By_the_people, By_the_People%2C_ for_the_People, People_are_People, People_are_people, People_to_People, People%2C_People	The_People, People
citizenry	Citizenry , Citizens_%E2%80%93_ Party _of_the_Citizenry	Citizens%E2%80%93 Party_of_the_Citizenry, Citizens-Party_of_the_Citizenry, Citizens_-_ Party_of_the_Citizenry	Citizenry
multitude	Multitude , Multitude :_War_and _Demo cracy_in_the _Age_of _Empire	Feeding_the_multitude, Feeding_of_the_multitude, A_Multitude_of_ Casualties, In_Your_Multitude, The_Assembled_ Multitude, Assembled_Multitude, Big_Daddy_Multitude	Multitude
masses	{}	{}	{}
hoi polloi	Hoi_polloi	The_hoi_polloi	Hoi_polloi, The_hoi_polloi
the great unwashed	The_Great _Unwashed	Great_Unwashed, Unwashed_biodiesel	The_Great_ Unwashed, Great_Unwashed

Table 4.1 – L'ensemble d'exemplaires de la ressource People ainsi que le résultat de matching

4.3 Problématique de la découverte des liens

Tim Berners-Lee définit le quatrième principe des données liées comme suit : "Inclure des liens vers d'autres URIs, afin que les gens puissent découvrir plus de choses" (Berners-Lee, 2006). Par conséquent, la découverte des liens entre les ensembles de données dans le LOD Cloud est une tâche très importante pour l'intégration des données, la recherche de connaissances et l'interrogation des sources de données liées. Comme indiqué précédemment, la tâche de découverte des liens dans Linked Data permet de trouver automatiquement des instances dans deux ensembles de données liées S (source) et T (cible) qui devraient être liés les uns aux autres, par exemple, par un lien *owl :sameAs*. Donc, le problème de découverte des liens peut être décrit comme suit :

Définition 1 : "Soit deux ensembles d'instances S (source) et T (cible), une mesure de similarité $\sigma : S \times T \rightarrow [0, 1]$ et un seuil $\theta \in [0, 1]$, le but du processus de

	Ensemble de données	Type de liens	Données	Parties d'intégration	Configuration	Exécution
La découverte des liens	Hétérogènes	owl :sameAs et autres	Interconnectées	Classes, propriétés et instances	Manuelle	En ligne, off-ligne
La résolution d'entités	Homogènes	équivalence	isolées	Attributs	Automatique, semi-auto	off-ligne

Table 4.2 – La différence entre la découverte des liens et la résolution d'entités

découverte de liens est de calculer l'ensemble the set $M = (s,t), \sigma(s,t) \geq \theta$" (Auer et al., 2011).
Soit deux ensembles de données liées S et T (soit du même domaine ou de différents domaines) et une relation R (soit *owl :sameAs* ou bien une autre relation), le but de la tâche de découverte des liens est de trouver toutes les paires $(s,t) \in S \times T$ telle que la relation entre eux $R(s,t)$ existe. Le résultat est un ensemble de mappings $M_{S\times T} = \{(s,R,t)|s \in S, t \in T\}$. Optionnellement, une valeur de similarité $0 \leq sim(s,t) \leq 1$ qui est calculée par un outil de découverte des liens peut être ajoutée aux entrées du processus de mapping pour exprimer la confiance d'un lien calculé. Dans ce cas, un mapping peut être représenté comme suit : $M_{S\times T} = \{(s,R,t,sim(s,t))|s \in S, t \in T\}$.
Le problème de découverte des liens est très similaire au problème de la résolution d'entités (qu'elle est appelée aussi la déduplication, la réconciliation de références ou le matching d'objets) qui a été déjà largement abordé (Elmagarmid et al., 2007), (Christen, 2012). En particulier, les techniques de résolution d'entités peuvent être utilisées dans le processus de découverte des liens dans Linked Data. Cependant, il existe des différences significatives entre la découverte des liens et la résolution d'entités, ce qui conduit au développement d'outils spécifiques pour la découverte des liens (Nentwig et al., 2015). Le tableau 4.2 illustre la différence entre la découverte des liens et la résolution d'entités.

La plupart des approches de résolution d'entités se concentrent sur des ensembles de données homogènes relativement simples et des objets structurés qui sont décrits par un ensemble d'attributs à valeur unique. En revanche, les approches de découverte des liens se concentrent sur des ressources hétérogènes et fortement inter-reliés dans les ensembles de données liées. En particulier, les ressources peuvent être : des instances, des propriétés ou les classes d'une ontologie. Ainsi, le processus de découverte des liens (voir la figure 4.1) comporte habituellement deux parties essentielles : une partie qui matche les classes et les propriétés des ontologies et une partie qui

matche les instances des ontologies. En outre, les techniques de résolution d'entités se concentrent sur la recherche des objets qui sont sémantiquement équivalents tandis que les techniques de découverte des liens visent à identifier plusieurs types de relations (y compris *owl :sameAs* et d'autres relations). Les approches de découverte des liens peuvent faire une exécution hors ligne classique, comme elles peuvent exécuter des requêtes en ligne pour intégrer des données provenant de plusieurs sources de données.
Typiquement, la résolution du problème de découverte de liens est difficile en raison de l'hétérogénéité sémantique et le grand volume des ensembles de données liées, ce qui rend difficile de répondre à des exigences majeures telles que l'efficacité et la performance. Une efficacité élevée exige de trouver tous les liens corrects entre les ensembles de données liées. La performance exige de découvrir les liens entre un grand nombre de ressources dans un temps très efficace. Atteindre une haute efficacité exige généralement des spécifications de liaisons complexes, une combinaison de plusieurs mesures de similarité et une configuration adéquate des paramètres tels que les seuils de similarité. La spécification manuelle de telles configurations est très difficile et prend du temps, donc ces configurations doivent être automatisées pour réduire l'effort de la spécification des liaisons.

4.4 Workflow de la découverte des liens

Actuellement, plusieurs approches de découverte des liens suivent plusieurs étapes pour trouver des liens entre les ensembles des données liées. La figure 4.1 illustre le workflow générique de la découverte des liens. Ce workflow reçoit en entrée deux ensembles de données liées (un ensemble de données source et cible), un fichier de configuration et des sources de connaissances externes (optionnelles). Les données d'entrée peuvent être des fichiers RDF, OWL, CSV (pour une exécution hors ligne) ou des requêtes SPARQL (pour une exécution en ligne). En particulier, les ressources des ensembles de données d'entrée peuvent être limiter, par exemple, un systéme de découverte de liens ne doit pas de comparer une source de données qui contient des ressources médicales avec une source de données qui contient des ressources sur les films.
Le fichier de configuration d'entrée est un fichier XML qui contient des informations sur les règles de comparaison de ressources, les mesures et les seuils de similarité, les ensembles de données sources et cibles et le type de relations générées en sortie. La figure 4.2 présente un exemple d'un fichier de configuration qui peut être utilisé par l'outil LIMES. La configuration manuelle est un problème majeur dans la tâche

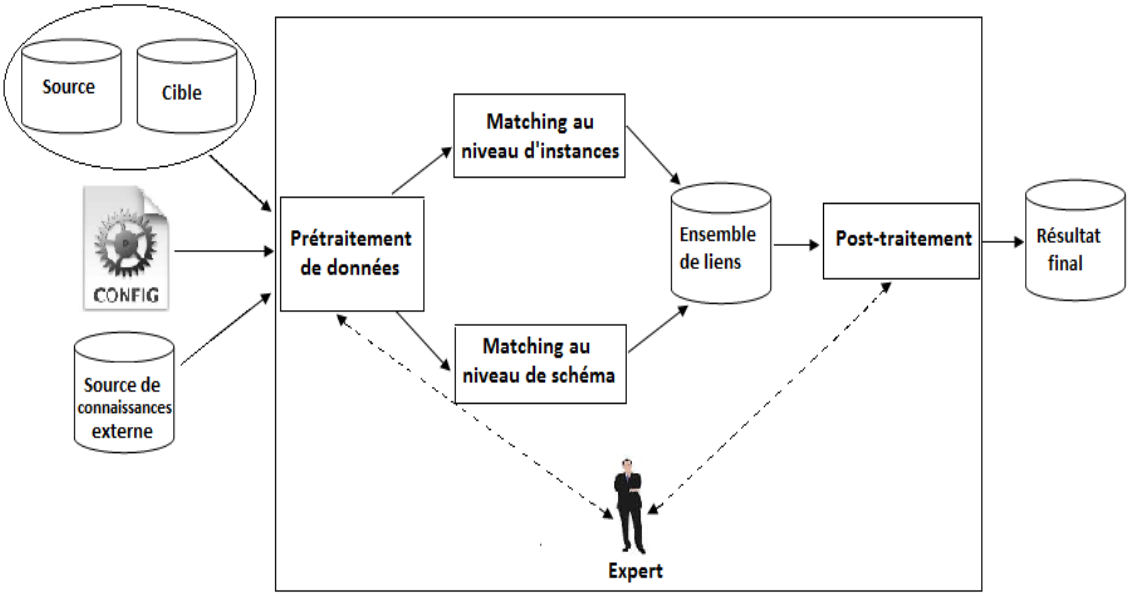

Figure 4.1 – Le workflow de la découverte des liens

de découverte des liens parce qu'elle est difficile à réaliser même par des experts de domaines. Donc, il est souhaitable d'automatiser certaines décisions comme le choix de mesures et de seuils de similarité. En option, les outils de découverte des liens peuvent réutiliser des mappings qui sont déterminés précédemment ou recevoir des sources de connaissances en entrée, comme les thésaurus. En se basant sur les règles de transitivité, les outils de découverte des liens peuvent utiliser des liens ou des mappings qui existent déjà pour trouver de nouveaux liens (par exemple, ils peuvent composer plusieurs liens *owl :sameAs* pour obtenir de nouveaux liens *owl :sameAs*) (Hartung et al., 2013). Des mapping publics référentiels tels que BioPortail (Noy et al., 2009) ou LinkLion (Nentwig et al., 2014) soutiennent la publication des liens et donc leurs réutilisations pour la détermination de nouveaux liens.

Le workflow de la découverte des liens comporte trois phases essentielles : le pré-traitement des données, le matching (en calculons les similarités) et le post-traitement des résultats (Nentwig et al., 2015).

Le pré-traitement permet de finaliser la spécification de liaison qu'elle est mentionnée dans le fichier de configuration et d'améliorer le temps d'exécution, par exemple en réduisant l'espace de recherche et par conséquent le nombre de comparaisons dans le processus de matching. En fait, la réduction de l'espace de recherche peut être faite par des approches de blocage et de filtrage. Le blocage permet de partitionner les ensembles de données d'entrées en plusieurs partitions ou blocs de telle façon que les liens sont déterminés entre les ressources de la même partition. Le filtrage

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSpy v2010 rel. 3 sp1 (http://www.altova.com)-->
<!DOCTYPE LIMES SYSTEM "limes.dtd">
- <LIMES>
- <PREFIX>
    <NAMESPACE>http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/</NAMESPACE>
    <LABEL>drugbank</LABEL>
  </PREFIX>
- <SOURCE>
    <ID>dailymed</ID>
    <ENDPOINT>http://lgd.aksw.org:5678/sparql</ENDPOINT>
    <GRAPH>http://www.instancematching.org/oaei/di/dailymed/</GRAPH>
    <VAR>?x</VAR>
    <PAGESIZE>1000</PAGESIZE>
    <RESTRICTION>?x rdf:type dailymed:drugs</RESTRICTION>
    <PROPERTY>dailymed:activeIngredient/rdfs:label AS lowercase</PROPERTY>
  </SOURCE>
- <TARGET>
    <ID>drugbank</ID>
    <ENDPOINT>http://lgd.aksw.org:5678/sparql</ENDPOINT>
    <GRAPH>http://www.instancematching.org/oaei/di/drugbank/</GRAPH>
    <VAR>?y</VAR>
    <PAGESIZE>1000</PAGESIZE>
    <RESTRICTION>?y rdf:type drugbank:drugs</RESTRICTION>
    <PROPERTY>rdfs:label AS lowercase</PROPERTY>
  </TARGET>
  <METRIC>trigrams(x.dailymed:activeIngredient/rdfs:label, y.rdfs:label)</METRIC>
- <ACCEPTANCE>
    <THRESHOLD>0.9</THRESHOLD>
    <FILE>examples/latc/finals/drugbank-dailymed-activeIngredients.ttl</FILE>
    <RELATION>dailymed:activeIngredient</RELATION>
  </ACCEPTANCE>
- <REVIEW>
    <THRESHOLD>0.9</THRESHOLD>
    <FILE>examples/latc/drugbank-dailymed.toReview.ttl</FILE>
    <RELATION>dailymed:activeIngredient</RELATION>
  </REVIEW>
  <EXECUTION>Simple</EXECUTION>
  <GRANULARITY>16</GRANULARITY>
  <OUTPUT>TTL</OUTPUT>
</LIMES>
```

Figure 4.2 – Un exemple d'un fichier de configuration

utilise les mesures et le seuil de similarité pour filtrer les paires de ressources qui ne peuvent pas répondre à la condition de similarité.
En outre, le pré-traitement permet de transformer et de nettoyer les données d'entrée (par exemple, la résolution des abréviations). Bien que le matching soit complètement automatique, il peut avoir une interaction de l'utilisateur (spécialement par des experts des domaines), par exemple pour vérifier les liens calculés avec une confiance plus faible. La phase principale du workflow de découverte des liens est le matching. Elle applique les spécifications de liaisons qui sont définies dans le fichier de configuration. En particulier, elle évalue les mesures de similarité spécifiées sur les paires de ressources. Des outils de découverte des liens possèdent une bibliothèque de différentes techniques de matching qui appliquent une mesure de similarité sur les ressources d'entrée pour relier les uns aux autres. Typiquement, une ressource peut être considérée selon son contexte (c'est-à-dire, comme des instances liées ou comme un contexte ontologique) ou évaluée comme une valeur de propriété atomique (c'est-à-dire, littérale). Les techniques de matching peuvent être classées à base de

structure ou d'élément selon la nature de la ressource. Les techniques de matching à base d'élément utilisent des mesures de similarité à base de chaînes de caractères (par exemple Levenshtein, n-gram, TF/IDF, Jaccard...etc) (Cheatham and Hitzler, 2013), des nombres ou des types de données spécifiques à un domaine (par exemple les coordonnées géographiques). Le calcul de similarité peut également utiliser différents types de sources de connaissances comme des dictionnaires spécifiques à un domaine et des thésaurus.
Les techniques de matching à base de structure (ou du contexte) sont des techniques sophistiquées qui visent à déduire la similarité des ressources à partir de la similarité de leurs contextes. Plusieurs approches appliquent des fonctions de calcul de similarité en se basant sur le contexte. Par exemple, certaines approches utilisent les entités qui sont très similaires pour trouver les entités correspondantes dans l'ensemble de leurs entités apparentées (Jimenez-Ruiz and Grau, 2011), (Huber et al., 2011). La recherche des correspondances peut aussi se limiter à des instances de classes équivalentes ou connexes en utilisant le contexte ontologique (c'est-à-dire, deux classes sont considèrées similaires si elles partagent les mêmes instances).
La dernière phase du workflow de découverte des liens permet de combiner les résultats de matching et de sélectionner les liens qui répondent aux conditions de la spécification de liaisons décrite dans le fichier de configuration. Les liens résultants peuvent être affinés ou réparés pour éviter les incohérences. Nous pouvons trouver des instances de l'ensemble de données source qui sont liées à plusieurs instances de l'ensemble de données cible. Par conséquent, le post-traitement pourrait appliquer une restriction en sélectionnant le meilleur lien, par exemple, avec la valeur de confiance la plus élevée. Les évaluations manuelles, qui sont faites par des experts de domaines, sont très utiles pendant le post-traitement afin de vérifier l'exactitude des liens calculés.

4.5 Utilisation des espaces métriques pour la recherche d'informations

Généralement, le problème de la recherche est limité par le type de données stockées dans les ensembles de données, la méthode de comparaison d'instances et la spécification de la requête par laquelle les utilisateurs expriment leurs besoins d'informations (Zezula et al., 2006). Traiter les données qui se trouvent dans les bases de connaissances en tant qu'objets métriques est un grand avantage dans le domaine de la recherche d'informations, parce que plusieurs classes de données sont conformes à la vue métrique. En fait, on peut construire des règles de partitionnements précises

en se basant sur des fondements mathématiques sous-jacentes de la notion d'espace métrique. L'utilisation de ces règles est très importante dans le cas ou les coûts d'exécution de la requête sont liés aux Entrées/Sorties et au CPU.
Déterminer la proximité, la similarité et la dis-similarité d'un objet de requête par rapport aux objets stockés dans une base de connaissances est très nécessaire pour la recherche d'informations. Une abstraction utile pour la proximité est fournie par la notion mathématique de l'espace métrique (Zezula et al., 2006). Un espace métrique permet de définir la distance entre les éléments d'un ensemble de données.
Définition 2 : "Soit D un domaine, d une mesure de distance sur D, et (D, d) un espace métrique. Etant donné un ensemble $X \subseteq D$ de n éléments, prétraiter ou structurer les données de sorte que les requêtes de proximité soient répondues efficacement" (Zezula et al., 2006).
Pratiquement, X peut être considéré comme un ensemble de données qui prend ses valeurs du domaine D et d est la mesure de proximité (c'est-à-dire, la fonction de distance définie pour une paire d'objets du domaine D). Dans un espace métrique $M = (D, d), d : D \times D \rightarrow R$ est une fonction telle que pour tout $x, y, z \in D$, on aura les caractéristiques suivantes :

1. La non-négativité : $\forall x,\, y \in D,\, d(x,\, y) \geq 0$;
2. La symétrie : $\forall x,\, y \in D,\, d(x,\, y) = d(y,\, x)$;
3. L'identité : $\forall x, y \in D,\, x = y \Leftrightarrow d(x,\, y) = 0$; qui est elle-même définie comme suit :
   (a) La réflexibilité : $\forall x, y \in D,\, d(x, x) = 0$;
   (b) La positivité : $\forall x, y \in D,\, x \neq y \rightarrow d(x, y) \geq 0$;
4. L'inégalité triangulaire : $\forall x,\, y,\, z \in D,\, d(x,\, z) \leq d(x,\, y) + d(y,\, z)$.

Nous parlons d'un espace quasi métrique si la distance entre deux points x et y est asymétrique (c'est-à-dire, $d(x, y) \neq d(y, x)$). En fait, il existe des techniques qui permettent de transformer une relation asymétrique en une relation symétrique. Par exemple : $d_{sym}(x, y) = d_{asym}(x, y) + d_{asym}(y, x)$.
Un espace est appelé pseudo-métrique si la distance entre x et y ne satisfait pas la propriété de positivité. Finalement, l'inégalité triangulaire peut être utilisée pour réduire le nombre de comparaisons nécessaires pour accomplir la tâche du matching.

4.6 Architecture proposée pour la découverte des liens

Comme le nombre de sources de données disponibles dans le LOD Cloud augmente rapidement, il est nécessaire de développer des outils de découverte des liens dans un temps efficace. Dans cette section, nous présentons une approche sans perte (Bencherif et al., Mai 2016) qui permet de découvrir des liens typés *owl :sameAs* entre les ensembles de données du LOD Cloud dans un temps très efficace. Notre approche vise à améliorer le framework LIMES (Ngonga and Auer, 2011) qui utilise l'inégalité triangulaire d'un espace métrique pour estimer la similarité entre les instances. En particulier, nous réduisons l'espace de recherche en utilisons l'inégalité triangulaire et l'indexation pour améliorer la complexité temporelle de notre algorithme. En outre, nous utilisons le Wordnet et plusieurs mesures de similarités syntaxiques pour améliorer la précision et le rappel de notre système. La figure 4.3 illustre l'architecture générale de notre système.

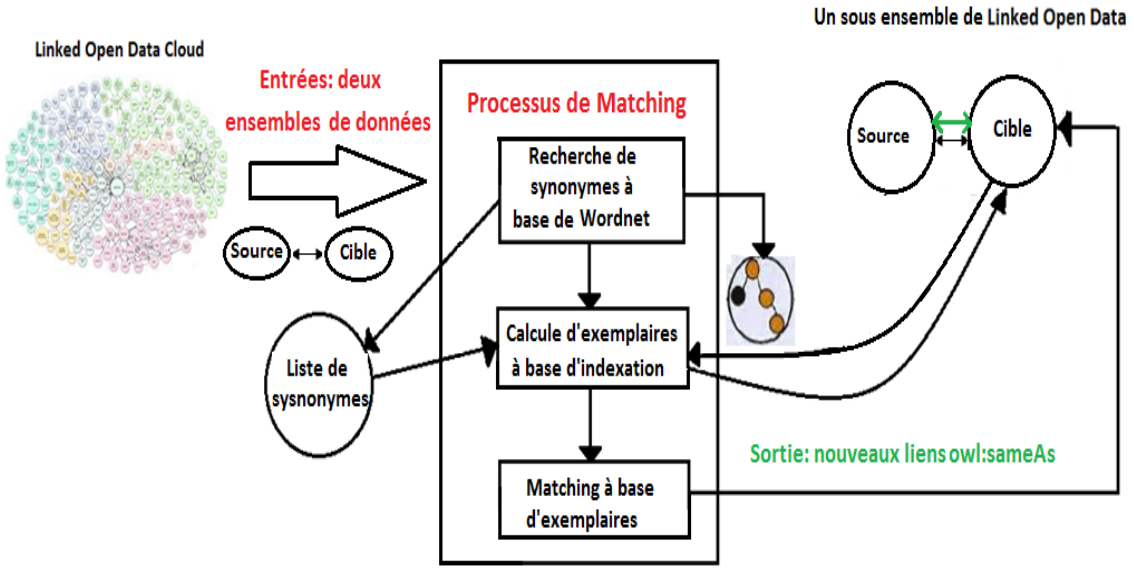

Figure 4.3 – L'architecture de notre système de découverte des liens

Tout d'abord, nous utilisons des indices inversés structurés pour obtenir les meilleurs candidats de matching pour chaque instance source. Comme les indices ne contiennent pas de synonymes, nous utilisons le Wordnet pour améliorer le résultat d'indexation et par conséquent l'efficacité de notre système. Ensuite, notre système utilise le résultat d'indexation pour calculer un ensemble d'exemplaires. Ainsi, il mappe chaque point de l'ensemble d'indexation à l'exemplaire le plus proche à lui.

Ensuite, nous calculons la distance entre chaque instance source et chaque exemplaire, et on approxime la distance entre chaque instance source et chaque élement de l'ensemble d'indexation. Finalement, notre système calcule le résultat de matching final telle que la distance entre une instance source et une instance cible est superièure à un seuil donné.

4.6.1 Utilisation de l'inégalité triangulaire pour approximer la distance

Le but principal de notre approche est de minimiser le nombre de comparaisons entre deux grands ensembles de données liées. l'inégalité triangulaire d'un espace métrique peut être utilisée pour réduire le nombre de comparaisons dans le processus de matching. Pour cela, on considère les ensembles de données source et cible comme des espaces métriques (M, d). Donc, $\forall x, y, z \in M, d(x, z) \leq d(x, y) + d(y, z)$ ...(1)
L'inégalité triangulaire implique également :
$\forall x, y, z \in M, d(x, y) \leq d(x, z) + d(z, y)$ ...(2)
De (1), nous pouvons déduire l'inégalité suivante : $d(x, z) - d(y, z) \leq d(x, y)$.
Et comme la distance $d(y, z)$ est symétrique (c'est-à-dire, $d(y, z) = d(z, y)$), nous pouvons également déduire que : $d(x, z) - d(z, y) \leq d(x, y)$ ...(3).
Les inégalités (2) et (3) conduisent à la condition bornée suivante :
$d(x, z) - d(z, y) \leq d(x, y) \leq d(x, z) + d(z, y)$ ...(4)
L'inégalité (4) a deux implications principales (Ngonga and Auer, 2011) :
1) Nous pouvons approximer la distance entre les points x et z $(d(x, z))$, si nous connaissons la distance entre x et un point de référence y $(d(x, y))$ ainsi que la distance entre le point de référence y et le point z $(d(y, z))$. Le point de référence y est appelé un exemplaire qu'il est utilisé comme un échantillon d'une portion de l'espace métrique (M, d). Ainsi, connaitre la distance entre un point x et n'importe quel exemplaire y permet de calculer des bornes inférieures et supérieures de la distance entre x et n'importe quel point z. Par conséquent, l'inégalité (4) permet d'approximer la distance entre x et y $(d(x, y))$. Par exemple, soit les points x, y, z, w présentés dans la figure 4.4. Nous pouvons déduire l'inégalité suivante : $d(x, y) - d(y, z) \leq d(x, z)$ et $d(x, w) \leq d(x, y) + d(y, w)$, en utilisant la borne inférieure ainsi que la borne supérieure de l'inégalité (4).

2) La deuxième implication de l'inégalité (4) signifie que la distance entre x et z pourrait être inférieure à un seuil θ, si la borne inférieure de l'approximation de la distance entre x et z via l'exemplaire y est inférieure à θ (Ngonga and Auer, 2011). Formellement, $d(x, y) - d(y, z) \leq \theta \rightarrow d(x, z) \leq \theta$. Ainsi, si la borne inférieure de l'approximation de la distance entre x et z via l'exemplaire y est supérieure à θ,

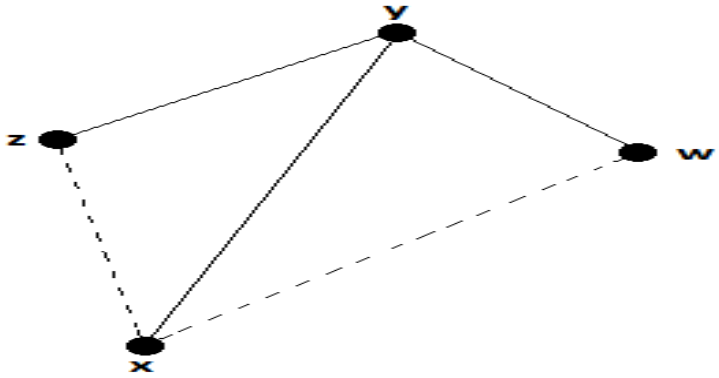

Figure 4.4 – Un exemple d'approximation de distances via l'exemplaire y

alors la distance entre x et z est supérieure à θ. Formellement, $d(x,y) - d(y,z) \geq \theta \rightarrow d(x,z) \geq \theta$. Donc, cette inégalité peut être utiliser pour réduire le nombre de comparaisons.

4.6.2 Utilisation du Wordnet pour rechercher les synonymes

Le Wordnet (Miller, 1995) est une base de données lexicale qui regroupe les mots anglais dans des ensembles de mots équivalents appelés synsets. Un exemple de synsets pour le mot "car" est :
car, auto, automobile, machine, motorcar – (4-wheeled motor vehicle ; usually propelled by an internal combustion engine ; he needs a car to get to work).
Il donne une brève définition, des exemples, des homonymes et les hyponymes de ces synonymes. Dans la littérature, il existe des versions de Wordnet en plusieurs langues, mais sa version anglaise est la plus complète jusqu'à ce jour [4] . Comme mentionné précédemment, le Wordnet peut être considéré comme un mélange d'un thésaurus et un dictionnaire, car il contient des concepts qui sont interconnectés par des relations sémantiques. En effet, il établit la distance sémantique entre deux concepts en fournissant six mesures de similarité et trois mesures de connexité (voir l'annexe B). Généralement, les applications modernes utilisent le Wordnet pour augmenter le nombre de données extraites par des systèmes de récupération d'informations et améliorer leurs précisions (Buscaldi and Rosso, 2009). Malgré son importance, les outils de découverte des liens n'intègrent pas le Wordnet dans leurs frameworks. Nous jugeons utile d'utiliser le Wordnet afin de combler cet écart et améliorer l'efficacité de découverte de liens. Dans cette étape, notre système cherche tous les synonymes de chaque instance source $s \in S$ en utilisant la base de données Wordnet

4. http://dictionnaire.sensagent.leparisien.fr/WordNet/fr-fr/

pour améliorer son efficacité. Puis il met les ressources résultant dans une liste pour les utiliser dans l'étape suivante.

4.6.3 L'indexation à base de Wordnet

Un système de recherche sémantique est un système de récupération d'informations qui effectue la mise en correspondance des requêtes et les résultats potentiels à un niveau conceptuel, il permet la recherche sur des milliards de documents stockés sur des millions d'ordinateurs (Jeffrey et al., 2010). Dans le contexte du web sémantique, il est nécessaire de récupérer des informations provenant de diverses sources pour interroger des données liées à différents niveaux de granularité. En fait, l'utilisateur saisit une requête qui se compose d'un ensemble de mots clés et attend en sortie une liste de pages web classées. La plupart des moteurs de recherche utilisent des indices inversés pour trouver des résultats potentiels d'une requête donnée. En outre, ces indices ne contiennent pas de synonymes et ne peuvent pas différencier entre les homonymes. Comme résultat de la requête d'entrée, l'utilisateur reçoit différents résultats de recherche lorsqu'ils utilisent différentes formes de conjugaison du même mot. Avec le croisement du mouvement LOD, nous voulons indexer un grand nombre de données structurées qui représentent des centaines de millions d'entités. Le but de cette étape est de récupérer rapidement les URIs correctes (en donnant une requête) à partir de grands ensembles de données disponibles dans le LOD Cloud. Pour cela, nous utilisons un indice pour récupérer les entités qui ont une petite distance avec les mots clés de la requête d'entrée en utilisant des mesures de similarité. Donc, notre système indexe l'étiquette qui est utilisée pour nommer l'instance source dans l'ensemble de données cibles pour obtenir une liste de candidats de matching. Il reçoit en entrée les SPARQL endpoint ou les RDF dumps de deux ensembles de données (source et cible) du LOD Cloud ainsi que l'étiquette qui est utilisée pour nommer l'instance source, et il récupère tous les triplets correspondants (à partir de l'ensemble de données cibles). Cela peut, efficacement et à moindre coût, filtrer de nombreuses entités qui ne répondent pas aux conditions de matching.
Afin d'indexer les instances sources dans les bases de connaissances cibles, nous utilisons le moteur de recherche Lucene [5] qui utilise une chaîne représentant une entité pour classer les URIs qui identifient l'entité (Demartini et al., 2013). Dans ce contexte, chaque triplet RDF peut être vu comme une phrase simple qui connecte un sujet (une URI), une propriété (une URI) et un objet (qui peut être soit une URI ou un littéral). Par exemple, le triplet ci-dessous encode le fait que la personne Tim_Berners-Lee (qui est identifiée par l'URI <http ://dbpedia.org/resource/Tim_Berners-

5. http://lucene.apache.org/

Lee> est né en 1955).

```
prefix db:<http://dbpedia.org/resource/>.
db:Tim_Berners-Lee <http://dbpedia.org/ontology/birthDate> 1955-06-08.
```

Nous utilisons deux indices durant la phase d'indexation :
1) Le premier indice contient tous les composants d'un triplet. Nous considérons chaque profile d'entité comme un document qui contient toutes les versions des noms d'entités attachées à lui dans le graphe. Afin de créer un indice inversé structuré, nous indexons séparément différents types d'information attachés à l'entité. Spécifiquement, nous créons un indice inversé structuré pour les trois pièces d'informations suivantes (Tonon et al., 2012) :
URI : nous segmentons l'URI qui est utilisée pour identifier une entité. Cette pièce correspond souvent à la requête de l'utilisateur.
Étiquette : nous sélectionnons manuellement une liste de propriétés qui pointent vers une étiquette ou une description textuelle de l'entité. Généralement, les propriétés qui se répètent fréquemment dans les ensembles de données liées du LOD sont : *label, title, name, full-name, family-name, given-name, has-pretty-name, prefLabel, given-name, nickname.*
Attributs : les autres propriétés qui n'appartiennent pas aux pièces d'informations précédentes sont considérées comme des attributs.
2) Le deuxième indice contient les synonymes de chaque ressource. Comme les indices ne contiennent pas de synonymes, nous utilisons le Wordnet pour chercher tous les synonymes de la requête clé pour étendre et améliorer le résultat d'indexation. Donc, notre système indexe l'instance source et ses synonymes dans l'ensemble de données cible pour récupérer toutes les entités correspondantes.

4.6.4 Calculer l'ensemble d'exemplaires en se basant sur le résultat d'indexation

L'objectif principal de cette approche est de réduire le nombre de comparaisons dans le processus de matching et par conséquent améliorer la complexité temporelle de notre système. Pour atteindre cet objectif, nous avons utilisé plusieurs méthodes qui permettent de réduire l'espace de recherche au cours du processus de matching. Dans l'étape précédente, nous avons indexé les instances sources dans l'ensemble de données cible pour filtrer de nombreuses entités qui ne répondent pas aux conditions

de matching. Alors, cette étape permet efficacement de réduire l'espace de recherche en traitant sur les entités candidates pour le matching. Comme les ensembles de données sources contiennent une grande quantité d'informations, nous ajoutons une autre méthode qui permet également de calculer un ensemble d'exemplaires de l'ensemble de donnée cible. Pour cela, nous considérons le résultat d'indexation comme un espace métrique et nous calculons un ensemble d'exemplaires de manière à ce qu'ils soient distribués de façon uniforme dans l'espace métrique (Ngonga and Auer, 2011). Cela permet de déterminer les entités les plus dissimilaires à partir du résultat d'indextation. Lorsque le premier algorithme de LIMES ajoute un exemplaire à la liste d'exemplaires, il ne supprime pas cet exemplaire de l'ensemble de données cible et donc il peut y avoir une duplication des exemplaires et par conséquent le nombre de comparaisons sera augmenté. Pour résoudre ce problème, nous supprimons chaque exemplaire de l'ensemble d'indexation. Cela peut efficacement réduire le nombre de comparaisons et donc améliorer la complexité temporelle.
L'algorithme 1 traite les trois étapes précédentes. Il reçoit en entrée un point $s \in S$ et un ensemble de données cible. Tout d'abord, notre algorithme cherche tous les synonymes de chaque instance source $s \in S$ et il les met dans une liste $v = v \cup s$ (étape 1). Ensuite, il indexe les éléments de la liste v dans l'ensemble de données cible pour récupérer tous les candidats de matching à partir de l'ensemble de données cible et il les met dans une liste T' (étape 2). Dans les étapes 3, 4 et 5, l'algorithme choisis un point aléatoire e_1 à partir du résultat d'indexation T' pour initialiser l'ensemble d'exemplaires et il met $E = e_1$, puis il enlève e_1 de T' pour supprimer les doublons. Ensuite, la similarité entre e_1 et chaque $t \in T'$ est calculée (étape 6). Pour calculer la similarité entre deux entités, nous utilisons des mesures de similarités syntaxiques telles que la mesure de Levenshtein, la mesure n-gram, etc. Tant que la taille de l'ensemble d'exemplaires E est inférieure à une valeur donnée n, notre algorithme sélectionne un exemplaire $e' \in E$ telle que la distance totale entre e' et chaque $e \in E$ est maximale. Ensuite, il ajoute ce point à la liste d'exemplaires E et il le retirons de l'ensemble T' pour éviter la redondance d'exemplaires (étape 8 et 9). Dans l'étape 10, la distance de e' à chaque $t \in T'$ est calculée. Enfin, chaque instance $t \in T'$ est mappée avec l'exemplaire le plus proche de lui (étape 11).
La complexité temporelle de cet algorithme est $O(|E|(|T'| - |E|))$ où :
$|E|$ est la taille de l'ensemble d'exemplaires ;
$|T'|$ est la taille de l'ensemble qui contient le résultat d'indexation.

Cependant, le choix correct du nombre d'exemplaires $|E|$ conduit à un nombre beaucoup plus faible de comparaisons.

Algorithm 1 Le calcul d'exemplaires en se basant sur le résultat d'indexation

Entrées : Point $s \in S$, nombre d'exemplaires n, le Wordnet, l'ensemble de données cible T

Sorties : L'ensemble d'exemplaires E et leurs correspondances dans T

1. Chercher tous les synonymes de s dans le Wordnet et les mettre dans la liste $v = v \cup \{s\}$.
2. Indexer les éléments de v dans T et mettre le résultat dans T'.
3. Choisir un point aléatoire $e_1 \in T'$;
4. Initialiser $E = E \cup \{e_1\}$;
5. **Supprimer e_1 de T' ;**
6. Calculer la similarité entre e_1 et tout $t \in T'$;

Tant que $|E| < n$ faire

7. Choisir un exemplaire e' / $e' \in argmax_t \sum_{t \in T'} \sum_{e \in E} d(t, e)$
8. $E = E \cup \{e'\}$;
9. **Supprimer e' de T'**
10. Calculer la similarité entre e' et tout $t \in T'$;

fin

11. Mapper chaque instance $t \in T'$ à un exemplaire $e \in E$ / $d(t, e)$ est minimale ;
12. Retourner E ;

4.6.5 Résultat final du matching

Dans cette étape, nous réutilisons le second algorithme de LIMES (Ngonga and Auer, 2011). Pour faire le matching final, nous utilisons l'inégalité triangulaire pour approximer les distances. Pour cela, nous stockons le résultat de la septième étape de l'algorithme 1 (qui consiste à mapper des instances qui se trouvent dans l'ensemble T' à l'ensemble d'exemplaires E) dans une liste L_e qui est triée par ordre décroissant par rapport à leur distance à e. Donc, la liste L_e regroupe les éléments suivants : $L_e = \lambda_1^e, \lambda_2^e, ..., \lambda_i^e$.

Pour trouver la liste de matching entre les instances sources $s \in S$ et les instances cibles $t \in T$:

a) L'algorithme calcule la distance entre le point s et chaque point $e \in E$, $(d(s, e))$. Ensuite, il vérifie si cette distance est inférieure ou égale à un seuil donné θ (Formellement, $d(s, e) \leq \theta$).

b) L'algorithme approxime la distance entre le point s et chaque point $t \in (T' - E)$ via l'exemplaire e, $(d(s, t))$. Ensuite, il vérifie si cette distance est inférieure ou égale à un seuil donné θ (Formellement, $d(s, e) - d(e, \lambda_i^e)$). L'algorithme 2 (Ngonga and Auer, 2011) montre le calcule de la distance $d(s, e)$ et l'approximation de la distance $d(s, t)$ via l'exemplaire e.

Tout d'abord, notre algorithme initialise la liste de matching M avec une liste vide. Ensuite, il calcule la similarité entre l'instance source s et chaque exemplaire

Algorithm 2 Le matching à base de l'ensemble d'exemplaires

Entrées : L'ensemble d'exemplaires E, point $s \in S$, seuil θ
Sorties : Liste de matching M pour s

1. $M = \emptyset$;
Pour $e \in |E|$ faire
    si $m(s,e) \leq \theta$ alors
        2. $M = M \cup \{e\}$
    fin
    pour $i = 1...|L_e|$ faire
        si $(m(s,e) - m(e,\lambda_i^e)) \leq \theta$ alors
            si $m(s,\lambda_i^e) \leq \theta$ alors
                3. $M = M \cup \{\lambda_i^e\}$
            fin
        sinon
            break ;
        fin
    fin
fin
4. retourner M ;

$e \in E$. Si le score de cette similarité est supérieur au seuil θ ($Sim(s,e) \geq \theta$) alors l'algorithme ajoute l'exemplaire e à la liste de matching M. Ensuite, la similarité entre l'instance source s et tout $t \in L_e$ est approximée. Si le score de cette similarité ($Sim(s,e) - Sim(e,\lambda_i^e)$) ainsi que le score de similarité entre l'exemplaire e et l'instance λ_i^e ($Sim(e,\lambda_i^e)$) sont supérieurs au seuil θ alors l'algorithme ajoute l'instance λ_i^e à la liste de matching M. La complexité temporelle de notre système est $O((|E| + |S|)(|T'| - |E|))$ ou $T' \subseteq T$.

4.6.6 Visualisation des données liées

Avec l'augmentation des ensembles de données disponibles dans le LOD Cloud ; l'interprétation et l'exploitation des données dépasse la capacité humaine (principalement pour les utilisateurs sans expérience avec les technologies du web sémantique). En général, l'application des techniques de visualisation d'informations aux données liées propose une stratégie parfaite pour obtenir une bonne compréhension de sa structure et donc facilite l'accès à l'information par tous les utilisateurs pour découvrir de nouvelles informations.
Le résultat du processus de découverte des liens doit être visualisé pour que l'utilisateur ou l'expert humain puissent détecter toutes les erreurs et vérifier la précision et la complétude des liens. Naturellement, la visualisation des connections entre les ressources est une tâche principale qui aide les utilisateurs finaux à comprendre la

structure principale des ensembles de données et à interpréter les données d'une manière significative.
Actuellement, plusieurs outils ont été développés pour la visualisation des données liées. Ils peuvent être classifiés en deux catégories : la première catégorie affiche des informations textuelles sur les ensembles de données liées et la seconde affiche des interfaces avec des options de visualisation (Aba-Sah and Matthew, 2011). Néanmoins, les outils de visualisation des données liées ne prennent pas en considération la visualisation des connections externes (c'est-à-dire, les relations entre une ressource d'un ensemble de données et des ressources d'un autre ensemble de données). Donc, nous devrons répondre à cette question : *Quand nous sélectionnons une ressource d'un ensemble de données liées, comment elle interagit avec des ressources de l'ensemble de données sélectionné et les ressources des autres ensembles de données du LOD Cloud ?*
Afin de répondre à ce besoin, nous développons un outil appelé LDVT (Bencherif and Malki, Novembre 2016) qui permet de visualiser une ressource d'un ensemble de données et toutes les instances connectées d'une façon dynamique. En outre, il affiche des statistiques sur la ressource d'entrée telles que : le nombre d'objets et de prédicats ainsi que le nombre de liens *owl :sameAs* liés à celui-ci. Notre outil prend en entrée un ensemble de données et une ressource et visualise en sortie un graphe incluant tous les prédicats et les objets liés à la ressource d'entrée. Tout d'abord, nous utilisons une requête SPARQL pour chercher tous les objets liés à la ressource d'entrée et nous les mettons dans un ensemble de noeuds. Ensuite, nous représentons tous les prédicats par un ensemble d'arêtes qui relient la ressource d'entrée avec ses objets. La figure 4.5 présente la visualisation des liens internes et externes d'une ressource.

Après la visualisation, l'utilisateur final peut zoom une zone d'un intérêt particulier. En outre, lorsque l'utilisateur sélectionne un prédicat ou un objet, il apparaît en gras pour clarifier la vision du triplet.

4.7 Evaluation

Selon (Auer et al., 2011), l'efficacité d'un système de découverte des liens consiste à choisir une mesure de similarité et un seuil appropriés. La performance consiste à réduire la complexité temporelle lors du mapping d'un grand nombre de ressources. Dans cette section, l'accent est mis sur l'expérimentation et l'évaluation de notre approche de découverte des liens en utilisant les grands ensembles de données liées. Particulièrement, nous évaluons l'efficacité et la performance de notre système. Ainsi,

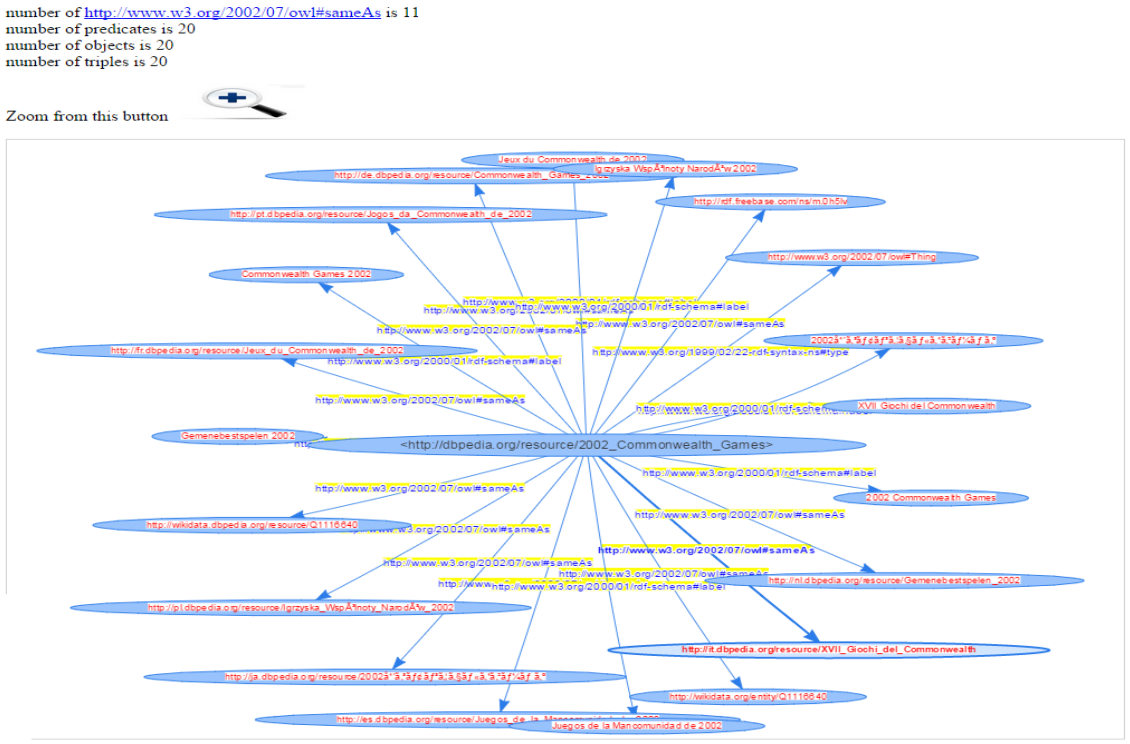

Figure 4.5 – Un exemple de visualisation d'un graphe RDF

nous présentons les résultats de deux expérimentations dans lesquelles nous avons utilisé notre approche de découverte des liens d'identité entre les différents ensembles de données disponibles dans le LOD Cloud. Les expérimentations sont faites sur un système 64 bits avec une machine Intel Core i5 2,5 GHz avec 8 Go de RAM.

4.7.1 Évaluation de la performance

Pour que notre système soit capable de découvrir des liens d'identité entre de très grands ensembles de données liées, il est nécessaire de réduire le nombre de comparaisons pour optimiser le temps d'exécution durant le processus de matching. L évaluation de l'efficacité et la performance de notre système est faite sur quatre ensembles de données réels (DBpedia [6] , LinkedCT [7] , Drugbank [8], Bio2RDF [9]).

La définition des valeurs optimales pour le nombre d'exemplaires permet de minimiser le nombre de comparaisons dans le processus de matching et donc conduit à des temps d'exécution encore plus petits. D'une part, un petit nombre d'exem-

6. http://dbpedia.org/sparql
7. http://data.linkedct.org/sparql
8. http://www4.wiwiss.fu-berlin.de/drugbank/sparql
9. http://mesh.bio2rdf.org/sparql

instance source $s \in S$	$\|T\|$	Le nombre de candidats pour le matching	Le nombre de comparaisons de notre approche	Le nombre de comparaisons de LIMES	La différence entre notre système et LIMES
<http ://dbpedia.org/resource /People>	4346	20	4	65	61
<http ://dbpedia.org/resource /Albert_Einstein>	4772	15	3	69	66
<http ://dbpedia.org/resource /Organisation>	12701	5	2	112	110
<http ://dbpedia.org/resource /America>	5000	7	2	70	68
<http ://dbpedia.org/resource /Cameroun>	1000	0	0	32	32

Table 4.3 – Comparaisons entre notre système et LIMES en termes de nombre de comparaisons

plaires conduit à une grande approximation de distances et pourrait par conséquent conduire à un plus grand nombre de comparaisons. D'autre part, un grand nombre d'exemplaires conduit à une surcharge considérable pour le calcul d'exemplaires. Cependant, ce processus ne nécessite pas un grand nombre de comparaisons durant le processus de matching.

Le tableau 4.3 illustre une comparaison entre notre approche et LIMES en termes de nombre de comparaisons. Nous calculons le nombre de compaisons de quatre instances de DBpedia avec 4346 instances de Drugbank, 4772 instances de LinkedCT, 12701 instances de DBpedia et 1000 instances de Bio2RDF.

Le framework LIMES utilise toutes les instances de la base de connaissances cible pour calculer le nombre d'exemplaires que nous avons définis par $\sqrt{T}$; tandis que notre système réduit l'espace de recherche en utilisant une portion de la base de connaissances cible pour calculer le nombre d'exemplaires. Cette portion est construite par l'indexation qui récupère pour chaque instance source tous ses candidates de matching à partir de la base de connaissances cible. Comme le résultat d'indexation est classé en utilisant la mesure TF-IDF, nous utilisons les 20 premières instances récupérées pour calculer l'ensemble d'exemplaires. Par exemple, si nous utilisons un ensemble de données composé de 4346 instances de Drugbank, le nombre d'exemplaires calculé par le framework LIMES est $\sqrt{4346} = 65$, et donc chaque instance source sera comparée 65 fois. Alors, le nombre d'exemplaires calculés par notre système est $\sqrt{20} = 4$, c'est à dire, chaque instance source sera comparée uniquement 4 fois. Ainsi, notre système réduit énormément 61 comparaisons par rapport au

Source S	Cible T	$\|S\|$	$\|T\|$	Notre approche	LIMES			SILK
					th=0.80	th=0.90	th=0.95	
DBpedia	LinkedCT	9509	9509	2.43	15	10	6	25
Drugbank	DBpedia	5291	5291	1.28	12	8	5	18
Bio2RDF	DBpedia	50031	74458	3.62	78	52	31	130
Drugbank	LinkedCT	3544	3544	0.99	6	5	2	11

Table 4.4 – Le temps d'exécution de notre système, LIMES et SILK. Tous ces temps sont donnés en secondes

nombre de comparaisons de LIMES. Nous évaluons la performance de notre système en utilisant des ensembles de données réelles pour montrer qu'il améliore le temps d'exécution pendant le processus de mapping. Pour cela, nous avons effectué des expérimentations en utilisant quatre ensembles de données liées, comme le montre le tableau 4.4. Donc, nous trouvons des entités similaires de (DBpedia et LinkedCT), (Drugbank et DBpedia), (Bio2RDF et DBpedia) et (Drugbank et LinkedCT) en comparant leurs étiquettes.

Afin de comparer le temps d'exécution de notre système avec celui de LIMES 0.6 [10] et de SILK 2.6 [11], nous le calculons comme suit :

A) Le temps nécessaire pour récupérer les instances de la base de connaissances source et cible ;

B) Le temps pour indexer les instances sources dans la base de connaissances cible en utilisant le Wordnet ;

C) Le temps nécessaire pour comparer les instances sources avec celles des bases de connaissances cibles ;

D) Le temps d'affichage de résultats finals.

Le tableau 4.4 illustre les différentes tailles des bases de connaissances sources et cibles ainsi que le temps d'exécution de notre système, LIMES et SILK.

Bien que notre système nécessite 9,72% du temps d'exécution de SILK et 16,2% du temps d'exécution de LIMES lors de la comparaison des entités de DBpedia et LinkedCT, il nécessite seulement 3% du temps d'exécution de SILK et 4,64% du temps d'exécution de LIMES lors de la comparaison des entités de Bio2RDF et DBpedia. Cette évaluation montre que notre système est 11 fois plus rapide que SILK et aussi presque 6 fois plus rapide que LIMES.

En conclusion, notre évaluation montre que notre approche conduit à un nombre de comparaisons beaucoup plus faible et un temps d'exécution plus petit que LIMES et SILK.

10. https://github.com/aksw/limes
11. http://silkframework.org

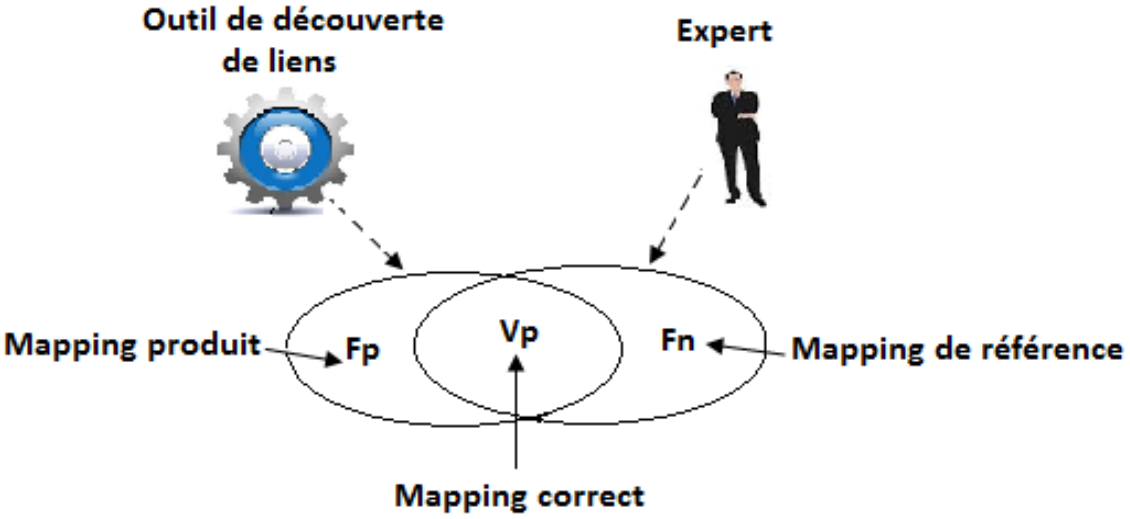

Figure 4.6 – Les phases principales d'un processus d'évaluation de la qualité des mappings

4.7.2 Évaluation de l'efficacité

Dans cette section, nous évaluons expérimentalement l'efficacité de notre système pour la tâche découverte des liens entre des ensembles de données liées. Afin d'atteindre cet objectif, nous sélectionnons un sous-ensemble de DBpedia, Drugbunk et LinkedCT, et nous validons le résultat du processus de mapping manuellement. Les mesures d'évaluation de la qualité de mapping (y compris la précision, le rappel, F-mesure, etc) sont très nécessaires pour mesurer l'efficacité d'un système de découverte des liens. En outre, elles assurent une comparaison entre les méthodes de découverte des liens en se basant sur les résultats obtenus. Généralement, ces mesures sont utilisées dans plusieurs domaines tels que le traitement de requêtes, la recherche d'informations, etc (Zghal, 2010). La figure 4.6 présente les étapes principales d'un processus d'évaluation de la qualités de mappings.

La première étape du processus d'évaluation de la qualité des mappings consiste à référencier à un résultat fournis par un expert humain ou à résoudre le problème manuellement. On considère le résultat de cette étape un mapping de référence. La deuxième étape consiste à utiliser un outil de découverte des liens pour générer automatiquement un ensemble de mapping entre deux ensembles de données liées. Le résultat obtenu automatiquement est considéré comme un mapping produit. La troisième étape du processus d'évaluation de la qualité des mappings consiste à comparer un mapping de référence avec un mapping produit en se basant sur trois ensembles essentiels : le nombre de liens trouvés par l'outil de découverte des liens (mapping produit), le nombre de liens trouvés manuellement ou par un expert (map-

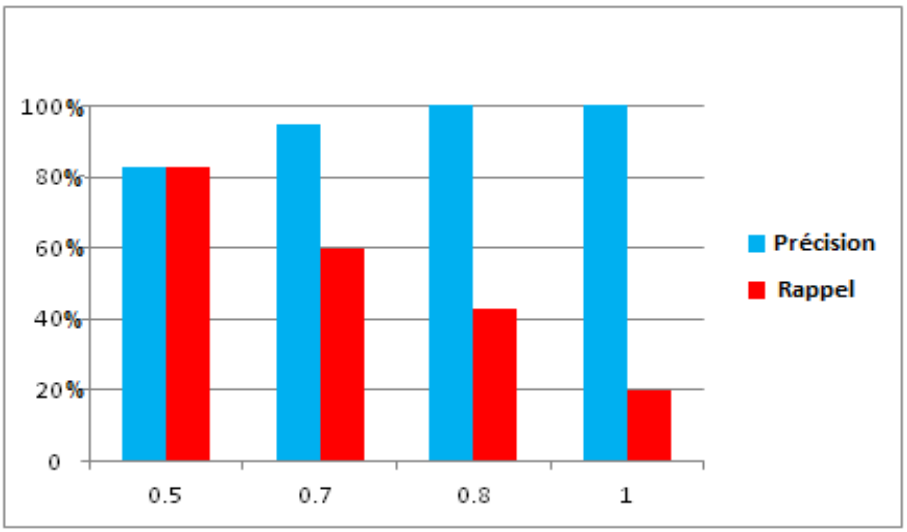

Figure 4.7 – Précision et rappel par rapport au seuil. L'axe x montre le seuil, et l'axe y montre les valeurs de précision et de rappel

ping de référence) et le nombre de liens qui sont apparus dans l'intersection des deux ensembles (l'ensemble de mappings produit et l'ensemble de mappings de référence). Il représente l'ensemble de liens trouvés à la fois par l'outil de découverte des liens et par l'être humain.

Afin de calculer la précision et le rappel de notre système, nous comparons l'ensemble de mappings de références avec ceux de notre système.

La précision représente le rapport du nombre de liens trouvés à la fois par l'être humain et par notre système au nombre de liens trouvés par notre système. Ainsi, elle revoie les liens corrects. Donc, la précision de notre système est définie comme suit :

$$Précision = \frac{Vp}{(Vp+Fp)}$$

Le rappel représente le rapport du nombre de liens trouvés à la fois par l'être humain et par notre système au nombre de liens trouvés par l'être humain. Donc, nous calculons le rappel de notre système comme suit :

$$Rappel = \frac{Vp}{(Vp+Fn)}$$

Où :

Vp (Vrai positif) : le nombre de liens sélectionnés à la fois manuellement et par notre approche.

Fp (Faux positif) : le nombre de liens qui sont sélectionnés par notre approche et qui ne sont pas sélectionnés manuellement.

Fn (Faux négatif) : le nombre de liens qui sont sélectionnés manuellement et qui ne sont pas sélectionnés par notre approche.
En fixant le seuil, nous pouvons obtenir une haute précision ou un rappel élevé. Ainsi, si le seuil s'approche de la valeur 1.0 alors la précision sera maximale bien que le rappel sera faible (car lorsque nous sélectionnons un seuil élevé, le nombre de matchings récupérés par le système est faible). Cela signifie que notre système renvoie beaucoup plus de résultats dans le processus d'appariement en raison de l'utilisation de Wordnet et des mesures de similarités syntaxiques.

4.8 Conclusion

Notre approche permet de réduire le nombre de comparaisons qui est effectué pendant le processus de matching. Pour cela, nous avons utilisé l'indexation à base de Wordnet pour calculer l'ensemble d'exemplaires ainsi que le résultat final du processus de matching. Nous avons évalué la performance de notre système en comparant sa complexité temporelle avec celles de LIMES et de SILK, et en montrant que notre système est le plus performant. De plus, nous avons évalué l'efficacité de notre système en calculant les valeurs de précision et de rappel montrant qu'il est efficace parce qu'il utilise des mesures de similarité sémantiques et syntaxiques.
Suite à ce processus de découverte de liens dans les données liées, nous poursuivons dans le chapitre suivant notre démarche d'intégration, en proposant une approche à base d'un algorithme génétique pour la fusion des données liées.

Chapitre 5

Utilisation des algorithmes génétiques pour la fusion des données liées

Contents

Sourire à votre frère est un acte de charité.

Prophète Mohammad (paix et bénédiction d'Allah soient sur lui)

5.1 Introduction

La capacité de choisir efficacement et rapidement des données hétérogènes à partir du LOD est une étape clé vers le développement des applications des données liées. Actuellement, la plupart de fournisseurs des données liées extraient automatiquement des informations à partir de plusieurs sources de données en utilisant divers extracteurs d'informations (Dong et al., 2014). Avec l'expansion des sources de données qui sont disponibles dans le LOD Cloud, le désir d'accéder à toutes ces sources à travers une interface cohérente et unique a été l'objectif de plusieurs recherches dans le domaine d'intégration des données liées (Mendes et al., 2012). Afin d'atteindre cet objectif, plusieurs systèmes d'intégration des données liées ont été développés. En fait, un système d'intégration des données permet à l'utilisateur d'obtenir une vue complète et concise de toutes les données existantes sans avoir besoin d'accéder à toutes les sources de données séparément (Jens and Felix, 2009). Une vue complète des données consiste à inclure toutes les ressources dans l'ensemble de données résultant, et une vue concise permet d'éviter la duplication des données dans l'ensemble de données résultant et de présenter des informations non contradictoires à l'utilisateur. En effet, obtenir un ensemble de données concise est une tâche difficile car les informations relatives aux entités sont stockées dans plus d'une source. Automatiser le processus de fusion des données liées a été le défi de nombreuses recherches dans le domaine d'intégration des données liées. Les problèmes les plus importants dans ce contexte sont de (1) surmonter l'effet que plusieurs ensembles de données représentent le même objet du monde réel en utilisant différentes entités, (2) choisir les solutions (les ressources) adéquates et correctes pendant la tâche de fusion (c'est-à-dire, développer un système de fusion efficace), et enfin (3) minimiser le temps d'exécution du processus de fusion.
Le reste du chapitre est organisé comme suit : la section 2 décrit le problème de la fusion des données. La section 3 illustre une classification des conflits et les stratégies de la fusion des données. Dans la section 4, nous décrivons en détail notre approche de fusion des données liées. La section 5 présente nos expérimentations ainsi qu'une comparaison entre l'approche proposée et les approches existantes de fusion des données liées.

5.2 Evaluation de la qualité et la fusion des données

Un système d'intégration des données vise à combiner les données hétérogènes et les présenter sous un format unique pour faciliter l'interrogation des sources de données sous-jacentes. Actuellement, la tâche d'intégration des données hétérogènes qui sont disponibles dans le LOD Cloud a attirée l'attention de plusieurs chercheurs dans le domaine de la recherche d'informations. Cependant, le challenge principal dans le contexte d'intégration des données liées est de surmonter l'hétérogénéité sémantique des données en suivant trois étapes : trouver les ressources équivalentes au niveau de schémas, détecter les entités similaires au niveau d'instances ainsi que l'évaluation de la qualité et la fusion des données pour obtenir une représentation complète et consistante.

Comme les fournisseurs des ensembles de données du LOD Cloud utilisent différents vocabulaires pour représenter le même type d'information, le mapping de vocabulaires ou l'alignement au niveau de schémas est devenu une tâche cruciale pour transformer ces informations en un seul schéma cible (Bryl et al., 2014). L'établissement des liens de type *owl :sameAs* entre les ensembles de données liées au niveau d'instances permet d'identifier les différentes URIs qui représentent le même objet du monde réel, par exemple des personnes ou des places.

L'objectif principal de ces deux étapes (le mapping au niveau de schémas et au niveau d'instances) est d'identifier les différentes représentations d'une ressource dans plusieurs ensembles de données liées, on prend leurs résultats comme des entrées pour la fusion des données. L'étape de l'évaluation de la qualité et la fusion des données garantisse la qualité et la consistance de données qui vont être utilisées par des applications de données liées.

L'évaluation de la qualité permet de choisir des données de haute qualité qui ne contiennent pas des informations contradictoires. Généralement, l'interprétation de la qualité des données dépend de ce qui utilise ces informations et pourquoi il a l'intention de les utiliser (c'est-à-dire l'aptitude à l'utilisation) (Juran, 1994). Si un utilisateur considère que des données sont de haute qualité pour une tâche donnée, un autre utilisateur peut considérer que ces mêmes données ne sont pas suffisantes pour une autre tâche. L'aptitude à l'utilisation des données peut dépendre de plusieurs dimensions de la qualité telles que : la complétude (completeness), la concision (conciseness), la consistance (consistency), la précision (accuracy), la pertinence (relevancy), la vérifiabilité (verifiability), la compréhensibilité (understandability), l'objectivité (objectivity), la crédibilité (believability), la disponibilité (availability)

et l'actualité (timeliness)(Bizer and Cyganiak, 2009). Typiquement, seul un sous ensemble de ces dimensions de la qualité des données est pertinent pour une tâche donnée parce qu'elles sont indépendantes les unes des autres. Par conséquent, les consommateurs de l'information déterminent les dimensions et les niveaux de la qualité qui sont requis pour une tâche spécifique (Naumann, 2002). Dans le contexte d'intégration des données, les dimensions de la qualité les plus pertinentes sont la complétude (completeness), la concision (conciseness) et la consistance (consistency). Spécifiquement, un système d'intégration des données vise à augmenter la complétude et la concision des données qui sont disponibles pour les applications et leurs utilisateurs (Jens and Felix, 2009). Une augmentation de la complétude est obtenue en ajoutant plus de ressources (plus d'instances, plus de concepts et plus de propriétés décrivant des objets) au système. Une croissance de la concision est obtenue en supprimant les données redondantes et en fusionnant les entrées en double ainsi que les concepts et les propriétés communes en une seule représentation.
La fusion permet de combiner et de merger les ressources dupliquées en une seule représentation où le problème d'inconsistance des données sera résolu. Dans le contexte d'intégration des données, la fusion des données est définie comme "le processus de fusion de plusieurs enregistrements qui représentent le même objet du monde réel en une représentation unique, consistante et propre" (Jens and Felix, 2009). La fusion des données est l'étape qui suit les étapes de mapping au niveau de schémas et au niveau d'instances, elle est considérée comme un moyen de traitement et de résolution des conflits qui existent dans les sources de données pour détecter les valeurs vraies sous-jacentes. Par exemple, un système de fusion des données génère une seule ressource en fusionnant deux ressources similaires qui existent dans deux ensembles de données liées.

5.3 Classification des conflits et les stratégies de la fusion des données

Comme indiqué précédemment, la fusion des données est le processus qui vise à combiner des informations à partir d'une source de données avec des informations provenant d'autres sources de données. Les informations combinées par le processus de fusion sont ensuite utilisées pour faciliter l'interrogation de ces sources de données. Comme les sources de données utilisent différentes représentations de la même information, (Jens and Felix, 2009) classifie les conflits en trois types principaux :
Des conflits schématiques comportent les conflits qui existent au niveau des ontologies du LOD Cloud, y compris les conflits des concepts et des propriétés.

Des conflits d'identités comportent les conflits qui existent au niveau d'instances qui identifient plusieurs représentations du même objet du monde réel. Ces deux types de conflits sont résolus au cours des deux premières étapes d'intégration des données (c'est-à-dire, le mapping au niveau de schémas et le mapping au niveau d'instances).

Des conflits de données est le troisième type de conflits, ils sont causés par les différentes représentations des mêmes objets du monde réel. Un conflit de données est présenté dans un triplet RDF si les ressources liées à ses composants, qui sont sémantiquement équivalents, sont différentes. L'exemple ci-dessous illustre un conflit de données qui est présenté dans deux triplets RDF de différentes sources de données (DBpedia la version anglaise et português).

Triplet 1 : Juiz_de_Fora	dbedia-owl :areaTotal	$1437km^2$
Triplet 2 : Juiz_de_Fora	dbedia-owl :areaTotal	$1436.850km^2$

Ces deux triplets résultent un conflit de données qui est causé par les différentes représentations des mêmes objets du monde réel (Juiz_de_Fora). Ils comportent différentes informations sur la propriété **dbedia-owl :areaTotal** qui décrit la superficie de **Juiz_de_Fora**. Le premier triplet déclare que la superficie de **Juiz_de_Fora** est $1437km^2$, alors que le deuxième triplet déclare que la superficie de **Juiz_de_Fora** est $1436.850km^2$.

En fait, il existe deux types de conflits de données : l'incertitude sur une ressource (causée par l'information manquante) et les contradictions (causées par différentes valeurs de la même ressource).

L'incertitude : une incertitude est un conflit entre une valeur non nulle et une ou plusieurs valeurs nulles qui sont toutes utilisées pour décrire la même propriété d'un objet (Jens and Felix, 2009). Les valeurs non nulles dans un système de fusion des données sont considérées comme étant des valeurs inconnues et l'évaluation du système reste valable.

La contradiction : une contradiction est un conflit entre différentes valeurs non nulles qui sont utilisées pour décrire la même propriété de la même ressource. En d'autres termes, une contradiction est apparue si deux sources de données fournissent différentes informations sur le même objet. Ceci est également donné par le mapping au niveau de schémas et au niveau d'instances.

Généralement, les incertitudes sont plus faciles à gérer que les contradictions.

Afin de résoudre les différents types de conflits et de les fusionner en une seule présentation, (Jens and Felix, 2009) décrit plusieurs stratégies de fusion des données. La figure 5.1 illustre les types de gestion des conflits. En effet, il existe trois types principaux pour le traitement des conflits (Jens and Felix, 2009) :

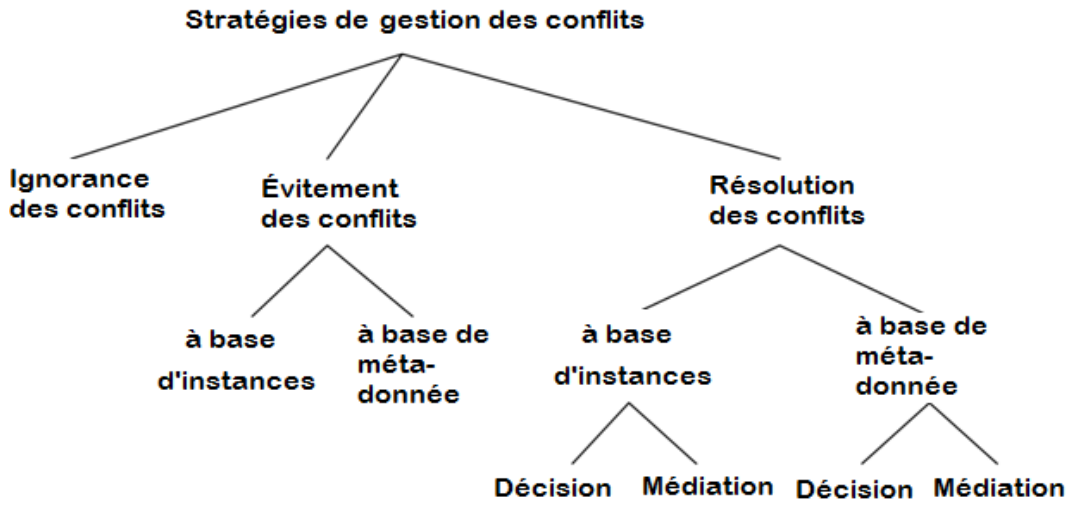

Figure 5.1 – Une classification de stratégies de traitement des conflits (Jens and Felix, 2009)

Les stratégies qui ignorent les conflits ne prennent pas une décision sur les informations contradictoires et parfois elles ne sont pas conscientes aux conflits des données. Un exemple de cette stratégie est la fonction "PASS IT ON" qui présente toutes les valeurs et reporte la résolution du conflit à l'utilisateur et la fonction "CONSIDER ALL POSSIBILITIES" qui crée toutes les combinaisons possibles.
Les stratégies qui évitent les conflits connaissent les conflits qui existent en général, mais ne peuvent pas détecter et résoudre ces conflits dans tous les cas. Un exemple de cette stratégie est la fonction "TRUST YOUR FRIENDS" qui traite les conflits en prenant les valeurs à partir d'une source préférée, la fonction "TAKE THE INFORMATION" qui préfère des valeurs non nulles sur les valeurs nulles et la fonction "NO GOSSIPING" qui retourne seulement les valeurs consistantes.
Les stratégies de résolution des conflits tiennent compte de toutes les données et les métadonnées avant de décider comment résoudre un conflit. On peut distinguer deux stratégies pour la résolution des conflits : des stratégies de décision qui choisissent une valeur parmi les valeurs existantes, et des stratégies de médiation qui choisissent une valeur qui n'existe pas nécessairement parmi les valeurs conflictuelles. Un exemple d'une stratégie de décision est la fonction "CRY WITH THE WOLVES" qui prend la valeur la plus fréquente, la fonction "KEEP UP TO DATE" qui prend la valeur la plus récente et la fonction "ROLL THE DICE" qui prend une valeur aléatoire. Un exemple d'une stratégie de médiation est la fonction "MEET IN THE MIDDLE" qui prend une valeur moyenne.

5.4 Nouvelle approche de fusion des données liées

Avec la croissance du nombre de sources de données qui sont disponibles dans le LOD Cloud, le nombre de valeurs conflictuelles qui existent dans ces sources augmente. En effet, le développement des outils d'intégration des données provenant plusieurs sources est crucial pour que les applications de Linked Data puissent consommer les données à partir de cet espace de données global d'une manière intégrée. Les principaux défis qui apparaissent dans le contexte de fusion des données liées sont 1) le grand volume de données qui existent dans le LOD Cloud et 2) l'utilisation de plusieurs entités pour représenter le même objet du monde réel. Nous proposons une nouvelle approche (Bencherif and Malki, 2016) à base d'un algorithme génétique qui vise à fusionner automatiquement, efficacement et rapidement de grands ensembles de données liées du LOD Cloud qui contiennent des informations dupliquées et contradictoires. La figure 5.2 illustre l'architecture de notre système de fusion des données liées.

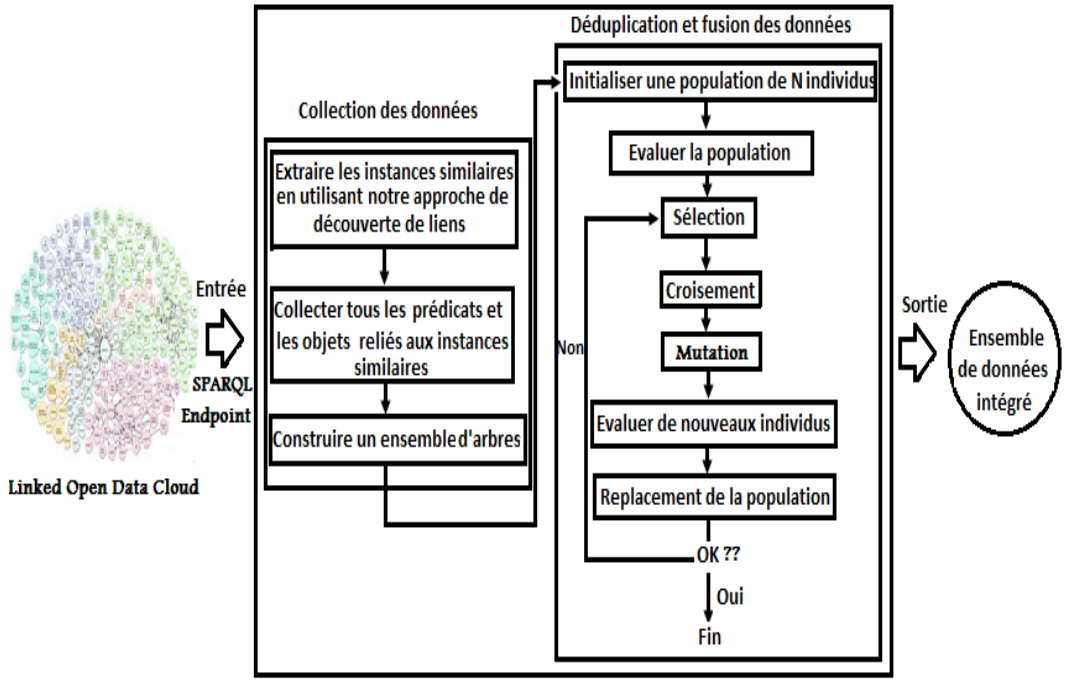

Figure 5.2 – L'architecture de notre approche de fusion des données liées

Dans une première étape, nous extrayons les instances réconciliées à partir du LOD Cloud pour détecter les descriptions des objets similaires (en collectant des instances similaires, nous pouvons identifier différents prédicats et objets qui indiquent des informations identiques ou inter-connectées). Ensuite, nous rassemblons tous les prédicats et les objets qui sont liés aux instances réconciliées pour construire

un ensemble d'arbres. Finalement, comme les algorithmes génétiques sont souvent utilisés pour optimiser la solution d'un problème donné, notre système aborde les problèmes de la fusion des données en proposant un algorithme génétique qui fusionne de façon incrémentale les données qui existent dans les arbres construits ; il doit choisir la manière correcte pour combiner ces données en utilisant une fonction d'évaluation. En outre, l'algorithme proposé tente de minimiser à la fois le temps d'exécution ainsi que le nombre d'instances, de prédicats et d'objets qui existent dans les ensembles de données liées.

5.4.1 Collection des données

Dans cette section, nous présentons trois étapes pour collecter des données à partir du LOD Cloud. Le processus de la collection des données prend en entrée un SPARQL endpoint d'un ensemble de données liées, puis il extrait les instances similaires et collecte tous les prédicats et les objets liés à eux pour construire un ensemble d'arbres. Dans cette recherche, une liste d'arbres forme notre espace de recherche ou chaque arbre est composé d'un sujet, un ensemble de prédicats et un ensemble d'objets.

Extraction des instances similaires

Les frameworks de découverte des liens permettent de découvrir les instances similaires qui existent dans le LOD Cloud. Généralement, les instances qui se rapportent à la même chose sont liées par le lien typé *owl :sameAs*. Les instances similaires aident à construire un ensemble d'arbres et identifier différents prédicats et objets qui indiquent des informations identiques ou connexes (Zhao and Ichise, 2013). Par conséquent, si nous fusionnons les instances similaires, les prédicats et les objets connexes, nous obtiendrons une vue unique et ainsi nous pouvons efficacement et rapidement interroger l'ensemble de données généré. Dans cette étape, nous utilisons notre outil de découverte de liens pour collecter les instances similaires à partir d'un ou deux ensembles de données liées du LOD Cloud.
Par exemple, les triplets ci-dessous représentent différentes entités qui sont utilisées pour représenter le même objet du monde réel.

<http ://dbpedia.org/resource/Category :1108_in_Europe> *owl :sameAs*
<http ://wikidata.dbpedia.org/resource/Q8087295>
<http ://dbpedia.org/resource/Category :1108_in_Europe> *owl :sameAs*
<http ://wikidata.dbpedia.org/resource/Q8087302>

Collection des prédicats et des objets qui sont liés aux instances similaires

Dans Linked Data, les données sont stockées sous forme de triplets RDF <Sujet, Prédicat, Objet>. Nous considérons chaque instance récupérée par la première étape comme le sujet de notre triplet RDF. Dans cette étape, notre système collecte tous les prédicats et les objets de chaque instance donnée par la première étape en utilisant une requête SPARQL. Ensuite, il classe les prédicats et les objets collectés en cinq classes : une URI, une date, une classe ontologique, une valeur numérique et une chaîne de caractères. Nous notons que cette classification permet de simplifier la tâche de fusion.

Classe ontologique : Peut être identifiée à partir des prédicats *rdf :type* et *skos :inScheme*.

Numéro : La valeur est constituée d'un ensemble de nombres.

Date : La valeur est constituée d'une date.

URI : Est une chaîne de caractères qui commence par "http : //".

Chaîne de caractères : La valeur qui se termine par "@en".

Le tableau 5.1 présente la classification de chaque ressource liée aux instances similaires.

Pour l'exemple précédent, la liste des prédicats et des objets qui sont reliés aux instances similaires sont :

Instance 1 : http ://dbpedia.org/resource/Category :1108_in_Europe

Liste de prédicats :

http ://www.w3.org/1999/02/22-rdfsyntax-ns#type, http ://www.w3.org/2000/01/rdf-schema#label, http ://www.w3.org/ns/prov#wasDerivedFrom, http ://dbpedia.org/ontology/wikiPageID, http ://dbpedia.org/ontology/wikiPageRevisionID, http ://www.w3.org/2004/02/skos/core#broader, http ://www.w3.org/2004/02/skos/core#prefLabel

Liste d'objets :

http ://www.w3.org/2004/02/skos/core#Concept, "1108 in Europe"@en, http ://en.wikipedia.org/wiki/Category :1108_in_Europe ?oldid=407698458, 20424333, 407698458, http ://dbpedia.org/resource/Category :1100s_in_Europe, "1108 in Europe"@ en

Instance 2 : http ://wikidata.dbpedia.org/resource/Q8087295

Liste de prédicats :

http ://www.w3.org/2000/01/rdfschema# label, http ://www.w3.org/2002/07/owl#sameAs

Liste d'objets :

http ://wikidata.org/entity/Q8087295, "Category :1108 in Europe"@en

Instance 3 : http ://wikidata.dbpedia.org/resource/Q8087302

Liste de prédicats :

http ://www.w3.org/2000/01/rdfschema#label, http ://www.w3.org/2002/07/owl#sameAs

URI	Date	chaîne de caractères	Classe ontologique	Valeur numérique
http ://dbpedia.org/resource /Category :1108_in_Europe		1108 in Europe	http ://dbpedia.org/ontology /wikiPageID	407698458
http ://en.wikipedia.org/wiki /Category :1108_in_Europe ? oldid=407698458		Category :1108 in Europe	http ://dbpedia.org/ontology /wikiPageRevisionID	20424333
http ://wikidata.dbpedia.org/resource /Q8087295		Category :1109 in Europe	http ://www.w3.org/2004/02/skos /core#broader	
http ://wikidata.dbpedia.org/resource /Q8087302			http ://www.w3.org/2004/02/skos /core#prefLabel	
http ://wikidata.org/entity /Q8087302			http ://www.w3.org/2004/02/skos /core#Concept	

Table 5.1 – Classification des ressources

Liste d'objets :

http ://wikidata.org/entity/Q8087302, "Category :1109 in Europe"@en

Construction d'un ensemble d'arbres

Le résultat de la deuxième étape de notre système est un ensemble de triplets où chaque triplet contient un sujet et une collection de prédicats et d'objets ; nous pouvons modéliser chaque triplet par un arbre qui représente une solution de fusion des données liées. Donc, nous considérerons un arbre T comme suit :
$T = S, l_{PS}, l_{OS}$; où : $(l_{PS} = P1_S, P2_S, ..., Pn_S)$ est la liste de prédicats collectés dans la deuxième étape.
$(l_{OS} = O1_S, O2_S, ..., On_S)$ est la liste d'objets collectés dans la deuxième étape.
Pour notre exemple, l'ensemble d'arbres construits est :
Arbre 1 :
$S1$: http ://dbpedia.org/resource/Category :1108_in_Europe
lP_{S1} :
http ://www.w3.org/1999/02/22-rdf-syntax-ns#type, http ://www.w3.org/2000/01/rdf-schema#label, http ://www.w3.org/ns/prov#wasDerivedFrom, http ://dbpedia.org/ontology/wikiPageID, http ://dbpedia.org/ontology/wikiPageRevisionID, http ://www.w3.org/2004/02/skos/core#broader, http ://www.w3.org/2004/02/skos/core#prefLabel}
lO_{S1} :
http ://www.w3.org/2004/02/skos/core#Concept , "1108 in Europe"@en , http ://en.wikipedia.org

/wiki/Category :1108_in_Europe ? oldid=407698458 , 20424333, 407698458 , http ://dbpedia.org/ resource/Category :1100s_in_Europe , "1108 in Europe"@en}

Arbre 2 :

$S2$: http ://wikidata.dbpedia.org/resource/Q8087295

lP_{S2} :

http ://www.w3.org/2000/01/rdf-schema#label , http ://www.w3.org/2002/07/owl#sameAs

lO_{S2} :

http ://wikidata.org/entity/Q8087295, "Category :1108 in Europe"@en

Arbre 3 :

$S3$: http ://wikidata.dbpedia.org/resource/Q8087302

lP_{S3} :

http ://www.w3.org/2000/01/rdf-schema#label, http ://www.w3.org/2002/07/owl#sameAs

lO_{S3} :

http ://wikidata.org/entity/Q8087302, "Category :1109 in Europe"@en

5.4.2 Algorithme génétique pour la fusion des arbres

Les arbres construits dans l'étape précédente sont fusionnés dans cette étape en utilisant un algorithme génétique. L'annexe A donne une brève introduction aux algorithmes génétiques. Notre algorithme reçoit en entrée un ensemble d'arbres qui représentent l'espace de recherche ou chaque arbre contient un sujet, une liste de prédicats et une liste d'objets. Il génère un seul arbre qui contient tous les éléments des arbres d'entrée, comme il ne contient pas des éléments en double.

Codage

Afin que notre algorithme génétique puisse rechercher une solution à notre problème, nous devons d'abord sélectionner la manière optimale pour encoder le problème avec un génome approprié. Lawrence Davis est un expert dans l'application des algorithmes génétiques pour des problèmes du monde réel, il a conseillé d'utiliser l'encodage qui est le plus naturel pour le problème (Coley, 1999). Dans notre approche, nous utilisons un codage sous forme d'arbre syntaxique. La figure 5.3 donne une meilleure idée de la façon dont un arbre est encodé.

Soit $l_{Arbre} = T_1, T_2, ..., T_m$ est l'ensemble d'arbres à fusionner. La population est composée d'une liste d'individus qui représentent un ensemble de solutions potentielles à un moment donné. Chaque individu représente un arbre qui est défini par un sujet, un chromosome de prédicats et un chromosome d'objets. Chaque chromosome est composé de m gènes où chaque gène, à son tour, est composé d'un prédicat ou

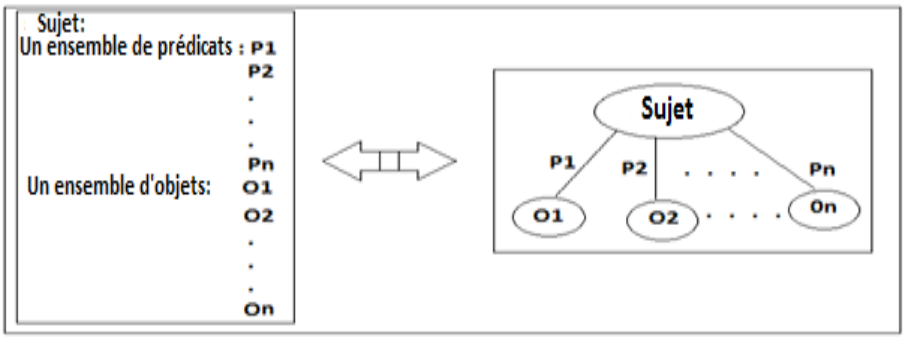

Figure 5.3 – Le codage sous forme d'arbre

d'un objet. Dans un premier temps, la population initiale est créée en combinant un ensemble d'individus (arbres) où chacun d'eux est basé sur différents sujets, prédicats et objets.
Nous considérons les prédicats et les objets comme des chromosomes différents parce qu'ils facilitent la classification des gènes (les gènes sont classés en cinq classes : classe ontologique, URI, valeur numérique, chaîne de caractère ou date). En outre, tous les prédicats sont des URIs ou des classes ontologiques, alors qu'un objet peut être un littéral, une URI ou une chaîne de caractères. Par conséquent, tous les chromosomes de prédicats sont classés en deux classes, tandis que les chromosomes d'objets sont classés en cinq classes. La comparaison de deux gènes du même type de données réduit la complexité et améliore l'efficacité de notre système.

Description de notre algorithme génétique

L'algorithme 3 représente notre algorithme génétique qui a été utilisé pour la fusion des données liées.

Les deux premières étapes de l'algorithme permettent : (1) d'initialiser la population en choisissant N arbres aléatoires de l'espace de recherche et (2) d'évaluer la population. La partie itérative de l'algorithme commence à la troisième étape. Cette partie sera répétée tant qu'il y aura des éléments similaires jusqu'à ce que l'espace de recherche soit exploré complètement. Les phases principales de la partie itérative sont la sélection des individus, le croisement, la mutation pour produire de nouveaux individus, l'évaluation et le remplacement de la population. Nous décrivons toutes ces étapes en détail dans les sections suivantes.
Fonction d'évaluation : la fonction d'évaluation permet de combiner des individus qui satisfont un ensemble de contraintes. En outre, la fonction d'évaluation est utilisée pour évaluer la qualité des individus. Elle doit punir les individus qui ne répondent pas aux contraintes et conduisent l'évolution vers l'accomplissement des

Algorithm 3 La fusion des arbres

Entrées : Un ensemble d'arbres
Sorties : Un seul arbre
1. Créer la population initiale ;
2. évaluer la population ;
3. Tant que (il existe des prédicats ou des objets similaires ou un prédicat 'owl : sameAs') et (l'espace de recherche n'est pas exploré) faire
    4. Sélectionner des individus ;
    5. Croiser les deux individus sélectionnés ;
    6. Éffectuer la mutation ;
    7. Évaluer les nouveaux individus ;
    8. Remplacer la population ;
fin
fin

contraintes. Elle détermine aussi si un individu est approprié pour le croisement et la mutation ou non. Ainsi, nous concevons une fonction d'évaluation qui compare deux gènes de différents arbres afin de déterminer quels chromosomes de la population sont les plus susceptibles pour se reproduire à l'aide des opérateurs génétiques : le croisement et la mutation.

Afin de répondre à ce besoin, nous considérons deux types importants d'algorithmes de matching :

a) La comparaison de type de données : Afin d'améliorer l'efficacité de notre système, nous comparons des entités qui ont le même type de données. Pour cela, et comme mentionné précédemment, nous classons toutes les entités en cinq classes : classe ontologique, URI, valeur numérique, chaîne de caractères et date. Nous notons que cette classification peut efficacement nous aider à choisir les meilleurs individus pour les opérateurs de croisement et de mutation.

b) La similarité à base de chaînes de caractères : L'évaluation d'une solution candidate est basée sur le calcul de similarité entre deux gènes. Deux mesures de similarité sont utilisées (la mesure de Levenshtein et Jaro).

$$sim(x,y) = \begin{cases} 1 - \frac{d(x,y)}{max(length(x), length(y))} \text{ .....mesure de Levenshtein} \\ \\ \frac{1}{3[\frac{c}{|x|} + \frac{c}{|y|} + \frac{(c-t)/2}{c}]} \text{ ..... measure de Jaro} \end{cases} \tag{5.1}$$

où :

Dans la mesure de Levenshtein, $d(x, y)$ représente le coût minimal de la transformation de la chaîne x en y (supprimer un caractère, insérer un caractère et remplacer

un caractère par un autre) (AnHai et al., 2012).
Dans la mesure de Jaro, c représente le nombre de caractères communs entre x et y, et t est le nombre de transpositions (une transposition est faite quand le caractère commun $i^{ème}$ de x et le caractère commun $i^{ème}$ de y sont différents) (AnHai et al., 2012). Par conséquent, notre fonction d'évaluation est définie comme suit :

$$sim(x, y) = \frac{Levenshtein\, mesure + Jaro\, mesure}{2}$$

La découverte des prédicats ou des objets identiques classés en Date et Nombre se faite par un matching exacte (deux prédicats ou objets de type Date ou Nombre sont similaires si la similarité entre eux est égale à 1). Les prédicats et les objets de type chaîne de caractères, URI et classe ontologique sont fusionnés si leurs scores de similarité est supérieure à un seuil de similarité prédéfini.
Sélection : Cette opération choisit les deux meilleurs individus pour le croisement. Pour cela, nous utilisons la sélection de tournoi binaire, elle peut être faite en appliquant une fonction d'évaluation qui distingue un chromosome d'un autre. Dans une sélection de k-tournoi, nous sélectionnons aléatoirement k individus de la population avec le remplacement (nous choisissons k = 2 en raison d'utilisation de la sélection de tournoi binaire), et nous préservons le meilleur d'entre eux.
Croisement : Il permet de fusionner les chromosomes qui contiennent des prédicats ou des objets similaires. Cela produira une nouvelle population d'individus (Offspring) qui détiennent tous les éléments des deux parents. L'opérateur de croisement remplace un gène d'un individu par un gène d'un autre individu. Donc, le processus de croisement consiste à sélectionner aléatoirement un gène du premier individu et calculer la fonction d'évaluation entre ce gène et tous les gènes du second individu. Si la fonction d'évaluation est supérieure à un seuil donné, l'algorithme remplace le gène du premier individu par le gène sélectionné du second individu. Sinon, les deux gènes sont préservés.
Mutation : L'opérateur de mutation permet de déterminer quel gène doit disparaître de la population dans chaque génération et être remplacé par le nouveau gène. Le critère de sélection est la valeur de la fonction d'évaluation qui calcule les scores de similarité entre les prédicats et les objets. L'algorithme choisit le gène qui doit disparaître en utilisant la fonction d'évaluation.
En appliquant les opérateurs génétiques sur les arbres construits dans l'étape précédente, l'algorithme passe par trois étapes :
Etape N :
Population = T_1, T_2, T_3.
Dans cette étape, l'algorithme calcule la fonction d'évaluation (la similarité) entre deux individus (arbres), sélectionne un couple pour le croisement (le deuxième et

le troisième arbre dans notre exemple), crée l'arbre résultant (T_{23}) et remplace le couple (T_2, T_3) par le nouvel individu T_{23}. Nous notons que la mutation insère le nouvel offspring à la place du couple (T_2, T_3). Donc, le nouvel offspring est T_{23} :
Sujet : http ://wikidata.dbpedia.org/resource/Q8087295, http ://wikidata.dbpedia.org/resource/Q8087302
L'ensemble de prédicats :
http ://www.w3.org/2000/01/rdfschema#label, http ://www.w3.org/2002/07/owl#sameAs
L'ensemble d'objets :
"Category :1109 in Europe"@en, "Category :1108 in Europe"@en
Etape N + 1 :
Population = T_1, T_{23}.
Comme le nouvel offspring T_{23} contient le prédicat <http ://www.w3.org/2002/07/owl#sameAs> (voir l'ensemble des prédicats du nouvel offspring T_{23}), l'algorithme doit fusionner les instances liées via ce prédicat. Après la fusion, le résultat sera le nouvel individu T'_{23} :
Sujet : http ://wikidata.dbpedia.org/resource/Q8087295
L'ensemble des prédicats :
http ://www.w3.org/2000/01/rdfschema#label
L'ensemble des objets :
"Catégorie : 1109 en Europe" @en, "Catégorie : 1108 en Europe" @en
Etape N + 2 :
Population = T_1, T'_{23}.
Dans cette étape, l'algorithme calcule la fonction d'évaluation entre deux individus, sélectionne un couple pour le croisement (le premier et le second arbre), crée l'arbre résultant (T_{123}) et remplace le couple (T_1, T'_{23}) par le nouvel individu T_{123}. Donc, le nouvel offspring est T_{123} :
Sujet : http ://dbpedia.org/resource/Category :1108_in_Europe
L'ensemble de prédicats :
http ://www.w3.org/2000/01/rdfschema#label, http ://www.w3.org/2002/07/owl#sameAs, http ://www.w3.org/1999/02/22-rdf-syntax-ns#type, http ://www.w3.org/ns/prov#wasDerivedFrom, http ://dbpedia.org/ ontology/wikiPageID, http ://dbpedia.org/ontology/wikiPageRevisionID, http ://www.w3.org/2004/02/skos/core#broader, http ://www.w3.org/2004/02/skos/core#prefLabel
L'ensemble d'objets :
"Category :1109 in Europe"@en, "Category :1108 in Europe"@en, http ://www.w3.org/2004/02/skos/core#Concept, 407698458, http ://en.wikipedia.org/wiki /Category :1108_in_Europe ?oldid=40769845, 20424333, http ://dbpedia.org/resource/Category :1100s_in_Europe

5.5 Evaluation

Dans cette section, nous évaluons expérimentalement l'efficacité et la performance de notre système de fusion des données liées. Pour cela, nous utilisons les ensembles de données liées suivants :

DBpedia [1] **:** Il s'agit d'une source de crowdsourcing pour 111 langues qui extrait des données structurées à partir de Wikipedia dans ses langues correspondantes et les rend disponibles dans le LOD Cloud. La version anglaise de DBpedia décrit 4,8 millions choses, 1 445 000 personnes, 73 000 places, 241 000 organisations, etc. En outre, cet ensemble de données comprend 38 millions étiquettes et 1519 millions liens vers des ensembles de données externes [2] .

BBC Programms [3] **:** C'est un ensemble de données qui publie des données sur les programmes de télévision et de radio diffusés sur le canal de la BBC. Tout d'abord, il extrait des concepts sur les lieux, les peuples, les thèmes et les organisations à partir d'un article. Ensuite, il crée un ensemble de triplets pour relier les concepts et l'article. Enfin, il les publie en utilisant les principes de données liées [4].

Drugbank [5] **:** Il s'agit d'une base de connaissances bioinformatiques et chimiques qui combine des données pharmaceutiques, chimiques et pharmacologiques avec des informations compréhensibles, c'est-à-dire la séquence, le chemin et la structure. Il contient 8206 entrées de drogues [6], y compris 207 médicaments approuvés par le FDA (protéines / peptides), 1991 drogues de petits molécules approuvés par le FDA, plus de 6000 médicaments expérimentaux et 93 nutraceutiques.

5.5.1 Evaluation de l'efficacité

Selon (Jens and Felix, 2009), la qualité des données peut être évaluée à l'aide de plusieurs dimensions : la complétude (completeness), la concision (conciseness), la consistance (consistency), la précision (accuracy), la pertinence (relevancy), la vérifiabilité (verifiability), la compréhensibilité (understandability), l'objectivité (objectivity), la crédibilité (believability), la disponibilité (availability) et l'actualité (timeliness).

Afin d'expliquer notre contribution et d'évaluer l'efficacité de notre système, nous nous concentrerons sur trois arbres hétérogènes de DBpedia. Ainsi, nous calculons la complétude et la concision avant et après la tâche de fusion des ensembles de

1. http://dbpedia.org/Downloads37
2. http://dbpedia.org/about
3. http://www.bbc.co.uk/programmes
4. https: // www. W3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/DataSets
5. http://www4.wiwiss.fu-berlin.de/drugbank/ sparql
6. http://www.drugbank.ca

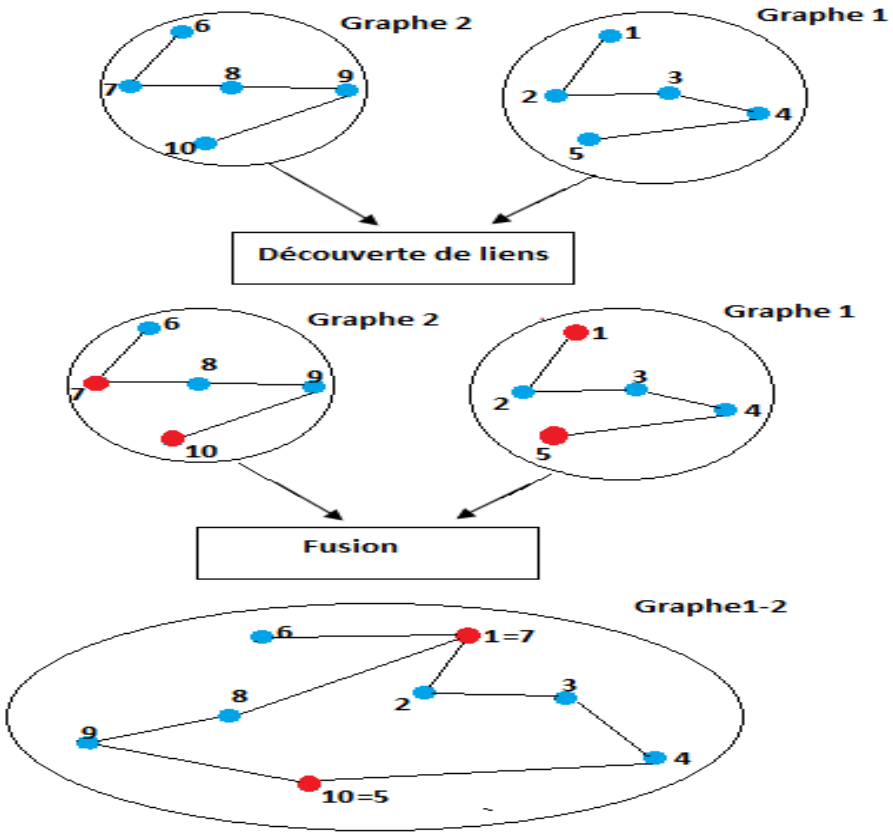

Figure 5.4 – Un exemple de fusion de deux graphes RDF

données liées. Dans nos expérimentations, nous avons extrait des instances liées via le prédicat *owl :sameAs* à partir de l'ensemble de données DBpedia et nous avons construit un ensemble d'arbres en utilisant les prédicats et les objets liés aux instances similaires, puis nous avons fusionné les arbres construits en utilisant l'algorithme génétique. Comme le système d'intégration des données liées vise à faire accroitre la complétude et la concision des données qui sont disponibles dans le LOD Cloud, nous proposons une définition de la complétude et la concision dans la processus de fusion des données liées.

En analogie avec l'évaluation d'un système en utilisant des mesures de précision et de rappel dans le domaine de la recherche d'informations, les mesures de complétude et de concision sont déterminées par le nombre d'objets supplémentaires et uniques dans un ensemble de données et dans tout l'univers. La figure 5.4 présente un exemple de fusion de deux graphes RDF.

a) La complétude : La mesure de la complétude est l'équivalent du rappel dans le domaine de la recherche d'informations. Elle est calculée en déterminant le nombre d'objets et de propriétés dans un ensemble de données liées. la complétude peut accroitre en ajoutant plus d'objets et plus de propriétés à un ensemble de don-

nées. En d'autres termes, la complétude permet de calculer le pourcentage d'objets entourés d'un ensemble de données liées.
L'exemple de la figure 5.4 présente une combinaison de deux graphes RDF. Les deux graphes contiennent en totale dix ressources (il existe cinq ressources dans chaque graphe dont six valeurs sont distinctes et quatre valeurs sont identiques). La fusion des deux graphes, telle qu'elle est indiquée dans la figure 5.4, a une complétude de 1 (10/10). Elle est donc maximale dans l'exemple, tandis que les deux graphes 1 et 2 ne sont pas exhaustives (5/8).
Dans ce qui suit, nous proposons une définition de la complétude des données liées à deux niveaux :
Définition 1 : Un ensemble de données liées est considéré complet au niveau de schémas s'il comprend toutes les classes et les propriétés obligatoires pour une tâche donnée. Elle est représentée par le nombre de classes et de propriétés uniques dans un ensemble de données par rapport au nombre total de classes et de propriétés uniques dans la réalité. Une crroissance est obtenue en combinant des sources qui fournissent des classes et des propriétés supplémentaires. En fait, les classes et les propriétés supplémentaires ne peuvent pas être incluses dans la liste de mappings de schémas.
Définition 2 : Un ensemble de données liées est considéré complet au niveau d'instances s'il inclut toutes les instances requises pour une tâche donnée. Elle est représentée par le nombre de représentations d'instances uniques dans un ensemble de données par rapport au nombre total d'instances uniques dans le monde réel. Par conséquent, la complétude au niveau de schémas mesure le pourcentage d'instances couvertes par un ensemble de données liées. Évidemment, nous ne pouvons pas calculer l'intégralité d'un ensemble de données que si nous connaissons l'ensemble des classes, des propriétés et d'instances.
b) La concision : À son tour, la mesure de concision est l'équivalent de la précision dans le domaine de la recherche d'informations. Elle est calculée en déterminant le nombre d'objets et de propriétés uniques dans un ensemble de données liées. La concision peut accroitre en supprimant des données inutiles, et en fusionnant les objets équivalents et les propriétés communes. Par conséquent, nous pouvons déterminer la concision en calculant le nombre d'objets et de propriétés uniques dans un ensemble de données liées.
Pour l'exemple de la figure 5.4, la concision du graphe résultant est 1 (8/8) ; car il contient huit ressources différentes et huit représentations de ressources différentes dans le résultat.
Dans ce qui suit, nous proposons une définition de la concision des données liées à

deux niveaux :

Définition 3 : Un ensemble de données liées est considéré concis au niveau de schémas s'il ne comprend pas de classes ou de propriétés surnuméraires (c'est-à-dire qu'une même classe ou propriété est représentée par des noms différents). Donc, elle mesure le nombre de classes ou de propriétés uniques dans un ensemble de données par rapport au nombre total de classes ou de propriétés.

Définition 4 : Un ensemble de données liées est considéré concis au niveau d'instances s'il n'inclut pas d'instances surnuméraires (c'est-à-dire, deux instances similaires sont représentées par des noms différents). Donc, elle mesure le nombre d'instances uniques dans un ensemble de données par rapport au nombre total de représentations d'instances dans l'ensemble de données.

Par conséquent, la concision peut accroitre dans les ensembles de données liées en supprimant les données redondantes, et en fusionnant les données en double et les données communes en une seule représentation. La tableau 5.2 illustre le nombre de prédicats et le nombre d'objets dans des graphes RDF ainsi que les valeurs de la complétude et de la concision avant le processus de la fusion.

	Avant la fusion					
	Nombre de prédicats	Nombre d'objets	complétude		Concision	
			Prédicats	objets	Prédicats	objets
Graphe1	7	7	1	0.85	1	0.85
Graphe1	2	2	1	0.50	1	0.50
Graphe1	2	2	1	0.50	1	0.50

Table 5.2 – Complétude et la concision avant la fusion

Le tableau 5.3 présente les valeurs de la complétude et de la concision après le processus de la fusion ; il démontre que notre système accroit la complétude et la concision dans les graphes RDF.

La complétude des objets mesure la proportion du nombre d'objets uniques dans un ensemble de données liées par rapport au nombre total d'objets uniques dans le monde réel (c'est-à-dire, dans toutes les sources d'un système intégré)(voir l'équation (1) (Jens and Felix, 2009)). De même, la complétude des prédicats mesure la proportion du nombre de prédicats uniques dans un ensemble de données liées par rapport au nombre total de prédicats uniques dans le monde réel (c'est-à-dire, dans toutes les sources d'un système intégré).

$$\text{Complétude des objets} = \frac{Le\ nombre\ des\ objets\ uniques\ dans\ l'ensemble\ de\ données}{Le\ nombre\ de\ tous\ les\ objets\ uniques\ dans\ l'univers} \quad ...(1)$$

	Aprés la fusion					
	Nombre de prédicats	Nombre d'objets	complétude		Concision	
			Prédicats	objets	Prédicats	objets
Graphe1	7	7	1	0.83	1	1
Graphe2	1	1	1	1	1	1
Graphe3	1	1	1	1	1	1

TABLE 5.3 – Complétude et la concision aprés la fusion

La figure 5.5 présente le pourcentage de la complétude et de la concision de deux populations avant et après le processus de fusion. Donc, la complétude de la première population était de 63% et la complétude de la seconde population était de 72% avant le processus de fusion. Après le processus de fusion, la complétude de la première population devient 100% et la complétude de la seconde population devient 85%. Par conséquent, nous remarquons que notre système peut accroitre efficacement la complétude des populations.
Par ailleurs, la concision des objets est mesurée par le nombre d'objets uniques dans un ensemble de données liées par rapport au nombre total de représentations d'objets uniques dans l'ensemble de données (voir l'équation (2) (Jens and Felix, 2009)). De la même manière, la concision des prédicats mesure le nombre de prédicats uniques d'un ensemble de données liées par rapport au nombre total de prédicats uniques.

$$\text{Concision des objets} = \frac{\textit{Le nombre des objets avec des valeurs uniques dans l'ensemble de données}}{\textit{Le nombre de tous les objets uniques dans l'ensemble de données}} \quad \ldots (2)$$

Comme le montre la figure 5.5, la concision de la première population est de 54% et la concision de la seconde population est de 45% avant le processus de fusion.

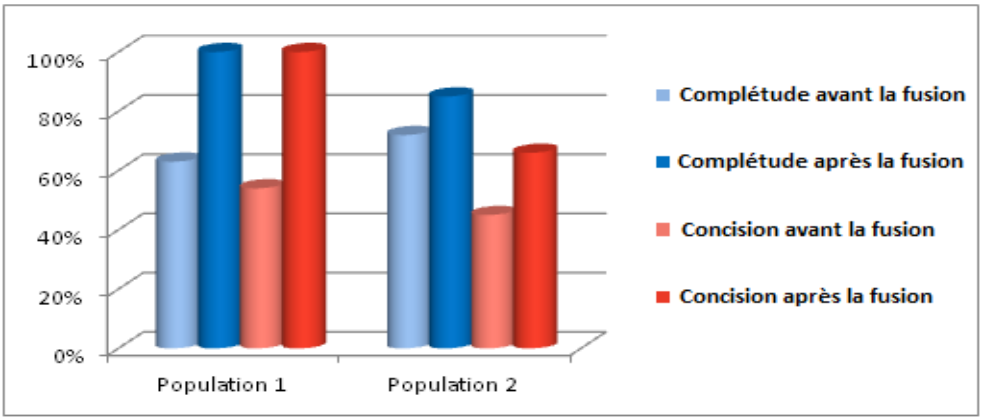

FIGURE 5.5 – Le pourcentage de la complétude et de la concision avant et après la fusion

Ensemble de données	Nombre d'instances extraites	Temps d'exécution de notre approche (millisecondes)	Temps d'exécution de Sieve (millisecondes)
Drugbank	5000	145	301
BBC Programmes	10000	190	332
DBpedia	29999	357	768
DBpedia	89996	675	3875

Table 5.4 – Le temps d'exécution de notre approche et de Sieve pour la fusion de grands ensembles de données liées

Après le processus de fusion, la concision de la première population devient 100% et la concision de la seconde population devient 66%. Par conséquent, nous remarquons que notre système augmente efficacement la concision des populations.

5.5.2 Evaluation de la performance

Dans cette section, nous évaluons la performance de notre système de fusion. Toutes les expérimentations sont faites sur un système 64 bits avec une machine Intel Core i5 2,5 GHz avec 8 Go de RAM. Le temps d'exécution total est calculé de la manière suivante :
1) Le temps nécessaire pour collecter les instances similaires ainsi que leurs prédicats et leurs objets à partir des ensembles de données liées ;
2) Le temps nécessaire pour appliquer les opérateurs génétiques sur les informations extraites ;
3) Le temps nécessaire pour afficher les résultats de la fusion.
Le tableau 5.4 illustre une comparaison entre notre approche et Sieve 0.5 [7] en termes du temps d'exécution.

En principe, le temps d'exécution n'est pas influencé par le processus de la fusion, mais en interrogeant les ensembles de données et en collectant des informations à partir d'eux. Pour cela, nous considérons uniquement dans notre évaluation le temps nécessaire pour effectuer le deuxième et le troisième composant du temps d'exécution total parce que le premier composant du temps d'exécution total dépend de la latence du réseau.
Bien que Sieve nécessite 332 millisecondes pour fusionner 10000 entités, notre approche nécessite seulement 190 millisecondes pour les fusionner. Ainsi, le temps d'exécution de notre approche représente 17,41% du temps d'exécution de Sieve pour fusionner 89996 entités. Notre système conduit à un temps d'exécution signi-

7. http://www4.wiwiss.fu-berlin.de/bizer/sieve/

ficatif en dépit du grand nombre d'éléments dans les ensembles de données liées du LOD Cloud. Les résultats de cette expérimentation sont encourageantes ; le temps d'exécution de notre système est plus performant à celui de Sieve.

5.6 Conclusion

Comme le nombre d'ensembles de données du LOD Cloud augmente considérablement, des centaines de milliers de ressources ont des valeurs conflictuelles. Ainsi, le processus de fusion des données vise à fournir une vue unifiée à l'utilisateur. Un algorithme génétique pour la fusion des données liées est présenté dans ce chapitre. Notre système met en oeuvre une approche performante pour la fusion des informations conflictuelles provenant de grands ensembles de données du LOD Cloud. Nous avons évalué notre approche en utilisant des ensembles de données réelles en montrant qu'elle fait accroitre à la fois la complétude et la concision des ensembles de donnés liées. De plus, notre évaluation montre que notre approche est 5 fois plus rapide que Sieve.

Chapitre 6

Conclusion Générale et perspectives

L'objectif des travaux qui sont menés dans le cadre de cette thèse est de proposer des approches dans le cadre d'intégration des données liées. Ces approches permettent d'achever le dernier principe des données liées, à savoir "Inclure des liens vers d'autres URIs, afin que les gens puissent découvrir plus de choses", et de fournir une vue unifiée des ensembles de données liées afin que les applications de données liées puissent interroger ces ensembles de données d'une manière efficace.
En fait, un système d'intégration des données liées vise à découvrir des liens au niveau de schémas (c'est-à-dire, des liens entre les classes et les propriétés des ontologies) et au niveau d'instances. Ensuite, il évalue la qualité des données et merge les classes et les propriétés équivalentes ainsi que les instances similaires afin d'obtenir un ensemble de données complet, concis et consistant.
Le résultat des travaux menés est la proposition d'une méthode de découverte des liens entre les ensembles de données liées du LOD Cloud et un outil de visualisation du résultat de la découverte des liens et une approche de fusion des données liées. Nous avons utilisé différentes mesures de similarités (à savoir les mesures syntaxiques et les mesures sémantiques). En outre, nos approches exploitent la structure interne des ensembles de données liées qui vont être intégrés.
Afin de trouver les différentes entités qui existent dans plusieurs ensembles de données liées du LOD Cloud et décrivent le même objet du monde réel, nous avons développé une approche de découverte des liens dans Linked Data. Notre approche de découverte des liens vise à minimiser le temps d'exécution en améliorant la complexité temporelle. En outre, elle vise à utiliser le Wordnet et des mesures de similarités à base de chaînes de caractères pour améliorer l'efficacité de notre système. Cette approche a été évaluée en utilisant une grande quantité de ressources à partir des ensembles de données liées réels. En particulier, nous avons évalué la performance et l'efficacité de notre approche en utilisant différentes mesures de la qualité, à savoir

la précision et le rappel. De plus, nous avons comparé notre approche de découverte des liens avec d'autres approches appartenant à la littérature dédiée. Les résultats obtenus sont satisfaisants et encourageants parce que notre approche surperforme des approches de l'état de l'art dans les expérimentations qui sont faites pour mapper les grands ensembles de données liées. Ainsi, notre système renvoie beaucoup plus de résultats dans le processus de mapping en raison de l'utilisation du Wordnet et de plusieurs mesures de similarités syntaxiques pour calculer la similarité.
Le résultat du processus de découverte des liens doit être visualisé pour que l'utilisateur final puisse comprendre la structure principale des ensembles de données et interpréter les données d'une manière significative. Pour cela, nous avons développé un outil qui permet de visualiser le résultat du processus de découverte des liens pour aider les utilisateurs et l'expert humain à détecter des erreurs dans le processus d'appariement d'une part et de vérifier la complétude et la précision des liens, d'autre part.
Afin de fournir une vue unique de plusieurs ensembles de données liées et de les interroger d'une manière efficace, nous avons développé une nouvelle approche de fusion des données liées. Comme les algorithmes génétiques sont souvent utilisés pour optimiser la solution d'un problème donné, nous avons proposé un algorithme génétique pour fusionner les données liées. Notre approche de fusion vise à combiner les ressources dupliquées qui existent dans différents ensembles de données liées du LOD Cloud. En outre, elle permet de minimiser le temps d'exécution lors de la fusion de grands ensembles de données liées. Cette approche a été évaluée en utilisant différents mesures de qualité, à savoir la complétude (l'équivalent du rappel dans les systèmes de recherche d'informations) et la concision (l'analogue de la précision dans les systèmes de recherches d'informations). Nous avons comparé la complétude et la concision des ensembles de données liées avant et après le processus de la fusion pour montrer que notre approche fait accroitre les valeurs de la complétude et de la concision au niveau de schémas et au niveau d'instances ainsi qu'elle améliore les résultats de la fusion. De plus, nous avons évalué la performance de notre système de fusion des données liées en comparant son temps d'exécution avec celui de Sieve 5.0 et nous avons montré qu'il a conduit à un temps d'exécution significatif en dépit du grand nombre d'éléments dans les ensembles de données liées du LOD Cloud.

Perspectives :

Les perspectives de cette thèse portent sur deux volets. Le premier volet inclut des travaux futurs visant l'amélioration de notre méthode de découverte des liens.

Le second volet inclut des travaux futurs permettant d'améliorer notre approche de fusion des données liées.

a) Les travaux futurs pour l'amélioration de notre méthode de découverte des liens comportent les tâches suivantes :

1) Calculer automatiquement le nombre d'exemplaires. La définition des valeurs optimales pour le nombre d'exemplaires permet de minimiser le nombre de comparaisons dans le processus de matching et donc conduit à des temps d'exécution encore plus petits. Dans nos expérimentations, nous avons utilisé $\sqrt{T}$ comme un nombre d'exemplaires. Donc, nous avons besoin d'un moyen pour calculer automatiquement le nombre d'exemplaires.

2) Sélectionner une spécification de liens appropriée. Généralement, la configuration des frameworks de découverte des liens est effectuée manuellement. En fait, le choix d'une mesure de spécification de liens appropriée est essentiel pour améliorer l'efficacité d'un système de découverte des liens dans Linked Data. Donc, soutenir l'utilisateur pour trouver une mesure de similarité et un seuil appropriés au cours du processus de découverte des liens est un problème qui doit encore être traité dans nos travaux futurs.

3) Proposer des approches de découverte des liens au niveau de schémas. Dans nos expérimentations, nous avons appliqué notre approche de découverte des liens pour trouver des liens au niveau d'instances et nous avons ignoré la découverte des liens au niveau de schémas (à savoir les classes et les propriétés liés aux instances). Donc, nous devrons intégrer d'autres méthodes de calcule de similarité dans notre système de découverte des liens afin qu'il puisse découvrir des liens au niveau de schémas.

b) Les travaux futurs pour l'amélioration de notre approche de fusion des données liées comportent les tâches suivantes :

1) Améliorer la fonction d'évaluation de l'algorithme génétique. La fonction d'évaluation de notre algorithme génétique combine deux mesures de similarités à base de chaînes de caractères. En fait, le développement d'une méthode automatique et efficace pour le calcule de similarité permet d'augmenter la concision de notre système de fusion des données liées.

3) Évaluer notre approche de fusion des données liées en utilisant d'autres dimensions de qualité, à savoir la précision et la consistance. Dans notre évaluation, nous avons pris en considération deux dimensions de la qualité : la complétude et la concision, alors qu'on peut évaluer la qualité en utilisant d'autres dimensions. Donc, l'évaluation de la qualité des données à l'aide d'autres dimensions sera l'objectif de nos travaux futurs.

4) Enrichir les ensembles de données liées fusionnés par de nouvelles relations. Après le processus de fusion des données liées, nous devrons développer des méthodes d'enrichissement des données liées pour ajouter de nouvelles relations au résultat de la fusion (par exemple, des relations d'équivalences, des relations de subsomptions, des relations de disjonctions... etc.). En fait, l'enrichissement des ensembles de données liées après la fusion permet de faciliter l'interrogation et de trouver des inconsistances dans ces ensembles de données.

Bibliographie

Xbrl - extensible reporting business report language, June 19, 2016. http://www.xbrl.org/.

Edgar system fillings, June 19, 2016. http://www.sec.gov/cgi-bin/browse-edgar?action=getcurrent.

Dadzie Aba-Sah and Rowe Matthew. Approaches to visualising linked data : A survey. *Semantic web*, 2(2) :89–124, April 2011.

Serge Abiteboul, Ioana Manolescu, Philippe Rigaux, Marie-Christine Rousset, and Pierre Senellart. *Web Data Management*. Cambridge University Press, New York, NY, USA, 2011.

Ben Adida and Mark Birbeck. Rdfa primer - bridging the human and data webs - w3crecommendation, 2008. http://www.w3.org/TR/xhtml-rdfa-primer/.

Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing linked datasets. In *Proceedings of the 2nd Workshop on Linked Data on the Web (LDOW2009)*, 2009.

Nikolov Andriy, Mathieu d'Aquin, and Motta Enrico. Unsupervised learning of link discovery configuration. In *Proceedings of the 9th International Conference on The Semantic Web : Research and Applications*, ESWC'12, pages 119–133, Berlin, Heidelberg, 2012. Springer-Verlag.

Doan AnHai, Halevy Alon, and Ives Zachary. *Principles of Data Integration*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012.

Sören Auer, Jens Lehmann, and Axel-Cyrille Ngomo Ngonga. Introduction to linked data and its lifecycle on the web. In *Proceedings of the 7th International Confe-*

rence on Reasoning Web : Semantic Technologies for the Web of Data, RW'11, pages 1–75, Berlin, Heidelberg, 2011. Springer-Verlag.

David Beckett. Rdf/xml syntax specification (revised) - w3c recommendation, 2004. http://www.w3.org/TR/rdf-syntax-grammar/.

David Beckett and Tim Berners-Lee. Turtle - terse rdf triple language, 2008. http://www.w3.org/TeamSubmission/turtle/.

François Belleau, Nolin Marc-Alexandre, Tourigny Nicole, Rigault Philippe, and Morissette Jean. Journal of biomedical informatics. *Bio2rdf : Towards a mashup to build bioinformatics knowledge systems*, 41(5) :706–16, 2008.

David Ben-David, Tamar Domany, and Abigail Tarem. Enterprise data classification using semantic web technologies. In *Proceedings of the 2010 International Conference on Posters ; Demonstrations Track - Volume 658*, ISWC-PD'10, pages 189–192. CEUR-WS.org, 2010.

Khayra Bencherif and Mimoun Malki. Int. j. metadata, semantics and ontologies. *An effective and time-efficient approach for Linked Data fusion using genetic algorithms*, 11(2) :110–123, 2016.

Khayra Bencherif and Mimoun Malki. Linked data enrichment : State-of-art. In *Proceedings of the 1st International Conference on Business Intelligence and Applications*, March 2016. https://icbia.com.

Khayra Bencherif and Mimoun Malki. Linked data visualization tool. In *Proceedings de la 5ème Edition du Workshop International sur l'Innovation et Nouvelles Tendances dans les Systèmes d'Information, INTIS 2016*, pages 105–113, Novembre 2016.

Khayra Bencherif, Mimoun Malki, and Soumia Berrahal. Indexing-based link discovery in linked data. In *Le proceeding de 10ième édition de la conférence sur Avancés des Systèmes Décisionnels - ASD 2016*, pages 101–112, Mai 2016.

Tim Berners-Lee. Linked data - design issues, 2006. http://www.w3.org/DesignIssues/LinkedData.html.

Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5) :34–43, May 2001.

Tim Berners-Lee, Roy Fielding, and Larry Masinter. Rfc 2396 - uniform resource identifiers (uri) : Generic syntax, August 1998. https://tools.ietf.org/html/rfc2396.

Diego Berrueta and Jon Phipps. Best practice recipes for publishing rdf vocabularies - w3c note, 2008. http://www.w3.org/TR/swbp-vocab-pub/.

Christian Bizer and Richard Cyganiak. Quality-driven information filtering using the wiqa policy framework. *Web Semant.*, 7 :1–10, 2009.

Christian Bizer and Andreas Schultz. The r2r framework : Publishing and discovering mappings on the web. In *1st International Workshop on Consuming Linked Data (COLD 2010), Shanghai*, 2010.

Christian Bizer, Richard Cyganiak, and Tobias Gauss. The rdf book mashup : From web apis to a web of data. In *Proceedings of the Workshop on Scripting for the Semantic Web*, 2007a.

Christian Bizer, Tom Heath, Danny Ayers, and Yves Raimond. Linking open data. In *Proceedings of the Poster Session at the 4th European Semantic Web Conference*, 2007b.

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. Journal of web semantics : Science, services and agents on the world wide web. *Dbpedia - a crystallization point for the web of data*, 7(2) :154–165, 2009.

Dan Brickley and Ramanathan V Guha. Vocabulary description language 1.0 : Rdf schema, w3c recommendation, 2004. http://www.w3.org/TR/rdf-schema/.

Volha Bryl, Christian Bizer, Robert Isele, Mateja Verlic, Soon Gill Hong, Sammy Jang, Mun Yong Yi, and Key-Sun Choi. Interlinking and knowledge fusion. In Auer Sören, Bryl Volha, and Tramp Sebastian, editors, *Linked Open Data*, volume 8661 of *Lecture Notes in Computer Science*, pages 70–89. Springer, 2014.

Davide Buscaldi and Paolo Rosso. Indexing with wordnet synonyms may improve retrieval results. In *CLEF (1)*, 2009.

Michelle Cheatham and Pascal Hitzler. String similarity metrics for ontology alignment. In *In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, JosianeXavier Parreira, Lora Aroyo, Natasha Noy, ChrisWelty, and Krzysztof Janowicz, editors, The Semantic Web - ISWC 2013, volume 8219 of Lecture Notes in Computer Science*, 2013.

Peter Christen. *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection.* Data-Centric Systems and Applications. Springer, 2012.

Xiao Chuan, Wang Wei, Lin Xuemin, Yu Jeffrey Xu, and Wang Guoren. Efficient similarity joins for near-duplicate detection. *ACM Trans. Database Syst.*, 36(3) : 15 :1–15 :41, August 2011.

David A. Coley. *An Introduction to Genetic Algorithms for Scientists and Engineers.* World Scientific, 1999.

Isabel F. Cruz, Afsheen Rajendran, William Sunna, and Nancy Wiegand. Handling semantic heterogeneities using declarative agreements. In *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, GIS '02, pages 168–174, New York, NY, USA, 2002. ACM.

Isabel F. Cruz, William Sunna, Nalin Makar, and Sujan Bathala. A visual tool for ontology alignment to enable geospatial interoperability. *J. Vis. Lang. Comput.*, 18(3) :230–254, June 2007.

Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. AgreementMaker : efficient matching for large real-world schemas and ontologies. *Proceedings of the VLDB Endowment*, 2(2) :1586–1589, August 2009.

Isabel F. Cruz, Matteo Palmonari, Federico Caimi, and Cosmin Stroe. Building linked ontologies with high precision using subclass mapping discovery. *Artif. Intell. Rev.*, 40(2) :127–145, 2013.

Mariana Damova, Atanas Kiryakov, Kiril Simov, and Svetoslav Petrov. Mapping the central lod ontologies to proton upper-level ontology. In *In : ISWC international workshop on ontology matching (OM), vol 689 of CEUR workshop proceedings*, pages 61–72, 2010.

Lin Dekang. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Large-scale linked data integration using probabilistic reasoning and crowdsourcing. *The VLDB Journal*, 22(5) :665–687, October 2013.

Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Kevin Murphy, Shaohua Sun, and Wei Zhang. From data fusion to knowledge fusion. *Proc. VLDB Endow.*, 7(10) :881–892, June 2014.

Kevin Dreßler and Axel-Cyrille Ngonga Ngomo. On the efficient execution of bounded jaro-winkler distances. In *Proceedings of Ontology Matching Workshop*, 2014.

Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection : A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1) :1–16, 2007.

Ukkonen Esko. Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.*, 92(1) :191–211, January 1992.

Jérôme Euzenat. An api for ontology alignment. In *Proceedings of the 3rd International Semantic Web Conference (ISWC'04)*, pages 698–712, 2004a.

Jérôme Euzenat. State of the art on ontology alignement. Technical report, INRIA, August 2004b.

Roy Fielding. Hypertext transfer protocol - http/1.1. request for comments : 2616, 1999. http://www.w3.org/Protocols/rfc2616/rfc2616.html.

Fabien Gandon. The semantic web : latest advances and new domains. In *12th European Semantic Web Conference (ESWC 2015)*, pages 1–75. Springer-Verlag, 2015.

Pierre Gewelt. Interrogation récursive du web sémantique. *Mémoire présenté en vue de l'obtention du diplôme de Ingénieur civil informaticien*, 2012.

Hugh Glaser, Ian C. Millard, Won-Kyung Sung, Seungwoo Lee, Pyung Kim, and Beom-Jong You. Research on linked data and co-reference resolution. Technical report, University of Southampton, 2009.

W3C OWL Working Group. Owl 2 web ontology language document overview, w3c recommendation, 2009. http://www.w3.org/TR/owl2-overview/.

Toni Grütze, Christoph Böhm, and Felix Naumann. Holistic and scalable ontology alignment for linked open data. In Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas, editors, *LDOW*, volume 937 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

Olaf Hartig and Andreas Langegger. A database perspective on consuming linked data on the web. *Datenbank-Spektrum, Semantic Web Special Issue*, 10(2) :57–66, 2010.

Michael Hartung, Anika Grob, and Erhard Rahm. Composition methods for link discovery. In *In BTW*, 2013.

Tom Heath and Christian Bizer. *Linked Data : Evolving the Web into a Global Data Space*. 2011.

Tom Heath, Christian Bizer, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3) :1–22, 2009.

Sebastian Hellmann, Sören Auer, and Jens Lehmann. Linkedgeodata - adding a spatial dimension to the web of data. In *Proceedings of the International Semantic Web Conference*, 2009.

Martin Hepp. Goodrelations : An ontology for describing products and services offers on the web. In *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management*, 2008.

Graeme Hirst and David St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. In Christiane Fellbaum, editor, *WordNet : An Electronic Lexical Database*, pages 305–332. MIT Press, 1998.

Jakob Huber, Timo Sztyler, Jan Noessner, and Christian Meilicke. Codi : Combinatorial optimization for data integration results for oaei 2011. In *In Proc. of the 6th International Workshop on Ontology Matching*, 2011.

Robert Isele, Anja Jentzsch, and Christian Bizer. Efficient multidimensional blocking for link discovery without losing recall. In Marian Amélie and Vassalos Vasilis, editors, *WebDB*, 2011.

Ian Jacobs and Norman Walsh. Architecture of the world wide web, 2004. http://www.w3.org/TR/webarch/.

Prateek Jain, Pascal Hitzler, Amit P. Sheth, Kunal Verma, and Peter Z. Yeh. Ontology alignment for linked open data. In *In : International semantic web conference (ISWC 2010), vol 6496 of Lecture Notes in Computer Science*, pages 402–417. Springer-Verlag, 2010.

Matthew Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 84 (406) :414–420, 1989.

Pound Jeffrey, Mika Peter, and Zaragoza Hugo. Ad-hoc object retrieval in the web of data. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 771–780, New York, NY, USA, 2010. ACM.

Bleiholder Jens and Naumann Felix. Data fusion. ACM Computing Surveys, 2009.

Anja Jentzsch, Oktie Hassanzadeh, Christian Bizer, Bo Andersson, and Susie Stephens. Enabling tailored therapeutics with linked data. In *Proceedings of the WWW 2009 Workshop on Linked Data on the Web*, 2009.

Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the Int'l. Conf. on Research in Computational Linguistics*, pages 19–33, 1997.

Ernesto Jimenez-Ruiz and Bernardo Cuenca Grau. Logmap : Logic-based and scalable ontology matching. In *In The Semantic Web-ISWC 2011*, 2011.

Amit Krishna Joshi, Prateek Jain, Pascal Hitzler, Pete Z. Yeh, Kunal Verna, Amit P. Sheth, and Mariana Damova. Alignment-based querying of linked open data. In *Proceedings of Ontologies, DataBases, and Applications of Semantics (ODBASE) 2012*, Rome, Italy, September 2012.

Joseph Juran. *The Quality Control Handbook*. McGraw-Hill, New York, 3rd edition, 1994.

Graham Klyne and Jeremy J. Carroll. Resource description framework (rdf) : Concepts and abstract syntax - w3c recommendation, 2004. http://www.w3.org/TR/rdf-concepts/.

Georgi Kobilarov, Tom Scott, Yves Raimond, Silver Oliver, Chris Sizemore, Michael Smethurst, Christian Bizer, and Robert Lee. Media meets semantic web - how the bbc uses dbpedia and linked data to make connections. In *In The SemanticWeb : Research and Applications, 6th European SemanticWeb Conference*, pages 723–737, 2009.

Claudia Leacock, George A. Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Comput. Linguist.*, 24(1) :147–165, March 1998.

VI Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10 :707, 1966.

Vanessa Lopez, Victoria Uren, Marta Reka Sabou, and Enrico Motta. Cross ontology query answering on the semantic web : An initial evaluation. In *Proceedings of the Fifth International Conference on Knowledge Capture*, K-CAP '09, pages 17–24, New York, NY, USA, 2009. ACM.

José Luis and Sanchez Cervantes. Discovering and linking financial data on the web. In *The Second International Conference on Data Analytics*, 2013.

Deborah L. McGuinness and Frank Van Harmelen. Owl web ontology language overview, w3c recommendation, 2004. http://www.w3.org/TR/2004/REC-owlfeatures-20040210/.

Noah Mendelsohn. The self-describing web - tag finding, 2009. http://www.w3.org/2001/tag/doc/selfDescribingDocuments.html.

Pablo N. Mendes, Hannes Mühleisen, and Christian Bizer. Sieve : Linked data quality assessment and fusion. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, EDBT-ICDT '12, pages 116–123, New York, NY, USA, 2012. ACM.

Jan Michelfeit and Jindrich Mynarz. New directions in linked data fusion. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, pages 397–400, 2014.

Alistair Miles and Sean Bechhofer. Skos simple knowledge organization system - reference, 2009. http://www.w3.org/TR/skos-reference/.

G. Miller. Wordnet : A lexical database for english language. *Communications of the ACM*, 38(11) :39–41, 1995.

Knud Möller, Tom Heath, Siegfried Handschuh, and John Domingue. Recipes for semantic web dog food - the eswc and iswc metadata projects. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference*, 2007.

Fabian M.Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago : a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web (WWW 2007)*, pages 697–706, 2007.

Felix Naumann. *Quality-Driven Query Answering for Integrated Information Systems*. Springer, Berlin Heidelberg New York, 2002.

Markus Nentwig, Tommaso Soru, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm. Linklion : A link repository for the web of data. In *In ESWC 2014 Posters & Demo session*, 2014.

Markus Nentwig, Michae Hartung, Axel-Cyrille Ngomo Ngonga, and Erhard Rahm. A survey of current link discovery frameworks. *Semantic Web*, (Preprint) :1–18, 2015.

Joachim Neubert. Bringing the "thesaurus for economics" on to the web of linked data. In *Proceedings of the WWW 2009 Workshop on Linked Data on the Web*, 2009.

Axel-Cyrille Ngonga Ngomo. A time-efficient hybrid approach to link discovery. In *Proceedings of OM@ISWC*, 2011.

Axel-Cyrille Ngonga Ngomo. Link discovery with guaranteed reduction ratio in affine spaces with minkowski measures. In *International Semantic Web Conference*, pages 378–393, 2012.

Axel-Cyrille Ngomo Ngonga and Sören Auer. Limes : A time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 2312–2317. AAAI Press, 2011.

Natalya Fridman Noy, Nigam H. Shah andPatricia L. Whetze, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L. Rubin, Margaret-Anne D. Storey, Christopher G.Chute, and Mark A. Musen. Bioportal : ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Research*, 37 (Web-Server-Issue) :170–173, 2009.

Siddharth Patwardhan. *Incorporating dictionary and corpus information into a context vector measure of semantic relatedness*. PhD thesis, 2003.

Nathalie Pernelle, Fatiha Saïs, Brigitte Safar, Maria Koutraki, and Tushar Ghosh. N2r-part : identity link discovery using partially aligned ontologies. In *Proceedings of the 2nd International Workshop on Open Data, WOD 2013, Paris, France, June 3, 2013*, pages 61–64, 2013.

Eric Prud'hommeaux and Andy Seaborne. Sparql query language for rdf, w3c recommendation, 2008. http://www.w3.org/TR/rdf-sparql-query/.

Dave Raggett, Arnaud Le Hors, and Ian Jacobs. Html 4.01 specification - w3c recommendation, 1999. http://www.w3.org/TR/html401/.

Yves Raimond, Christopher Sutton, and Mark Sandler. Automatic interlinking of music datasets on the semantic web. In *Proceedings of the 1st Workshop about Linked Data on the Web*, 2008.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

Banerjee Satanjeev and Pedersen Ted. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 805–810, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

François Scharffe, Yanbin Liu, and Chuguang Zhou. Rdf-ai : an architecture for rdf datasets matching, fusion and interlink. In *Proc. IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US)*, 2009.

Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the linked data best practices in different topical domains. In *In The Semantic Web-ISWC 2014*, pages 245–260. Springer, 2014.

Andreas Schultz, Andrea Matteini, Robert Isele, Christian Bizer, and Christian Becker. Ldif -linked data integration framework. In *Proceedings of the Second International Conference on Consuming Linked Data - Volume 782*, COLD'11, pages 125–130, Aachen, Germany, Germany, 2010. CEUR-WS.org.

John Sheridan and Jeni Tennison. Linking uk government data. In *Proceedings of the WWW 2010 Workshop on Linked Data on the Web*, 2010.

William Sunna and Isabel F. Cruz. Using the agreement maker to align ontologies for the oaei campaign 2007. In *Proceedings of the 2Nd International Conference on Ontology Matching - Volume 304*, OM'07, pages 133–138, Aachen, Germany, Germany, 2007. CEUR-WS.org.

Pedersen Ted, Patwardhan Siddharth, and Michelizzi Jason. Wordnet : :similarity : Measuring the relatedness of concepts. In *Demonstration Papers at HLT-*

NAACL 2004, HLT-NAACL-Demonstrations '04, pages 38–41, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

Alberto Tonon, Gianluca Demartini, and Philippe Cudré-Mauroux. Combining inverted indices and structured search for ad-hoc object retrieval. In *Proceedings of the 35th International ACM SIGIR Conference*, pages 125–134, New York, NY, USA, 2012.

Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank van Harmelen, and Henri Bal. Owl reasoning with webpie : calculating the closure of 100 billion triples. In *Proceedings of the ESWC 2010*, 2010.

Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov. Discovering and maintaining links on the web of data. In *Proceedings of the 8th International Semantic Web Conference*, ISWC '09, pages 650–665, Berlin, Heidelberg, 2009. Springer-Verlag.

William E. Winkler and Yves Thibaudeau. An application of the fellegi-sunter model of record linkage to the 1990 u.s. decennial census. Technical Report Statistical Research Report Series RR91/09, U.S. Bureau of the Census, Washington, D.C., 1991.

Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. *Similarity Search : The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 2006.

Sami Zghal. *Contributions à l'alignement d'ontologies OWL par agrégation de similarités*. PhD thesis, 2010.

Lihua Zhao and Ryutaro Ichise. *Ontology Integration for the Linked Open Data*. PhD thesis, 2013.

Annexe A

Algorithmes génétiques

Un algorithme génétique est un ensemble de modèles informatiques basés sur des aspects de la théorie de l'évolution de Darwin par la sélection naturelle. Il encode une solution potentielle à un problème spécifique sur un chromosome et applique des opérateurs de recombinaison (souvent appelés croisement) sur les chromosomes afin d'optimiser la solution du problème. L'objectif principal des algorithmes génétiques est de trouver une solution à des problèmes difficiles (c'est-à-dire les problèmes que nous n'ont pas une méthode exacte pour les résoudre dans un délai raisonnable). Nous pouvons passer d'une population de chromosomes à une nouvelle population en utilisant des opérateurs de sélection, de croisement et de mutation.

Encodage : Pour coder un chromosome, nous devons le représenter en utilisant une collection de gènes. Le choix de la bonne façon pour coder un chromosome est la tâche la plus importante puisqu'elle permet de représenter des données et des solutions et affecte la mise en oeuvre des opérations telles que le croisement et la mutation qui affectent directement le bon fonctionnement de l'algorithme génétique et sa convergence vers une solution correcte. Après l'encodage du problème et la génération d'une population initiale, les algorithmes génétiques lancent trois opérateurs : la sélection, le croisement et la mutation.

La fonction d'évaluation : L'évaluation d'un individu ne dépend pas de celle des autres individus ; le résultat fourni par la fonction d'évaluation permet de sélectionner ou de refuser un individu pour garder seulement les individus avec le meilleur score en fonction de la population actuelle. Cette méthode assure que les meilleurs individus seront retenus, tandis que les individus mal adaptés seront éliminés de la population.

La sélection : La fonction de sélection est basée sur la fonction d'évaluation précédemment définie ; elle identifie les meilleurs individus d'une population, sélectionne une sous-population de la population mère, élimine les individus les mois perfor-

mants et génère à leur place les meilleurs individus.

Le croisement : Permet d'échanger des informations entre les solutions de deux populations comme la reproduction sexuelle. Cela produira une nouvelle génération d'individus qui détiennent les caractéristiques des parents (comme le schéma d'héritage naturel). Après l'opérateur de sélection, nous choisissons un ou deux points dans la chaîne binaire et nous échangeons l'information entre les deux individus.

La mutation : Elle est utilisée pour changer aléatoirement la valeur d'un seul bit au sein d'un individu. D'une génération à l'autre, certains gènes du chromosome peuvent changer d'une façon aléatoire.

Annexe B

Mesures de similarité

Dans ce qui suit, nous allons présenter des méthodes de calcule de similarités qui visent à trouver des relations d'équivalences entre les entités de différentes ontologies.

Définition 1 : Une mesure de similarité, notée σ, permet de mesurer le degré de ressemblance entre deux entités (Euzenat, 2004b).

Dans la littérature, il existe plusieurs mesures de similarité, y compris : des mesures de similarité de chaînes de caractères et des mesures de similarité à base du Wordnet, qui visent à calculer le score de la similarité entre deux chaînes de caractères. Une mesure de similarité mappe une paire de chaînes (x, y) en un nombre réel dans l'intervalle $[0, 1]$ de telle sorte qu'une valeur plus élevée indique une plus grande similitude entre x et y (AnHai et al., 2012).

Les mesures de similarité de chaînes de caractères : Une mesure de similarité des chaînes de caractères mesure le rapport de la partie commune entre deux ontologies.

Définition 2 : La similarité de chaînes de caractères est une similarité $\sigma : S \times S \rightarrow [0, 1]$ telle que $\forall x, y \in S$ et soit t la plus longue chaîne commune entre x et y :

$$\sigma(x, y) = \frac{2|t|}{|x|+|y|}$$

Cette équation permet de calculer le plus long préfixe ou le plus long suffixe (Zhao and Ichise, 2013).

- La distance de Levenshtein (Levenshtein, 1966), appelée aussi Edit Distance, est conçue pour mesurer la similarité entre des chaînes de caractères qui peuvent contenir des fautes d'orthographe. Par conséquent, plus cette distance est plus petite, plus les deux chaînes sont similaires. La distance Levenshtein permet de calculer le coût minimal de transformation d'une chaîne x en une chaîne y. La transformation d'une chaîne de caractères est réalisée à l'aide des opérateurs suivants : supprimer

un caractère, insérer un caractère, et le remplacer par un autre.
La fonction de distance de Levenshtein $d(x, y)$ est convertie en une fonction de similarité $sim(x, y)$ comme suit :

$$Leven(x, y) = 1 - \frac{d(x,y)}{max(length(x),length(y))}$$

Comme le cout de transformation de la chaîne x à la chaîne y est le même que le coût minimal de transformation de y à x, on considère la distance $d(x, y)$ symétrique.
- La distance de Jaro (Jaro, 1989) permet de comparer des chaînes courtes, telles que les noms et les prénoms. Elle est basée sur le nombre et la proximité des caractères communs entre les deux chaînes. Soit deux chaînes x et y, on calcule leurs scores de similarité selon la mesure Jaro comme suit (AnHai et al., 2012) :

$$jaro(x, y) = \frac{1}{3} \times [\frac{c}{|x|} + \frac{c}{|y|} + \frac{(c-t)/2}{|c|}]$$

Où :
c est le nombre de caractères communs x_i et y_i (c'est-à-dire, x_i=y_i et $|i - j| \leq min\frac{\{|x|,|y|\}}{2}$) ;
t est le nombre de transpositions (on fait une transposition quand le caractère commun $i^{ème}$ de x et le caractère commun $i^{ème}$ de y sont différents).
- La distance de Jaro-Winkler (Winkler and Thibaudeau, 1991) est principalement utilisée pour la détection des doublons. Cette mesure est utilisée pour traiter de chaînes de caractères courtes comme des noms ou des mots de passe. Elle est conçue pour tenir compte des cas où les deux chaînes ont un score Jaro faible, mais elles partagent un préfixe et donc elles sont encore probablement similaires. Plus la distance Jaro-Winkler entre deux chaînes de caractères est faible, plus elles sont similaires. La distance Jaro-Winkler est calculée comme suit (AnHai et al., 2012) :

$$Jaro - Winkler(x, y) = (1 - PL \times PW) \times Jaro(x, y) + PL \times PW$$

Où :
PL est la longueur du préfixe commun de deux chaînes de caractères (au début de la chaîne jusqu'à un maximum de 4 caractères) ;
PW est un coefficient qui permet de favoriser les chaînes avec un préfixe commun. Winkler a fixé la valeur $p = 0, 1$.
- La distance de q-gram (Esko, 1992) est généralement utilisée pour mesurer la similarité entre les commentaires de deux entités. Elle est exploitée spécialement pour comparer des chaînes de caractères qui constituées plusieurs mots (Zghal, 2010). La mesure q-gram détermine le nombre de séquences qui se trouvent à la fois dans les deux chaînes de caractères. Elle est définie comme suit :

$$q - gram(x, y) = \frac{|ngram(x,n) \cap ngram(y,n)|}{(min(|x|,|y|) - n + 1}$$

Où :
$ngram(x, n)$ est l'ensemble de sous-chaînes de x de longueur n.
b) Les mesures de similarité à base de Wordnet : Dans la littérature, le Wordnet [1] a fournit six mesures de similarité et trois mesures de connexité (relatedness) en se basant sur la base de données lexicale WordNet (Ted et al., 2004).
- **Les mesures de similarité :** Trois mesures de similarité de WordNet (Resnik, Lin et Jiang and Conrath (JCN)) sont basées sur le contenu d'information de LCS (Least Common Subsumer) des concepts.
La mesure Resnik (Resnik, 1995) est basée sur le concept de contenu d'information qui définit la généralité ou la spécificité d'un concept dans un domaine spécifique. La similarité sémantique de Resnik est définie comme suit (Zhao and Ichise, 2013) :

$$simR(c_1, c_2) = IC(lcs(c_1, c_2))$$

où :
$lcs(c_1, c_2)$ est le plus petit commun subsumer de c_1 et c_2 ;
IC est le contenu d'information d'un concept c qui est définie comme suit (Zhao and Ichise, 2013) :

$$IC(c) = -\log(\frac{freq(c)}{freq(root)})$$

où :
$freq(c)$ est la fréquence d'un concept c ;
$freq(root)$ est la fréquence du concept racine, qui compte les occurrences de tous les concepts dans la taxonomie.
La mesure Lin (Dekang, 1998) est basée principalement sur la mesure Resnik qui utilise le contenu d'information du plus petit commun subsumer $lcs(c1, c2)$. Elle est définit comme suit :

$$simL(c_1, c_2) = \frac{2 \times IC(lcs(c_1,c_2))}{IC(c_1) + IC(c_2)}$$

La mesure JCN (Jiang and Conrath) (Jiang and Conrath, 1997) intègre le contenu d'information des concepts IC et le contenu d'information du plus petit commun subsumer lcs. Cette mesure est basée sur la mesure de Renik, elle est définie comme suit (Zhao and Ichise, 2013) :

$$simJCN(c_1, c_2) = \frac{1}{IC(c_1) + IC(c_2) - 2 \times IC(lcs(c_1,c_2))}$$

1. http://wn-similarity.sourceforge.net/

Les trois autres mesures de similarité (LCH, WUP et PATH) sont basées sur des longueurs de chemins entre deux concepts.
LCH (Leacock and Chodorow) (Leacock et al., 1998) est une mesure de similarité sémantique qui détermine le nombre de liens entre deux systèmes d'entrée. Cette mesure trouve le plus court chemin entre deux concepts dans le Wordnet en utilisant la langueur maximale trouvée dans une hiérarchie de type is-a (les relations de subsomptions). L'équation ci-dessous permet de calculer la similarité sémantique réduite entre deux concepts dans le Wordnet (Zhao and Ichise, 2013) :

$$simLCH(c_1, c_2) = -\log(\frac{len(c_1,c_2)}{2\times max_{c\in Wordnet} depth(c)})$$

Où :
$len(c_1, c_2)$ est le plus court chemin entre c_1 et c_2 dans le Wordnet ;
$depth(c)$ est la longueur du chemin entre les concepts c_1, c_2 et la racine (c'est-à-dire, la profondeur des concepts c_1 et c_2).
La mesure WUP (Wu and Palmer) (Wu and Palmer, 1994) calcule la profondeur des LCS de concepts en utilisant la somme de profondeurs des concepts individuels. Cette mesure est définie comme suit (Zhao and Ichise, 2013) :

$$simWUP(c_1, c_2) = \frac{2\times depth(lcs(c_1,c_2))}{len(c_1,lcs(c_1,c_2)+len(c_2,lcs(c_1,c_2)+2\times depth(lcs(c_1,c_2)))}$$

Où :
$depth(lcs(c_1, c_2))$ est la profondeur globale de la hiérarchie.
La mesure PATH est une mesure de référence qui est égale à l'inverse du plus court chemin entre deux concepts. Elle est définie comme suit (Zhao and Ichise, 2013) :

$$simPATH(c_1, c_2) = \frac{1}{len(c_1,c_2)}$$

Où
$len(c_1, c_2)$ est la longueur du plus court chemin entre c_1 et c_2.
- **Les mesures de connexité :** Dans la littérature, il existe trois méthodes qui permettent de calculer la connexité entre deux concepts : HSO, LESK et VECTOR.
La mesure HSO (Hirst and StOnge) (Hirst and St-Onge) classifie les relations dans le Wordnet en trois catégories selon les directions : des liens vers le haut, des liens vers le bas et des liens horizontaux. Ensuite, HSO établit des liens entre les concepts en trouvant un chemin qui n'est pas trop long et ne change pas de direction. HSO établit un lien entre deux concepts si :
- Les deux concepts appartiennent au même synset.
- Les deux concepts appartiennent aux synsets reliés par un lien horizontal.
- L'un des concepts est un mot composé qui comprend le deuxième concept. Dans ce ca, les deux concepts sont reliés par une relation is-a.
La similarité de HSO est calculée comme suit (Zhao and Ichise, 2013) :

$$simHSO(c_1, c_2) = C - len(c_1, c_2) - k \times turns(c_1, c_2)$$

Où :
C et k sont des constants (dans la pratique, $C = 8$ et $K = 1$) ;
$turns(c_1, c_2)$ est le nombre de fois que le chemin entre C_1 et C_2 change de direction.
La mesure LESK (Satanjeev and Ted, 2003) incorpore des informations du Wordnet (c'est-à-dire, les commentaires des concepts) et trouve des chevauchements entre ces commentaires et les concepts qui sont directement liés avec eux (Zhao and Ichise, 2013). Cette mesure considère la similarité entre deux sens comme le nombre de mots en commun dans leurs définitions.
La mesure VECTOR (Patwardhan, 2003) crée une matrice de co-occurrence pour chaque concept utilisé dans le WordNet à partir d'un corpus donné. Ensuite, elle présente chaque concept ou commentaire avec un vecteur qui est la moyenne de ces vecteurs de co-occurrence.
Typiquement, les mesures de similarité syntaxiques et sémantiques, qui sont présentées en dessus, sont principalement utilisées pour le calcule automatique des liens entre les concepts des ontologies.

Résumé

Les données liées, telles qu'elles sont proposées par Tim Berners-Lee (2006), visent à partager et à interconnecter des données structurées sur le web sous forme d'une représentation lisible par la machine pour former un seul espace de données global. Dans ce contexte, Linked Open Data Cloud est un projet qui permet de publier et d'interconnecter des données structurées sur le web conformément aux principes des données liées. Avec le nombre croissant de données disponibles dans le LOD Cloud, le problème d'hétérogénéité des données dans ces sources augmente et, par conséquent, le besoin d'accéder à toutes ces sources via une interface unique et cohérente a été le défi de nombreuses recherches dans le domaine d'intégration des données liées. En fait, le processus d'intégration des données liées nécessite trois étapes principales : l'établissement de liens typés au niveau d'instances et l'alignement de différents vocabulaires utilisés pour décrire les entités ainsi que l'évaluation de la qualité et la fusion des données. Dans la littérature, il existe plusieurs travaux qui visent à réduire l'hétérogénéité en appliquant plusieurs méthodes d'intégration des données sur les données liées. Cependant, ces méthodes ne sont pas totalement satisfaisantes et le problème d'intégration reste ouvert pour la proposition de nouvelles contributions. Pour intégrer des ensembles de données liées, nous avons proposé plusieurs méthodes dans cette thèse. Afin de découvrir des liens typés au niveau d'instances, nous avons proposé une méthode qui vise à réduire le nombre de comparaisons lors du mapping de grands ensembles de données liées. De plus, nous avons développé un outil appelé LDVT (Linked Data Visualization Tool) qui permet de visualiser le résultat du processus de découverte de liens pour vérifier la précision et l'exhaustivité des liens. En outre, nous avons proposé une nouvelle approche pour fusionner les données liées à l'aide d'un algorithme génétique. Notre approche vise à combiner des valeurs conflictuelles de différents ensembles de données pour obtenir une vue unifiée de ces données. Nos méthodes d'intégration des données liées ont été évaluées en utilisant des ensembles de données réels à partir du LOD Cloud. Nous avons également comparé les méthodes proposées avec d'autres méthodes d'intégration de la littérature.

Keywords : Données Liées, Découverte de liens, Web sémantique, Intégration, Visualisation.

Abstract

Linked Data, as proposed by Tim Berners-Lee (2006), aims to share and interlink structured data on the web in a machine-readable representation format to form a single global data space. In this context, the Linked Open Data Cloud is a project that allows publishing and interlinking structured data on the web according to the Linked Data principles. With the ever growing of data sets available in the LOD Cloud, the problem of data heterogeneity in these sources increases and therefore the need to access to all these sources through a single and coherent interface has been the challenge of many research in the field of Linked Data integration. In fact, the Linked Data integration process necessitates three main steps :
The establishment of typed links at instances-level and the alignment of the different vocabularies that are used to describe entities as well as data quality assessment and data fusion. In the literature, there are several works that aim to reduce the heterogeneity by applying several data integration methods on Linked Data. However, these methods are not completely satisfactory and the integration problem remains open for the proposal of new contributions.
In order to integrate the linked datasets, we proposed several methods in this thesis. In order to discover typed links at instances-level, we proposed a method that aims to reduce the number of comparisons when mapping large data sets. In addition, we developed a tool called LDVT (Linked Data Visualization Tool) that allows visualizing the result of the link discovery process to check the accuracy and the completeness of links. Furthermore, we proposed a novel approach to fuse Linked Data using a genetic algorithm. Our approach aims to combine conflicting values from different data sets to obtain a unified view of them. Our methods of Linked Data integration has been evaluated using real data sets from the LOD Cloud. We also compared the proposed methods with others from the literature.

Keywords : Linked Data, Link discovery, Semantic Web, Integration, Visualization.