Découverte de mappings dans un système pair-à-pair sémantique : application à SomeRDFS

François-Élie Calvier

Thèse

Université Paris Sud 11

présentée pour obtenir le grade de

Docteur de l'Université Paris-Sud 11

Spécialité : Informatique

Par

François-Élie CALVIER

Découverte de Mappings dans un Système Pair à Pair Sémantique : Application à SomeRDFS

Composition du jury :

Bernd Amann Professeur à l'Université Paris 6	Examinateur
Zohra Bellahsène Professeur à l'Université Montpellier 2	Examinatrice
Salima Benbernou Professeur à l'Université Paris 5	Rapporteur
Christine Froidevaux Professeur à l'Université Paris-Sud	Examinatrice
Ollivier Haemmerle Professeur à l'Université de Toulouse le Mirail	Rapporteur
Chantal Reynaud Professeur à l'Université Paris-Sud	Directrice

Laboratoire de Recherche en Informatique, INRIA Saclay-île-de-France, U.M.R. CNRS 8623,
Université Paris-Sud, 91405 Orsay Cedex, France

Table des matières

Introduction

Les systèmes pair-à-pair sont connus du grand public pour les possibilités qu'ils offrent en matière de partage de fichiers. De façon générale, un système pair-à-pair permet la réalisation collective d'une tâche par plusieurs unités d'exécution connectées par un réseau. Ces unités sont appelées pairs car elles possèdent toutes un même ensemble de fonctionnalités nécessaires à la réalisation de la tâche collective. Cette organisation particulière, exempte de contrôle centralisé et ne reposant pas sur un modèle hiérarchique fort, permet une grande robustesse lors du passage à l'échelle du Web.

Contexte

Les systèmes pair-à-pair sont des systèmes de partage. La ressource partagée dépend de la tâche à réaliser : on peut citer comme exemples le partage de flux multimédia dans le cas du streaming, le partage de temps de calcul dans le cas du calcul réparti ou le partage d'un service dans le cas de la téléphonie... Pour permettre son identification, chaque ressource doit être décrite par le pair qui la met à disposition. Cette description peut prendre des formes variées. La forme la plus simple de description est le nom de la ressource. Pour avoir accès à une ressource partagée, il est alors nécessaire de connaître son nom exact. Des moteurs de recherche peuvent aider à retrouver le nom d'une ressource, mais leur efficacité est limitée. Pour accéder plus efficacement aux ressources, il convient d'utiliser des mécanismes supplémentaires. On distingue deux grand types de solutions : les travaux portant sur l'intégration de données et ceux portant sur l'intégration de connaissances.

Les travaux sur l'intégration de données dans un cadre distribué portent sur l'optimisation de deux mécanismes permettant aux pairs de connaître les ressources partagées par les autres pairs et d'y accéder : la publication et l'indexation. La publication consiste à informer les autres pairs du partage d'une ressource. L'indexation consiste à créer et stocker un index établissant un lien entre des éléments de description des ressources et la localisation de ces ressources, permettant ainsi de facilement y accéder. Dans ces systèmes, la notion de schéma de description des ressources n'intervient pas, ceci rend impossible une interrogation fine des ressources du réseau.

Les travaux sur l'intégration de connaissances dans un cadre distribué portent sur des systèmes, appelés systèmes d'inférence pair-à-pair, constituant des réseaux de pairs entre lesquels des communications s'établissent via des liens sémantiques, appelés mappings. Chaque pair décrit les ressources qu'il partage à l'aide d'un schéma. Des mécanismes de raisonnement basés sur l'inférence exploitent cette description, ce qui permet à un pair

de réaliser un traitement particulier à partir de ses propres ressources. Ces mêmes mécanismes exploitent également les mappings pour se propager aux autres pairs du réseau et ainsi permettre une réalisation collective du même traitement par l'ensemble des pairs du réseau. Dès lors, il n'est plus nécessaire de connaître précisément le nom des ressources qui interviennent dans des traitements. Les travaux présentés dans cette thèse se situent dans le cadre de ces systèmes.

Le terme pair-à-pair est utilisé par opposition aux architectures centralisées dans lesquelles il existe deux types d'acteurs bien distincts : les serveurs, qui mettent à disposition des ressources, et les clients qui demandent les ressources mises à disposition par les serveurs. La figure 1 montre l'architecture en étoile caractéristique d'un réseau centralisé.

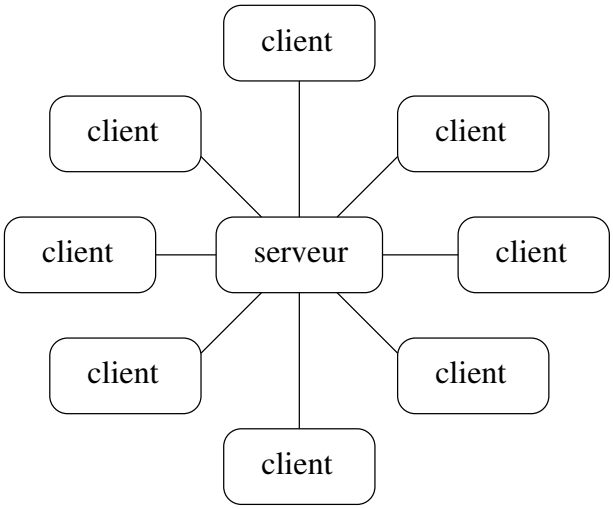

FIG. 1 – Exemple de réseau centralisé

Dans un système pair-à-pair, chaque acteur du système joue alternativement les deux rôles. La figure 2 montre une architecture de réseau pair-à-pair totalement décentralisé. Tous les pairs ont exactement les mêmes fonctionnalités et il n'existe pas de gestion des pairs. Lorsqu'un nouveau pair veut se connecter à ce type de réseau, il doit établir un mapping avec un des pairs appartenant déjà au réseau.

Certains systèmes pair-à-pair ont une architecture mixte. Dans ces systèmes, certains pairs ont le statut particulier de super pair. Ces super pairs sont chargés d'exécuter des tâches supplémentaires nécessaires à la gestion du système. Les super pairs sont choisis pour leur capacité (puissance de calcul, bande passante élevée ou fiabilité) à accomplir les tâches supplémentaires. Les super pairs constituent un réseau pair-à-pair qui peut également avoir ses propres super pairs. Dans ce type d'architecture, les super pairs ont un rôle de serveur pour les autres pairs. Dans certains cas, le réseau pair-à-pair se construit autour du réseau de super pairs. Les pairs se connectent au réseau par l'intermédiaire des super pairs. Une fois connectés, les pairs établissent des liens directs entre eux. Dans d'autres cas, le réseau pair-à-pair pré-existe sans super pairs et certains pairs obtiennent le statut de super pairs lorsqu'ils remplissent un certain critère choisi en fonction de la

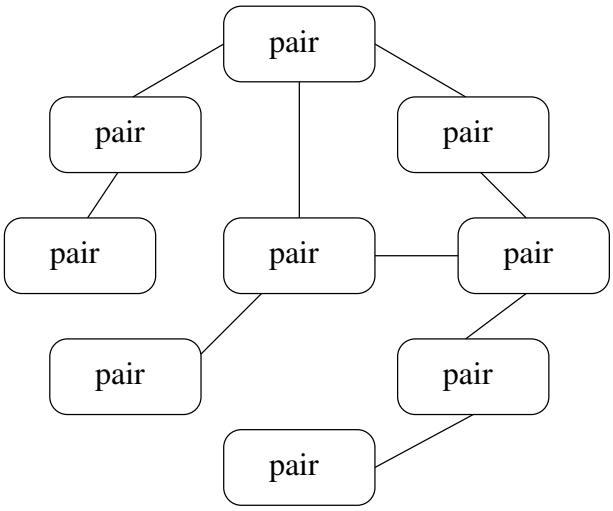

FIG. 2 – Exemple de réseau pair-à-pair

tâche supplémentaire à accomplir.

Les travaux de cette thèse ont été réalisés dans le cadre des systèmes d'inférence pair-à-pair totalement distribués. Tous les pairs ont donc exactement le même rôle et aucun n'a de vue sur la globalité du système. Par ailleurs, chaque pair est autonome. Il peut rejoindre ou quitter le réseau librement et à tout moment. Plus précisément, ce travail a été réalisé dans le cadre du système SomeRDFS, un système pair-à-pair de gestion de données (PDMS) dont les données des pairs sont décrites à l'aide d'ontologies. Une ontologie est " la spécification d'une conceptualisation d'un domaine de connaissance " Gruber (1993), ou, plus simplement, la description formelle d'un point de vue sur un domaine.

Il existe plusieurs langages formels permettant de décrire des ontologies. Les principales recommandations du W3C[1] sont RDF(S) et OWL. Chaque langage possède un pouvoir d'expression qui lui est propre. Ainsi, le choix du langage est important. Selon son pouvoir d'expression, il sera possible de traduire des informations plus ou moins fines et structurantes. Dans SomeRDFS, les ontologies sont décrites en RDF(S).

Problématique

La richesse des réponses aux requêtes posées aux PDMS dépend du nombre de mappings entre les ontologies des différents pairs. Augmenter le nombre de mappings permet donc d'améliorer les réponses données par le système à une requête qui lui a été posée. C'est à l'augmentation de ce nombre de mappings que nous nous intéressons dans cette thèse.

Augmenter le nombre de mappings consiste à découvrir des liens sémantiques entre les éléments des ontologies des différents pairs du système. Ce problème, connu sous le nom d'alignement d'ontologies, fait l'objet de nombreux travaux depuis quelques années.

[1] http://www.w3.org/

L'ensemble des travaux existants considèrent toutefois que les ontologies devant être rapprochées sont connues et bien identifiées. Dans le cadre des systèmes d'inférence pair à pair, cette hypothèse est remise en cause. Les pairs sont autonomes, c'est-à-dire conçus et gérés indépendamment les uns des autres. Chaque pair ne détient que ses propres données, sa propre ontologie et ses propres mappings et, du fait du cadre entièrement distribué, il ignore les ontologies, les mappings et les données des autres pairs. Ce contexte, bien spécifique, rend le problème de la recherche de mises en correspondances entre ontologies tout à fait particulier. Les solutions proposées doivent prendre en compte l'absence de contrôle centralisé des ontologies à apparier. Elles doivent, par ailleurs, permettre d'aligner un nombre d'ontologies qui peut être très important. Chaque pair détenant sa propre ontologie, il peut être nécessaire d'aligner des millions d'ontologies.

Approche proposée

Pour répondre aux problèmes évoqués dans la section précédente, nous proposons des techniques automatiques ou semi-automatiques ainsi qu'une méthodologie de découverte de mises en correspondance entre ontologies prenant en compte la stratégie des pairs concernant les mappings recherchés et basées sur l'exploitation des mécanismes de raisonnement des PDMS. Nous nous intéressons à l'identification de deux sortes de mappings :

1. des raccourcis de mappings correspondant à une composition de mappings existants.
2. des mappings nouveaux et pertinents ne pouvant être inférés en l'état actuel du système.

Le processus de découverte des raccourcis de mappings est basé sur le processus de réponse aux requêtes des utilisateurs mis en œuvre dans SomeRDFS. Ce processus est composé de deux étapes bien distinctes, l'étape de réécriture et l'étape d'évaluation. Les techniques proposées pour découvrir la deuxième catégorie de mappings consistent, dans un premier temps, à identifier des éléments de l'ontologie d'un pair donné qui sont considérés comme particuliers, dans la mesure où ils permettent d'identifier des mises en correspondance jugées intéressantes pour ce pair. Dans un second temps, pour chacun des éléments identifié, des techniques de sélection sont appliquées de façon à rassembler un ensemble de candidats au mapping (éléments d'ontologies distantes) à considérer. Cette approche est bien adaptée à notre contexte d'étude car elle limite le processus d'alignement à un ensemble restreint d'éléments. Enfin, un processus d'alignement spécifique est appliqué aux éléments des ontologies obtenus en sortie de l'étape de sélection. Ce processus met en œuvre des adaptations de techniques existantes ainsi que des techniques innovantes tirant parti des spécificités du cadre de travail.

Contributions

Les contributions apportées par ce travail sont :

1. La proposition d'une méthode de sélection de mappings déductibles à partir du raisonnement mis en œuvre dans SomeRDFS. Ces mappings sont appelés *Raccourcis de Mappings*. La méthode proposée permet de distinguer les cas où ces mappings

sont utiles au système, des cas où ils sont néfastes pour le système. Cette méthode s'appuie principalement sur l'analyse des requêtes posées par les utilisateurs.

2. La proposition d'une méthode de découverte de nouveaux mappings. Elle se fonde sur l'analyse des réponses aux requêtes des utilisateurs et sur la pose de requêtes au système. L'objectif est d'aboutir à des mappings jugés intéressants par l'administrateur d'un pair.
3. La proposition d'un environnement de validation de mappings. L'objectif consiste à mettre à la disposition de l'administrateur un ensemble d'outils pouvant l'aider à valider des mappings qui lui sont suggérés.
4. Une implémentation des méthodes proposées dans cette thèse au sein du prototype SpyWhere, un module de découverte de mappings intégré à SomeRDFS.
5. Un ensemble d'expérimentations réalisées grâce au prototype SpyWhere. Ces expérimentations permettent de valider les propositions de cette thèse.

Organisation du manuscrit

Ce manuscrit comporte 6 chapitres :

chapitre 1 : Ce premier chapitre est un état de l'art correspondant aux travaux que nous avons réalisés, l'alignement d'ontologies et la découverte de mappings dans le cadre de systèmes pair-à-pair. Dans ce chapitre, nous positionnons également nos travaux par rapport à ceux du domaine.

chapitre 2 : Ce chapitre présente les notions clés que nous avons définies et qui sont utilisées dans la suite du manuscrit. Ces notions concernent les critères utilisés pour la sélection et le filtrage des couples d'éléments à aligner : l'optimalité des mappings et la notion de stratégies.

chapitre 3 : Ce chapitre est dédié à la méthode de découverte de *Raccourcis de Mappings*. En précisant la définition de Raccourcis de Mappings, nous mettons en évidence la nécessité d'une sélection des Raccourcis de Mappings à ajouter dans le système. Nous présentons, ensuite une méthode permettant de faire apparaître les Raccourcis de Mappings qu'il est pertinent d'ajouter en exploitant les traces des interactions entre les utilisateurs et le système.

Les chapitres 4 et 5 sont consacrés à la découverte de nouveaux mappings.

chapitre 4 : Nous consacrons ce chapitre à la sélection des couples d'éléments à aligner, c'est-à-dire à la détermination des couples d'éléments qui constitueront l'entrée du processus d'alignement décrit dans le chapitre 5. Nous décrivons une méthode automatique permettant de trouver un ensemble de tels couples à partir d'un élément donné et à l'aide des mécanismes de raisonnement. Nous encadrons cette méthode de sélection par deux méthodes automatiques de filtrage. La première méthode de filtrage permet de choisir les

éléments pour lesquels il est *intéressant* de trouver des mappings. La seconde méthode de filtrage permet de choisir les couples d'éléments qu'il est *pertinent* d'aligner.

chapitre 5 : Ce chapitre présente une méthodologie pour l'alignement en pair-à-pair. Nous y décrivons les techniques d'alignement spécifiques exploitées par cette méthodologie. Ces techniques sont soit des techniques issues de travaux en contexte centralisé adaptées au contexte distribué, soit des techniques nouvelles basées sur les mécanismes de raisonnement mis à disposition dans le système.

chapitre 6 : Dans ce dernier chapitre, nous présentons SpyWhere, un prototype implémentant les principales propositions de cette thèse. Ce prototype nous a permis de réaliser des expérimentations pour valider notre approche. Nous présentons également la démarche expérimentale que nous avons suivie et l'analyse des résultats obtenus.

Enfin, le mémoire s'achève sur une conclusion dans laquelle, d'une part, nous résumons les travaux et contributions détaillées dans cette thèse et, d'autre part, nous évoquons un ensemble de perspectives possibles.

Chapitre 1

État de l'art

Sommaire

Introduction

Dans ce chapitre, nous présentons un état de l'art correspondant aux travaux que nous avons réalisés. Le problème auquel nous nous sommes intéressés portant sur la découverte de mappings entre ontologies, une première partie sera consacrée à l'alignement d'ontologies. Ce problème étant étudié dans le cadre d'un système pair-à-pair, nous décrivons ensuite ce cadre de travail ainsi que les travaux portant sur la découverte de mappings dans ce contexte. Enfin, une troisième partie sera consacrée au positionnement de notre travail par rapport à l'état de l'art. Nous présenterons le système SomeRDFS auquel notre approche a été appliquée, puis nous montrerons comment notre travail se situe par rapport à l'existant.

1.1 L'alignement d'ontologies

La découverte de mappings entre ontologies est un problème bien connu, de nombreux travaux s'y intéressent. Ce domaine de recherche, désigné par l'appellation *alignement*

d'ontologies, est très actif. Cette introduction sera consacrée à la définition de ce qu'est l'alignement d'ontologies, en précisant notamment les étapes qui composent ce processus.

Définition 1 (L'alignement d'ontologies) *L'alignement d'ontologies est une fonction f qui, appliquée à deux ontologies O et O', à un ensemble de mises en correspondances initiales A, à partir d'un ensemble de poids p s'appliquant aux techniques d'alignement mises en oeuvre dans le processus et d'un ensemble de ressources externes r, produit un ensemble de mises en correspondance A' entre les deux ontologies (Euzenat & Shvaiko (2007)).*

$$A' = f(O, O', A, p, r)$$

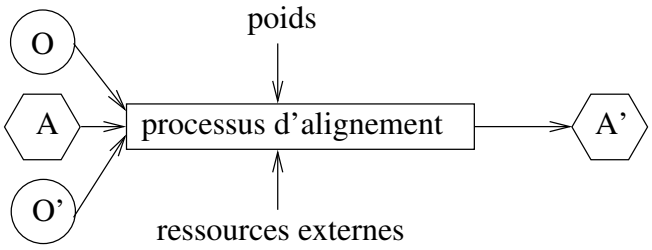

FIG. 1.1 – Processus d'alignement

Le processus d'alignement d'ontologies se compose de différentes étapes que l'on retrouve dans la majorité des systèmes implémentés (Ehrig (2007)). Ces étapes sont schématisées dans la figure 1.2.

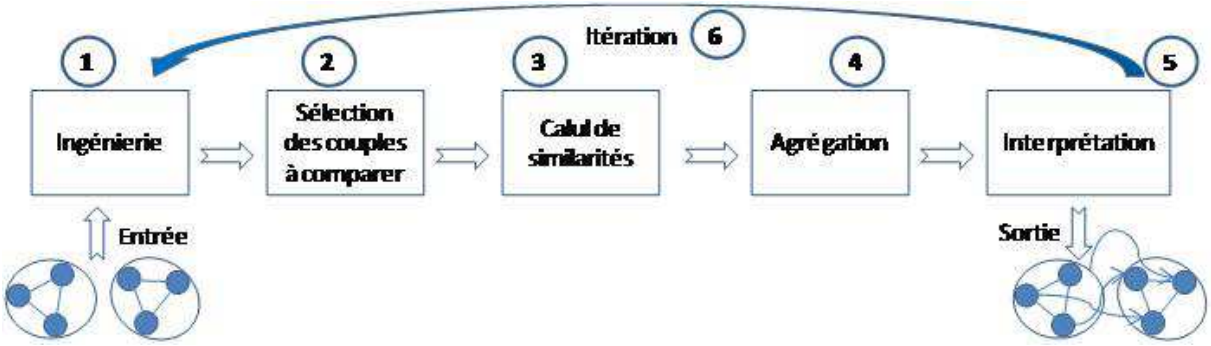

FIG. 1.2 – Les étapes du processus d'alignement

L'entrée comprend les ontologies à aligner, généralement au nombre de deux. Comme la figure 1.1 le montre, elle peut aussi comprendre des alignements.

L'étape d'ingénierie consiste à extraire des entrées, les données sur lesquelles les techniques d'alignement seront appliquées. Le processus peut, par exemple, n'exploiter qu'un sous-ensemble des primitives OWL, uniquement la taxonomie ou juste les noms des entités. Les éléments des ontologies extraits dépendent des techniques d'alignement qui seront ensuite mises en œuvre. Les éléments extraits peuvent correspondre aux identifiants comprenant les URIs ou les labels, aux propriétés (au sens OWL, c'est-à-dire attributs ou

relations) ainsi qu'aux relations de subsomption entre classes ou propriétés, aux déclarations de classes équivalentes à d'autres ou disjointes, à des axiomes. Il est également possible de considérer des éléments extérieurs à l'ontologie tels des ensembles de mots extraits de documents décrivant une instance.

L'étape de sélection des couples à comparer consiste à définir l'ensemble des couples d'éléments qui seront considérés dans le processus d'alignement. Il est ainsi possible de ne considérer que certains couples et d'en ignorer d'autres. Généralement toutefois, les outils d'alignement existants comparent tous les éléments de la première ontologie avec tous les éléments de la seconde ontologie, ou plus précisément, s'il existe des éléments de différentes natures, tous les concepts de la première ontologie avec tous les concepts de la seconde, toutes les relations de l'une avec les relations de l'autre et toutes les instances de l'une avec les instances de l'autre.

L'étape 3 calcule une similarité pour chaque couple d'éléments sélectionné dans l'étape 2 et en utilisant les données extraites à l'étape 1 correspondant à ces éléments. La similarité peut être calculée de différentes façons. Il existe de nombreuses fonctions de similarité qui souvent, reposent sur le calcul d'une distance sémantique. La distance sémantique entre deux éléments est une valeur normalisée comprise entre 0 et 1. La mesure de similarité (notée σ) correspond alors au complémentaire de la mesure de distance (notée δ).

$$\sigma(x, y) = 1 - \delta(x, y)$$

La distance (ou la similarité) peut être calculée de différentes façons correspondant, chacune, à une technique d'alignement. Les techniques d'alignement d'ontologies peuvent être classées selon le type d'éléments sur lesquels se base le processus d'alignement. On distingue alors les techniques terminologiques (exploitant les labels des différents composants des ontologies), les techniques structurelles (exploitant la structure des ontologies ou de leurs composants), les techniques extentionnelles (exploitant les données peuplant les ontologies) et les techniques sémantiques (comparant les interprétations, ou plus exactement les modèles, des entités considérées). Notons que plusieurs mesures de similarités peuvent être calculées pour un même couple d'éléments. Notons également que les techniques d'alignement diffèrent selon qu'elles calculent une similarité locale ou globale. Le calcul d'une similarité locale ne tient compte que du voisinage des éléments. Le calcul de similarités globales tient compte du fait que la similarité d'éléments peut dépendre de la similarité d'autres éléments. Ainsi la similarité de deux classes dépend par exemple de la distance entre leurs instances. La mise en œuvre des approches globales peut passer par la propagation de mesures de similarité dans un graphe. Il en est ainsi du système Rondo (Melnik *et al.* (2003)) basé sur l'algorithme de Similarité Flooding (SF) (Melnik *et al.* (2002)). SF met en correspondance des schémas convertis en graphes orientés et étiquetés. Des éléments des deux schémas distincts sont dits similaires si leurs éléments adjacents sont similaires. La similarité entre deux éléments est propagée à leurs voisins respectifs (les éléments reliés par des arcs) jusqu'à obtention d'un point fixe. L'algorithme exploite principalement les étiquettes des arcs, pour déterminer à chaque itération le poids et les nœuds vers lesquels il doit propager les similarités. Les approches globales peuvent également être mises en œuvre via la traduction des similarités en un ensemble d'équations

traduisant les dépendances entre les similarités et résolues par des techniques numériques. Chaque variable correspond à la similarité d'un couple d'éléments. Il y a autant d'équations que de variables. Cette approche est celle adoptée dans le système OLA (Euzenat & Valtchev (2004a)).

Dans le cas où plusieurs mesures de similarité sont calculées pour un même couple d'éléments, ces différentes valeurs peuvent être combinées de façon à obtenir, in fine, une seule valeur. Cette valeur unique résulte rarement d'une moyenne arithmétique simple. Elle est généralement obtenue en affectant des pondérations aux différentes mesures préalablement calculées. La détermination de ces pondérations, ou poids, est délicate. Des poids peuvent être affectés par défaut. Il peut également s'agir de paramètres que l'utilisateur peut modifier selon les ontologies alignées.

Enfin, l'étape finale du processus d'alignement est une étape d'interprétation consistant à déduire des alignements à partir des mesures de similarités préalablement calculées, individuelles ou résultant de l'agrégation de mesures individuelles. Les approches les plus courantes comparent les mesures obtenues à des seuils. Lorsque plusieurs techniques de similarité ont été appliquées sans donner lieu à une agrégation de mesures de similarité, le processus d'interprétation peut aussi consister à déduire des alignements à partir de la prise en considération des résultats des différentes techniques.

Lorsqu'une approche globale est adoptée, la similarité d'un couple d'éléments peut influencer la similarité d'un autre couple d'éléments. Par exemple, le fait que des instances soient équivalentes affecte la similarité des concepts correspondants. Dans ce cas, il est nécessaire de procéder par itérations. La similarité d'un couple d'élément sera donc recalculée à chaque itération en tenant compte de la similarité des couples d'éléments voisins.

De nombreux systèmes d'alignement mettent en œuvre ce processus d'alignement en exploitant des techniques variées. Nous présentons ci-après les techniques les plus usitées. Plusieurs classifications de techniques d'alignement ont été proposées dans la littérature, comme celle de Rahm & Bernstein (2001), de Kalfoglou & Schorlemmer (2003) et de Shvaiko & Euzenat (2005). Nous nous proposons, dans un premier temps, de présenter les différentes techniques d'alignement selon la classification de (Euzenat & Shvaiko (2007)). Dans un deuxième temps, nous revenons sur les techniques basées sur l'exploitation de ressources externes. Enfin, nous présentons comment ces différentes techniques présentées sont combinées au sein d'outils d'alignement.

1.1.1 Techniques d'alignement

Techniques terminologiques

Les techniques terminologiques comparent les termes utilisés comme labels pour désigner les différents éléments composant des ontologies. Ces techniques se répartissent en deux familles : les techniques syntaxiques et les techniques linguistiques.

Techniques syntaxiques : Les techniques syntaxiques sont basées sur des comparaisons de chaînes de caractères. Elles supposent que deux concepts sont sémantiquement proches si les termes qui les désignent sont similaires syntaxiquement. La similarité sémantique entre concepts est calculée à partir de leur distance sémantique et celle-ci est remplacée par la distance, plus facilement calculable, séparant les labels désignant respectivement ces concepts.

Il existe de nombreuses façons de calculer la distance entre deux deux chaînes de caractères s_1 et s_2. Nous proposons quelques exemples de ces techniques :

- **préfixe/suffixe** : ces techniques consistent à vérifier si s_1 commence (resp. se termine) par s_2 et vice versa. L'inclusion de chaîne constitue un lien terminologique fort. Les concepts désignés par ces deux chaînes sont donc supposés proches.
- **distance d'édition** (ou distance de Levenshtein (Levenshtein (1966))) : à chacune des trois opérations d'édition de chaîne (insertion, suppression et modification d'un caractère) est associé un coût. La distance d'édition $ed(s_1, s_2)$ est alors le coût associé à l'ensemble minimal des opérations nécessaires pour transformer s_1 en s_2.
- **mesure de Lin** (Lin (1998)) : cette mesure est une adaptation de la mesure de Jaccard (Jaccard (1901)) à la notion de n-gramme (Shannon (1948)). Les n-grammes d'une chaîne de caractères S donnée sont l'ensemble des sous-chaînes de n caractères consécutifs de S. Les expérimentations effectuées dans Langdon (2000) ont montré que les meilleurs résultats sont obtenus pour n = 3. On parle alors de 3-grammes ou tri-grammes. La mesure de Jaccard permet d'évaluer la similarité entre deux ensembles. Elle correspond au ratio du nombre d'éléments communs sur le nombre total d'éléments appartenant à ces ensembles. Une valeur de 1 est obtenue lorsque les ensembles sont identiques et une valeur de 0 est obtenue lorsque les ensembles sont disjoints. Dans la mesure de Lin, tri(x) représente l'ensemble des tri-grammes de la chaîne de caractères x et P(t) représente la probabilité d'apparition du tri-gramme t dans les termes du corpus. Cette probabilité est supposée indépendante des autres tri-grammes de la chaîne x et est estimée par la fréquence du tri-gramme dans le corpus.

Définition 2 (Mesure de Jaccard)

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Définition 3 (Mesure de Lin)
Soit $P(t)$ la probabilité d'apparition d'un tri-gramme t

$$L(x, y) = \frac{2 \times \Sigma^n_{t \in tri(x) \cap tri(y)} logP(t)}{\Sigma^n_{t \in tri(x)} logP(t) + \Sigma^n_{t \in tri(y)} logP(t)}$$

La comparaison des labels ne permet pas seulement d'identifier des liens d'équivalence. Lorsque deux concepts sont désignés par des labels identiques ou quasi identiques (variation de casse), un lien d'équivalence est trouvé. Dans les autres cas, lorsque la distance entre les deux labels est inférieure à un seuil choisi, il existe un lien de proximité entre les deux concepts. Dans certains cas, les liens de proximité identifiés par les techniques préfixe/suffixe peuvent être interprétés comme des liens de subsomption.

Cependant, ces techniques sont prises en défaut dans les cas d'homonymie. En effet, la distance entre les labels est alors nulle mais les concepts désignés peuvent être très éloignés. Inversement, dans le cas des synonymes, des labels différents peuvent désigner des concepts sémantiquement proches.

Techniques linguistiques : Les techniques linguistiques sont des techniques plus avancées dans le sens où le label d'un élément n'est pas exploité directement. Elles consistent à utiliser des techniques d'analyse linguistique ou des ressources linguistiques. Les techniques de comparaison de chaînes de caractères utilisées après analyse linguistique ou lors de l'utilisation de ressources linguistiques sont généralement assez simples (égalité).

Les techniques d'analyse linguistique (lemmatisation, étude morphosyntaxique, normalisation,...) permettent de trouver la forme standard de chacun des labels comparés. Deux termes sont alors proches s'il possèdent la même forme standard. Ces techniques permettent également d'identifier et d'interpréter les préfixes et suffixes présents dans un terme. Ceci peut préciser le type de lien existant entre les termes.

Une ressource linguistique (dictionnaire, thesaurus) décrit les liens sémantiques existant entre les différents termes d'un vocabulaire. L'exploitation de ressources linguistiques consiste à vérifier si un lien sémantique existe, dans ces ressources, entre deux termes des ontologies alignées. Cette exploitation est particulièrement utile dans le cas des synonymes ou lorsque les ontologies sont construites dans des langues différentes.

Les techniques linguistiques permettent d'identifier des liens sémantiques (synonymie, antonymie, hyponymie,...) entre des concepts. Elles permettent également le traitement des labels composés de plusieurs mots. Cependant, la transformation des liens sémantiques entre des termes en liens sémantiques entre des concepts peut poser problème. Par exemple, le lien entre deux concepts désignés par des termes synonymes n'est pas nécessairement un lien d'équivalence.

Techniques structurelles

Les techniques structurelles comparent différentes structures. On distingue deux types de structures : la structure interne d'un élément et la structure externe (ou relationnelle) correspondant à l'organisation topologique des éléments au sein de l'ontologie à laquelle ils appartiennent.

Structure interne : Chaque concept peut être considéré comme une structure définie par un ensemble d'attributs et de relations. On parle alors de structure interne. Chaque attribut possède un nom, une multiplicité et un type de données associé. De même, chaque relation possède un nom, des cardinalités, un domaine et une portée. Comparer deux concepts revient alors à comparer leurs attributs et relations respectifs sur la base des multiplicités, cardinalités, types de données, domaines et portées. Notons que la comparaison de la structure interne ne permet pas réellement d'identifier des similarités entre concepts. Les techniques exploitant la structure interne ont ainsi généralement pour objectif d'identifier des dissimilarités fortes.

Structure relationnelle Une ontologie peut être vue comme un graphe dont les concepts constituent les nœuds et dont les relations et les liens taxonomiques ou méréologiques constituent les arêtes. On parle alors de structure relationnelle. L'alignement de deux ontologies correspond dans ce cas à la recherche d'un homomorphisme de graphes (Garey & Johnson (1979)). La plupart des techniques exploitant la structure relationnelle n'utilisent que les liens taxonomiques.

Dans la pratique, les techniques exploitant la structure relationnelle des ontologies nécessitent un ensemble initial de correspondances entre les deux ontologies ou l'utilisation d'une ontologie de référence. Les deux ontologies sont alors considérées comme un unique graphe et la distance entre deux concepts correspond à une fonction du nombre d'arêtes qui les séparent dans ce graphe. Il existent de nombreuses fonctions pour calculer cette distance. Nous en proposons quelques exemples :

- **dissimilarité topologique** (Valtchev & Euzenat (1997)) : il s'agit de la mesure la plus communément utilisée. La distance entre deux concepts est la longueur du chemin le plus court entre ces concepts.
- **Wu-Palmer** (Wu & Palmer (1994)) : la longueur du chemin entre deux concepts n'a pas la même valeur lorsque les concepts sont généraux et lorsqu'ils sont spécifiques. Des concepts généraux très éloignés sémantiquement peuvent être séparés par un chemin très court (par exemple, s'ils sont sous la racine). À l'inverse, des concepts spécifiques relativement proches peuvent être séparés par un chemin de longueur moyenne si la granularité de l'ontologie le permet. La mesure Wu-Palmer intègre cette différence en prenant en compte, en plus de la longueur du chemin entre deux concepts, la distance entre les concepts et la racine.
- **généralisants/spécialisants de concepts similaires** (Dieng & Hug (1998)) : lorsque deux concepts sont similaires, leurs généralisants et spécialisants respectifs sont également similaires. Cette mesure est très intuitive mais son exploitation est délicate dans le cas de spécialisants/généralisants multiples.

Définition 4 (similarité de Wu-Palmer)
Soit $\delta(c_1, c_2)$ le nombre d'arcs séparant les concepts c_1 et c_2, soit $c_1 \wedge c_2$ l'ensemble des généralisants communs de c_1 et c_2 et soit r la racine de l'ontologie.

$$WP(c_1, c_2) = \frac{2 \times \delta(c_1 \wedge c_2, r)}{\delta(c_1, c_1 \wedge c_2) + \delta(c_2, c_1 \wedge c_2) + 2 \times \delta(c_1 \wedge c_2, r)}$$

Techniques extensionnelles

Les techniques extensionnelles comparent les instances associées à chaque concept. Lorsque deux concepts représentent un même ensemble d'instances, il y a de fortes chances que ces concepts soient similaires. À l'inverse, deux concepts pour lesquels il n'existe aucune instance commune ont peu de chances d'être similaires (Larson *et al.* (1989) et Sheth *et al.* (1988)). Cependant, cette analyse n'a de sens que si les deux ontologies comparées décrivent des ensembles de données identiques. Pour cela, il est nécessaire, soit de disposer de deux ontologies à comparer peuplées avec un jeu de données identiques, soit de faire de la réconciliation de références (Fellegi & Sunter (1969), Lim *et al.* (1993), Elfeky *et al.* (2002) et Saïs *et al.* (2009)).

Définition 5 (Similarité Jaccard)
Soit $P(X)$ la probabilité qu'une instance soit dans l'ensemble X.

$$\sigma(A, B) = \frac{P(A \cap B)}{P(A \cup B)}$$

Ces techniques sont intéressantes lorsque les données peuvent être identifiées de manière unique et que chaque concept décrit, en moyenne, un nombre conséquent d'instances.

Techniques sémantiques

Les méthodes sémantiques utilisent la sémantique de la théorie des modèles. Ce sont des méthodes déductives dont la mise en œuvre nécessite de disposer au départ d'alignements initiaux. Ceci est réalisé via une phase de pré-traitement dont l'objectif est de générer des ancres, c'est-à-dire des entités déclarées comme étant équivalentes après comparaison de leur nom ou déclarées équivalentes par l'utilisateur. Ainsi, nous incluons, dans les techniques sémantiques, des méthodes d'ancrage d'ontologies. Ces méthodes sont basées sur l'exploitation d'ontologies externes représentées dans un langage formel de façon à pouvoir raisonner dessus de manière automatique. Les alignements obtenus à l'aide de ces ressources externes sont ensuite exploités par des techniques déductives.

Techniques basées sur des ontologies externes : Ces ontologies, dites de background, définissent le contexte d'interprétation commun aux deux ontologies à aligner. L'idée sous-jacente est qu'une telle ontologie avec une bonne couverture du domaine d'application des ontologies à aligner, aide à sélectionner le sens devant être retenu pour les termes ayant plusieurs significations possibles. Ces ontologies diffèrent selon qu'elles sont spécifiques ou générales, selon le caractère plus ou moins formel du langage de représentation, selon qu'il s'agit d'une ontologie faisant référence dans le domaine ou non. L'exploitation d'une ontologie externe passe par une phase d'ancrage et de dérivation. Ce processus est décrit plus en détail dans la partie suivante.

Techniques déductives : Les techniques sémantiques ont pour objectif de rechercher des correspondances A telles que O, $O' \models$ A, A correspondant à une formule du langage, par exemple une relation de subsomption entre e et e' (e $\sqsubseteq$ e'). Ces techniques servent aussi à tester la satisfiabilité des alignements, en particulier pour supprimer des alignements qui conduiraient à des inconsistances dans l'ontologie issue d'un processus de fusion. Les techniques sémantiques peuvent correspondre au test de la satisfiabilité de formules propositionnelles (SAT). Ainsi, l'approche décrite dans Giunchiglia & Shvaiko (2003), Bouquet & Serafini (2003), Giunchiglia *et al.* (2004) et Shvaiko (2006) consiste (1) à construire une théorie du domaine, via l'utilisation de ressources externes et de matcheurs, constituée d'un ensemble d'axiomes relatifs aux ontologies alignées, (2) à construire une formule de matching r(c,c') pour chaque couple de concepts (c, c') des deux ontologies puis (3) à vérifier la validité de la formule : Axiomes $\models$ r(c,c'). Une formule propositionnelle est valide si et seulement si sa négation est insatisfiable, ce qui est vérifiable en utilisant un solveur SAT. Les techniques sémantiques peuvent également correspondre à des techniques de raisonnement de la logique de description. En logique de

description, les relations $(=, \sqsubseteq, \sqsupseteq, \perp)$ peuvent être exprimées par rapport à la subsomption. Le test de subsomption peut ainsi être utilisé pour établir des relations entre classes.

Les techniques sémantiques ont besoin de disposer d'alignements initiaux. En revanche, elles sont complètes, c'est-à-dire qu'elles permettent de trouver toutes les correspondances qui satisfont les alignements initiaux. Elles sont également consistantes dans la mesure où elles permettent de trouver des correspondances conduisant à des inconsistances. Ces méthodes sont aujourd'hui encore peu développées. Les techniques sémantiques, à la différence des autres techniques, ne sont pas étudiées dans le domaine de l'alignement de bases de données où la sémantique est très faible.

Techniques exploitant des ressources externes

Les techniques exploitant des ressources externes consistent à exploiter des mises en correspondance déjà présentes ou identifiables dans des ressources différentes de celles à aligner.

La description du processus présentée ci-dessous et le schéma qui l'illustre (cf. figure 1.3) s'appuient largement sur les travaux de Sabou *et al.* (2006) et de Aleksovski *et al.* (2006a). L'approche générale suivie par la plupart des travaux utilisant des ressources externes se décompose en deux phases : l'ancrage et la dérivation.

L'ancrage consiste à apparier chacun des éléments des ontologies à aligner avec un ou des éléments de la ressource externe, c'est-à-dire, à identifier l'ensemble des mappings entre la ressource externe et chacune des ontologies à aligner prises indépendamment. Les éléments de la ressource externe mis en correspondance avec ceux des ontologies à aligner sont appelés des ancres ou points d'ancrage. La phase d'ancrage est le plus souvent réalisée à l'aide de techniques terminologiques syntaxiques.

La dérivation consiste à s'appuyer sur la ressource externe pour :

- soit rechercher s'il existe des liens entre les différents points d'ancrage identifiés, afin d'essayer d'en dériver des mappings entre les éléments des ontologies à apparier.
- soit utiliser une mesure de similarité entre les éléments de la ressource externe, pour identifier pour chaque ancre de la ressource externe associée à un élément de l'ontologie source, l'ancre associée à un élément de l'ontologie cible qui lui est le plus similaire.

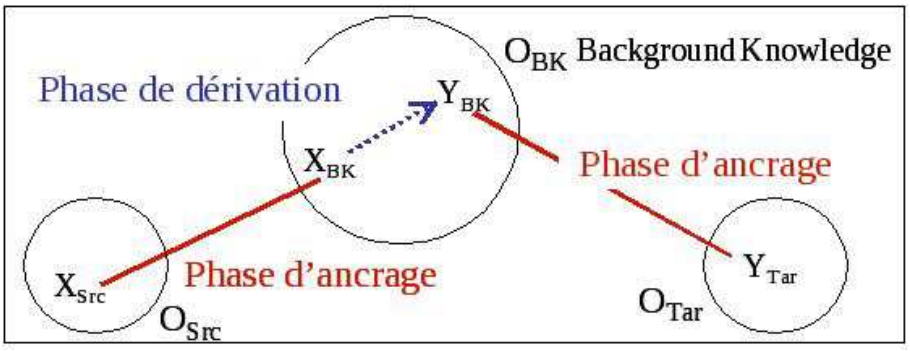

FIG. 1.3 – Schéma général de l'identification d'un mapping

A partir de ce schéma général, les travaux exploitant des ressources externes se diffé-

rencient en fonction des caractéristiques des ressources employées comme support, suivant que celles-ci sont des ontologies à la thématique ciblée ou au contraire une ressource généraliste comme WordNet. Nous présenterons tout d'abord les travaux s'appuyant sur la ressource généraliste WordNet, puis ceux ayant recours à des ressources externes ciblées.

Travaux basés sur l'utilisation de la ressource WordNet : WordNet est une ressource lexicale de langue anglaise, disponible sur internet, qui regroupe des termes (noms, verbes, adjectifs et adverbes) en ensembles de synonymes appelés synsets. Un synset regroupe tous les termes dénotant un concept donné. Le terme associé à un concept est représenté sous une forme lexicalisée, sans marque de féminin ni de pluriel. Les synsets sont liés entre eux par des relations sémantiques :

- relation de généralisation/spécialisation
- relation composant/composé

WordNet peut être utilisé de différentes façons pour la recherche de mappings. Une première technique consiste, comme dans Le *et al.* (2004), à étendre systématiquement le label d'un concept avec les synonymes appartenant au synset de chaque terme du label dans WordNet.

Une autre utilisation consiste à n'exploiter que les relations de généralisation/spécialisation entre les différents synsets (assimilables à des concepts) et à considérer WordNet comme une hiérarchie. Dans Giunchiglia *et al.* (2006), des relations d'équivalence entre deux nœuds sont inférées lorsque leur distance en nombre d'arcs, en suivant les liens de généralisation/spécialisation, est inférieure à un certain seuil.

D'autres travaux s'appuient sur cette hiérarchie pour calculer une mesure de similarité entre deux nœuds. C'est le cas de l'outil WordNet::Similarity (Pedersen *et al.* (2004)) disponible sur le Web. Il permet de calculer de façon interactive une similarité numérique entre deux concepts quelconques, en choisissant la mesure de similarité employée parmi plusieurs. Cette technique est également utilisée par Kalfoglou & Hu (2005) dans un des modules d'appariement, WNNameMatcher, pour identifier pour chaque ancre d'un concept de l'ontologie source, l'ancre du concept de l'ontologie cible qui lui est le plus similaire. La mesure de similarité employée est Wu-Palmer (cf. 1.1.1). Cette même mesure a été utilisée dans une des techniques d'alignement mises en œuvre dans TaxoMap (Reynaud & Safar (2007)). Elle donne des résultats pertinents quand les domaines d'application des ontologies à comparer sont proches et très focalisés. En revanche, lorsque les domaines d'application sont larges et ne se recoupent pas, les résultats sont beaucoup moins satisfaisants. Le problème est dû au phénomène de polysémie, i.e. aux différents synsets auxquels un même terme peut appartenir. Cette polysémie entraîne fréquemment des contre-sens et donc des rapprochements erronés.

Le problème de la polysémie est traité par le système S-Match (Giunchiglia *et al.* (2004)). Ce système calcule des relations logiques (telles que l'équivalence ou la subsomption) entre les concepts et traite le problème d'alignement entre deux nœuds de deux taxonomies comme un problème de satisfiabilité de formules de la logique propositionnelle.

Une phase de désambiguïsation plus simple est effectuée dans Mougin *et al.* (2006). Dans ces travaux, WordNet est utilisé comme ressource complémentaire pour aider à valider des mappings entre des ressources Web et l'ontologie médicale UMLS. L'ancrage des différents termes est effectué dans WordNet. Lorsque plusieurs mappings sont trouvés,

on privilégie ceux qui s'appuient sur des synsets dont la définition ou les hypernymes contiennent des termes appartenant à une liste prédéfinie de mots-clés du domaine biomédical.

Les travaux s'appuyant sur des ressources externes ciblées : Le problème de la polysémie évoqué dans le paragraphe précédent est évité quand les ressources qui servent de support sont ciblées. Dans un contexte clairement défini, les homonymes sont rares et la phase de dérivation peut être plus simple, sans être perturbée par les ambiguïtés.

Les travaux présentés ici recherchent des relations entre les ancres des concepts des ontologies à aligner en s'appuyant sur des règles de dérivation sémantiques.

Parmi les travaux exploitant ces règles de dérivation, certains supposent que la recherche peut s'effectuer sur une ontologie de support unique, préalablement identifiée et qui couvre a priori tous les concepts des ontologies à apparier. Ainsi dans Aleksovski *et al.* (2006a), dans Aleksovski *et al.* (2006b) ou dans Zhang & Bodenreider (2006), deux travaux dans le domaine médical, l'ontologie externe est plus complète et plus détaillée que les deux ontologies à rapprocher, et contient une description en compréhension du domaine des deux autres.

Dans Aleksovski *et al.* (2006a), les concepts à rapprocher sont des éléments issus de deux listes de vocabulaires plats, non structurés. L'ontologie externe utilisée pour rechercher les dérivations est une ontologie représentant des points de vue multiples (ou aspects), ce qui permet d'identifier plusieurs dérivations entre deux points d'ancrage, suivant les différents aspects. Dans Aleksovski *et al.* (2006b) ou dans Zhang & Bodenreider (2006), les concepts à rapprocher appartiennent à deux ontologies structurées par des relations de généralisation/spécialisation et l'ontologie externe contient des relations de type is-a et part-of.

A l'inverse, d'autres travaux considèrent que la recherche de dérivation ne peut s'effectuer qu'au sein de multiples ontologies de support, sélectionnées dynamiquement. Ainsi, dans Sabou *et al.* (2006), les auteurs supposent qu'il n'existe pas a priori, pour un domaine donné, une ontologie qui soit plus complète et plus détaillée que les deux ontologies à rapprocher, et qui puisse seule servir de support. Ils proposent donc d'exploiter l'ensemble des ontologies accessibles sur le Web par l'intermédiaire du moteur de recherche sémantique Swoogle. Pour identifier l'existence d'un mapping, les auteurs proposent de rechercher à la volée une ontologie qui permette l'ancrage simultané des deux concepts à apparier, puis de chercher s'il existe une dérivation entre les deux ancres dans l'ontologie considérée. Si une telle dérivation n'existe pas, le processus repart dans la recherche automatique d'une nouvelle ontologie permettant l'ancrage des deux concepts. L'approche peut paraître beaucoup plus coûteuse que la précédente puisqu'elle travaille séquentiellement sur toutes les paires de concepts possibles et qu'elle conduit a priori à répéter n fois la phase d'ancrage d'un même concept dans la même ontologie si on essaye de l'apparier à n concepts différents. Cependant, elle permet d'identifier à la volée, sans choix manuel préalable, les ontologies susceptibles de servir même à la découverte d'un seul mapping et elle est parallélisable. Lorsqu'aucune ontologie ne permet l'ancrage simultané des deux concepts à apparier, l'approche précédente peut être étendue récursivement en travaillant

sur plusieurs ontologies à la fois.

Même si elle peut être parallélisée, cette dernière stratégie reste encore plus coûteuse que la précédente. De plus, le fait de s'appuyer sur des ontologies identifiées à la volée fait resurgir le problème de la polysémie.

Pour détecter et rejeter les mappings incorrects dus à la polysémie, les auteurs proposent dans Gracia *et al.* (2007) de compléter leur approche en introduisant une phase d'étude des synsets associés dans WordNet aux termes intervenant dans les mappings obtenus. Deux termes X et Y sont considérés comme sémantiquement similaires s'il existe un synset S de X qui soit aussi un synset de Y ou qui soit relié à un synset de Y par une suite de relations de généralisation/spécialisation dans WordNet. Un mapping doit être rejeté si les termes désignant les éléments appariés ne sont pas sémantiquement similaires ou si les synsets considérés ne sont liés à aucun synset du terme désignant un des ancêtres des éléments appariés dans leurs ontologies d'origine.

1.1.2 Stratégie et outils d'alignement

Les techniques décrites dans la section qui précède sont des éléments de base pour l'alignement d'ontologies. Elles sont exécutées dans la phase 3 du processus schématisé figure 1.2. Les résultats de ces différentes techniques sont ensuite agrégés dans la phase 4 avant la phase d'interprétation produisant l'ensemble des mises en correspondance entre les deux ontologies à aligner. La phase d'agrégation met en œuvre une stratégie d'alignement. Cette stratégie indique comment les résultats des techniques mises en œuvre en phase 3 sont considérés. Dans cette section, nous décrivons les principales stratégies implémentées dans les systèmes actuels d'alignement pour lesquels nous énonçons, dans le tableau 1.1, les types de techniques mises en œuvre.

Les stratégies sont variées. Plusieurs techniques peuvent être appliquées à un même couple d'éléments ; on parle alors de composition. Plusieurs techniques peuvent être appliquées à différentes caractéristiques d'un même couple d'éléments, ceux-ci ayant de multiples facettes ; on parle alors d'agrégation au sens strict du terme.

Il existe différents modes de composition.

Les techniques peuvent être exécutées séquentiellement : on exécutera d'abord une technique terminologique basée sur la comparaison des labels des concepts, puis une technique structurelle, par exemple. L'objectif visé peut être l'intégration de données en ligne de façon à les fusionner mais, le plus souvent, on cherche à améliorer notre connaissance sur l'existence d'une similarité entre les éléments rapprochés. Ainsi, la technique structurelle pourra n'être appliquée que sur les couples d'éléments dont la technique terminologique a montré qu'ils pouvaient être similaires sémantiquement. Cette stratégie est mise en œuvre dans de nombreux outils d'alignement tels que Cupid (Madhavan *et al.* (2001)), Falcon-AO (Jian *et al.* (2005)). Il est également possible d'appliquer un ensemble de techniques séquentiellement mais les alignements proposés ne résultent de l'application que d'une technique donnée et d'une seule. Cette stratégie est mise en œuvre dans le système TaxoMap (Hamdi *et al.* (2009)). Une première technique est appliquée sur l'ensemble des couples d'éléments étudiés. Une seconde technique est appliquée sur l'en-

semble des couples pour lesquels un alignement n'a pas été trouvé à l'aide de la première technique, etc. L'ordre d'application des techniques est dans ce cas très important. Dans TaxoMap, il s'agit d'un paramètre que l'utilisateur peut modifier selon les spécificités des ontologies à aligner.

Les techniques d'alignement peuvent aussi être exécutées indépendamment les unes des autres. Dans ce cas, la stratégie peut consister à choisir le résultat le plus pertinent ou à fusionner les résultats. On pourra ainsi sélectionner les correspondances générées par toutes les techniques appliquées (ce qui revient à appliquer l'opérateur d'intersection sur l'ensemble des correspondances générées par chaque technique), ou sélectionner toutes les correspondances en leur associant un degré de confiance. Ainsi le système H-Match (Castano *et al.* (2006)) calcule des similarités linguistiques sur les noms et des similarités structurelles sur les contextes (les concepts voisins liés par des liens taxonomiques ou de composition). Une mesure globale de similarité sémantique est ensuite déterminée ; elle correspond à une combinaison linéaire des similarités linguistiques et structurelles. Cette strétégie est également mise en œuvre dans le système oMap (Straccia & Troncy (2005)) qui comprend plusieurs matchers exécutant des techniques terminologiques, extensionnelles et sémantiques. Ces techniques peuvent être exécutées en parallèle ou en séquence. L'exécution en parallèle est suivie d'une agrégation pondérée des résultats fournis par chaque technique. Les poids correspondent à la confiance accordée à chaque matcheur.

Lorsque des entités ont de multiples caractéristiques, on peut calculer la similarité pour chacune d'elles et agréger les résultats de façon à obtenir une valeur unique de similarité entre les entités. Ainsi, pour calculer la similarité entre deux classes, on peut agréger la similarité obtenue par comparaison de leur nom, de leurs super-classes, de leurs instances, de leurs propriétés. Des poids sont, dans ce cas, affectés aux différentes similarités intervenant dans le calcul. Cette strétégie est, par exemple, mise en œuvre dans le système OLA (OWL Lite Aligner) (Euzenat & Valtchev (2004b)) conçu avec l'idée d'équilibrer la contribution de chacun des composants des ontologies OWL (classes, contraintes, instances). Les ontologies sont traduites sous forme de graphes. La similarité entre deux nœuds dépend du type du nœud (classe, propriété, ..) et prend en compte toutes les caractéristiques de ce type de nœud (super-classes, propriétés, ...).

Notons que le rôle de l'utilisateur varie selon les techniques et les stratégies. L'utilisateur peut fournir les alignements initiaux. Il peut fixer les paramètres d'exécution. Ceux-ci peuvent consister à déterminer les techniques d'alignement à exécuter et leur ordre d'éxécution, comme cela est fait dans TaxoMap. Il s'agit, le plus souvent, de fixer les poids des techniques et les seuils pour déterminer la pertinence d'un alignement. La détermination des ces paramètres n'est pas simple, ce qui explique certains travaux actuels portant sur le raffinement de l'alignement. Le système COMA++ (Do & Rahm (2007)), par exemple, a pour objectif de construire des outils d'alignement puissants par combinaison d'outils existants puis de raffiner les résultats d'alignement obtenus considérés comme préliminaires. Le processus de raffinement est totalement automatique. Il consiste à réappliquer le processus d'alignement de COMA++ sur des groupes d'éléments dont la proximité a été établie par un premier traitement appliqué sur les ontologies dans leur globalité. Le système eTuner (Sayyadian *et al.* (2005)) adapte un alignement en recherchant de fa-

çon totalement automatique les valeurs les plus appropriées des paramètres des systèmes d'alignement qu'il met en œuvre.

Le tableau 1.1 présente quelques outils d'alignement illustrant ces différentes stratégies. Nous remarquerons l'existence de méthodes d'apprentissage automatique qui sont utilisées dans la mise en œuvre de techniques extensionnelles. De telles méthodes sont utilisées dans le système GLUE (Doan *et al.* (2004)), par exemple.

Outil	Types de techniques mises en œuvre	particularités
H-Match (Castano *et al.* (2006))	Terminologique (linguistique : thésaurus) et structurelle (interne et relationnelle)	Exécution en parallèle - 3 niveaux d'alignement : – linguistique – linguistique et structurelle (interne)- prise en compte des propriétés et de leurs contraintes – linguistique et structurelle - prise en compte des propriétés et des relations (taxonomiques).
Anchor-Prompt (Noy & Musen (2003))	Terminologique (syntaxique) puis structurelle (relationnelle)	Exécution séquentielle
Cupid (Madhavan *et al.* (2001))	Terminologique (syntaxique et linguistique : morphosyntaxique, normalisation, thesaurus) puis structurelle (relationnelle)	Exécution séquentielle
COMA & COMA++ (Do & Rahm (2007))	terminologique (syntaxique et linguistique : thesaurus)	Exécution en parallèle - COMA++ combine des outils d'alignement puis raffine automatiquement les résultats obtenus
Similarity flooding (Melnik *et al.* (2002))	Terminologique (syntaxique) puis structurelle (relationnelle)	Exécution séquentielle - Approche globale
MapOnto (An *et al.* (2005))	Sémantique (déductive)	Génère des mappings complexes
CtxMatch (Bouquet *et al.* (2003))	Terminologique (linguistique : WordNet) puis sémantique (déductive)	Exécution séquentielle
S-Match (Giunchiglia *et al.* (2004))	Terminologique (syntaxique et linguistique : WordNet) puis sémantique (déductive)	Exécution en parallèle
HCONE (Kotis *et al.* (2006))	Terminologique (linguistique : WordNet)	Outil d'alignement et de fusion d'ontologies - Implication possible de l'utilisateur pour préciser le contexte d'interprétation des termes considérés
ASCO (Le *et al.* (2004))	Terminologique (syntaxique et linguistique) puis structurelle (relationnelle)	Exécution séquentielle
OMEN (Mitra *et al.* (2005))	Terminologique (linguistique) , structurelle (relationnelle)	Méthode probabiliste basée sur les réseaux Bayésiens
FCA-merge (Stumme & Maedche (2001))	Terminologique (linguistique) et structurelle (génération de contextes)	Technique d'analyse des concepts formels exploitant les instances - Outil de fusion d'ontologies
LSD/GLUE (Doan *et al.* (2004))	Terminologique (syntaxique) et structurelle	Exécution séquentielle - Techniques d'apprentissage automatique exploitant les instances - GLUE est le successeur de LSD
NOM & QOM (Ehrig & Staab (2004))	Terminologique, structurelle (relationnelle) et extensionnelle	Exécution en parallèle
oMap (Straccia & Troncy (2005))	Terminologique, extensionnelle et sémantique	Exécution en parallèle
OLA (Euzenat & Valtchev (2004b))	Terminologique (syntaxique et linguistique : WordNet), structurelle (interne et relationnelle) et extensionnelle	Agrégation des similarités des caractéristiques des entités alignées - Approche globale
Falcon-AO (Jian *et al.* (2005))	Terminologique (linguistique), structurelle et extensionnelle	Exécution séquentielle
Taxomap (Hamdi *et al.* (2009))	Terminologique et structurelle (relationnelle)	Exécution séquentielle - Alignements résultant de l'application d'une seule technique

Tab. 1.1: Caractéristiques d'outils d'alignement existants

1.2 Découverte de mappings en contexte distribué

Nous décrivons, dans un premier temps, les différentes architectures associées aux systèmes pair-à-pair. Dans un deuxième temps, nous présentons un état de l'art des différents travaux portant sur la découverte de mappings dans ce contexte.

1.2.1 Architectures distribuées et structure d'un pair

Une architecture pair-à-pair permet à deux machines d'être l'une pour l'autre à la fois ressource et demandeur. Plus précisément, lorsqu'un pair est sollicité, il répond à partir des ressources qui lui sont propres puis sollicite, à son tour, les autres pairs auxquels il est connecté. Dans le cas des systèmes pair-à-pair sémantiques, les différents pairs sont connectés via des mappings entre leurs ontologies respectives. Nous présentons tout d'abord les différents types d'architectures distribuées possibles puis nous décrirons la structure d'un pair dans un PDMS.

Les architectures distribuées

On distingue les architectures totalement décentralisées, dans lesquelles tous les pairs sont équivalents en terme de fonctionnalités, et les architectures mixtes, dans lesquelles certains pairs ont des fonctionnalités supplémentaires.

Les architectures totalement décentralisées : Dans une architecture totalement décentralisée (figure 1.4), il n'existe que des pairs équivalents en termes de capacités et de fonctionnalités d'échange d'information. Ce type de systèmes compense l'absence de gestion globale du réseau par une multiplication des communications entre pairs.

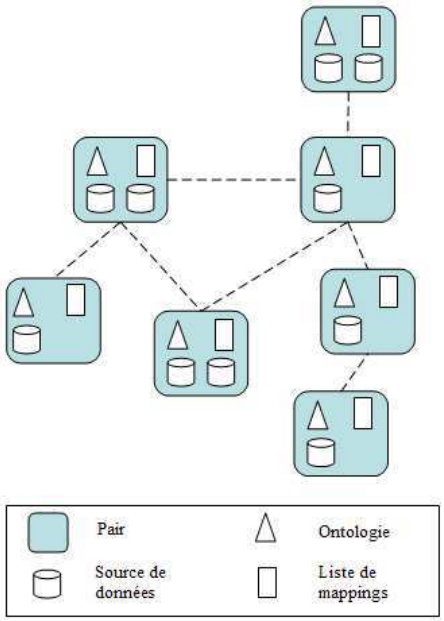

FIG. 1.4 – Exemple de système pair-à-pair totalement décentralisé

Les systèmes pair-à-pair totalement distribués se forment par des connexions successives entre les pairs. La structure du réseau peut très facilement varier : la déconnexion d'un pair peut entraîner la scission d'un réseau en deux sous réseaux déconnectés et la connexion de deux pairs peut engendrer la fusion de deux réseaux jusque là déconnectés.

Les architectures mixtes : Les architectures mixtes sont des architectures pair-à-pair dans lesquelles certains pairs, du fait de leurs qualités (puissance de calcul, bande passante élevée ou fiabilité) sont élevés au rang de super pairs et vont accomplir des tâches supplémentaires utiles à la gestion du réseau. Ce type d'architecture utilise parfois des réseaux à plusieurs couches où les super pairs jouent le rôle de serveurs pour les pairs et sont eux-mêmes organisés en réseau pair à pair.

On distingue deux catégories d'architectures mixtes.

Dans la première catégorie d'architecture mixte (cf. figure 1.5), les super pairs sont prédéfinis et existent avant le réseau. Les pairs se joignent au réseau en se connectant à un ou plusieurs super pairs. Chaque pair établit ensuite des liens avec les autres pairs connectés auprès de ce(s) super(s) pair(s).

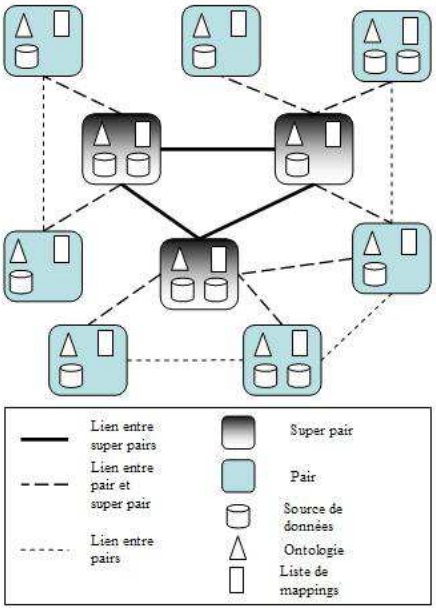

FIG. 1.5 – Exemple de système pair-à-pair avec super pairs préalablement définis

Dans ce type de systèmes, les super pairs sont très importants et leur déconnexion, bien qu'elle reste possible, est très handicapante pour le réseau car elle peut entraîner la déconnexion des pairs qui sont connectés via ce super pair.

Dans le cas des systèmes pair-à-pair sémantiques, les super pairs sont souvent associés à un domaine. Un pair se connecte alors au(x) super pair(s) associé(s) au(x) domaine(s) couvert(s) par son ontologie.

Le second type d'architecture mixte (cf. figure 1.6) est caractérisé par l'émergence des super pairs. Le réseau existe initialement sans super pairs mais des pairs répondant à certains critères (temps important passé sur le réseau, large bande passante) deviennent des super pairs.

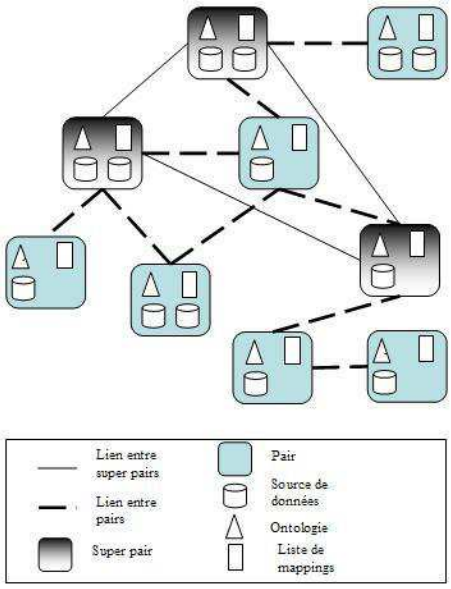

FIG. 1.6 – Exemple de système pair-à-pair avec super pairs élus

C'est ce type d'architecture qui est considéré dans Herschel & Heese (2005). Dans ces travaux, les super pairs appartiennent à deux catégories distinctes : les pairs experts et les pairs consultants. Les pairs consultants ont pour fonctionnalité supplémentaire de recenser les pairs experts. Les pairs experts sont spécialistes d'un domaine et possèdent des connaissances sur tous les pairs de ce domaine. Ils connaissent, par exemple, la finesse de description du domaine ou la qualité des informations délivrées. Dans Castano & Montanelli (2005), un super pair est le pair fondateur d'une communauté sémantique. N'importe quel pair peut fonder une communauté sémantique. Pour cela, un *fondateur de communauté* envoie un message d'invitation à l'ensemble des pairs du système contenant la description de la communauté en cours de construction. Les pairs dont l'ontologie correspond à la description de la communauté d'intérêt répondent à cette invitation. Dans une communauté sémantique, tous les membres se connaissent. Contrairement à un système avec super pairs prédéfinis, une fois la communauté fondée, la déconnexion de son fondateur n'a pas pour conséquence la disparition de la communauté.

Structure d'un pair dans un système pair-à-pair sémantique

Qu'ils soient totalement décentralisés ou mixtes, les systèmes pair-à-pair sémantiques sont constitués de pairs. Le but des systèmes pair-à-pair sémantiques est l'échange d'informations. Un pair doit pouvoir jouer, tour à tour, les rôles de source de données, de médiateur et de client interrogeant le système.

Pour remplir ces différents rôles, chaque pair possède une ou plusieurs sources de données intégrées qui peuvent être plus ou moins structurées et plus ou moins hétérogènes. Pour décrire ces données de manière homogène et les mettre à disposition de l'ensemble du système, chaque pair dispose d'une ontologie qui décrit le contenu des sources de données. Il possède aussi un ensemble de mappings établissant des relations entre les éléments des ontologies des pairs du réseau et lui permettant de communiquer avec eux. Ainsi, étant donnée une requête posée par un utilisateur, un pair pourra répondre soit directement par interrogation de ses propres sources de données, soit indirectement, en utilisant ses mappings pour propager l'interrogation à d'autres pairs.

Dans Castano *et al.* (2003a), une structure alternative permettant cette gestion est proposée. La connaissance de chaque pair est représentée par un schéma à deux niveaux (figure 1.7). Le premier niveau représente une ontologie dont les concepts, décrits par un nom et des propriétés, sont liés par des relations sémantiques. Il existe cinq types de relations sémantiques : *same as*, *kind-of*, *part-of*, *contains* et *associates*. *Same as* est la relation d'équivalence, *kind-of* est la relation d'hyperonymie, *part-of* est la relation de méronymie, *contains* traduit une relation proche de l'holonymie et *associates* représente toutes les relations qui ne rentrent pas dans les autres cadres. Le second niveau représente l'ensemble des pairs connus caractérisés par des propriétés (par exemple l'adresse IP). Les éléments des deux niveaux sont liés par des relations de localisation appelées *location relation*.

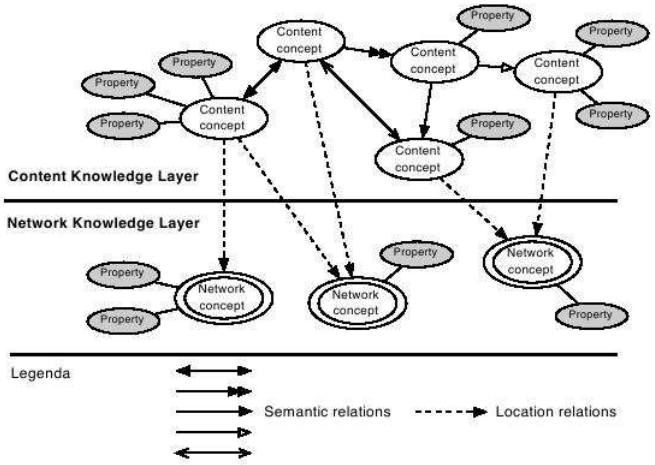

FIG. 1.7 – Architecture d'un pair dans Helios (Castano *et al.* (2003a))

Illustrons cette architecture par l'exemple de la figure 1.8 : l'ontologie du pair P1 représentée sur cette figure est composée des concepts *publication* et *chapter* définis par le pair P1, du concept *book* et des propriétés *author* et *title* définis par le pair P2 et intégrés par P1. Les concepts sont liés entre eux par des relations sémantiques (*book kind-of publication*, *chapter part-of publication*. Les concepts intégrés sont liés à un concept identifiant le pair (ici Pair #2) les ayant définis et par la relation spécifique *location-relation* : *book location-relation Pair #2*. Les informations concernant le pair P2 sont stockées sous la

forme de propriétés ("IP#").

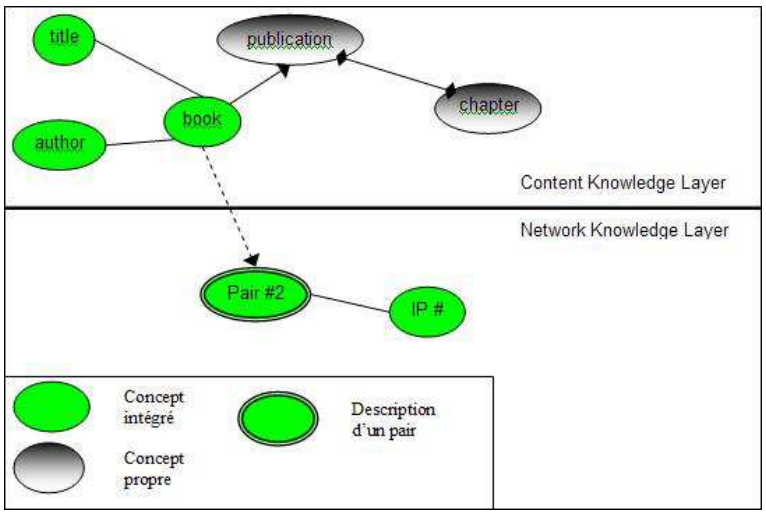

FIG. 1.8 – Exemple de pair dans Helios

1.2.2 Travaux portant sur la découverte de mappings en contexte distribué

La découverte de mappings consiste à trouver les relations sémantiques existant entre des concepts issus d'ontologies différentes. Dans le contexte d'un PDMS, ce problème est particulier du fait de la spécificité du PDMS. Nous nous limiterons, dans cette partie, à présenter les travaux apportant des solutions à la découverte de mappings se situant spécifiquement dans un contexte pair-à-pair. Notre présentation distinguera les travaux réutilisant des approches conçues en contexte centralisé, les travaux pour lesquels les ontologies ne sont pas directement mises en relation, les travaux exploitant des requêtes, et enfin ceux dont les mappings découverts sont générés automatiquement par des mécanismes de raisonnement.

Travaux réutilisant des approches éprouvées en contexte centralisé

Certains travaux proposent l'application de techniques éprouvées testées dans le cadre de systèmes d'intégration. Ainsi, dans Piazza Halevy *et al.* (2005), des outils et des techniques ont été développées pour faciliter et assister la création de mappings. Ces travaux ont plus particulièrement mis l'accent sur l'aspect automatique des techniques d'alignement à utiliser. La découverte de mappings se fait ainsi de façon totalement automatique (Tatarinov *et al.* (2003)). L'approche retenue est une composition de techniques d'apprentissage automatique variées basées sur les éléments des schémas à rapprocher et sur les données associées et de techniques consistant à déduire de nouveaux mappings à partir d'autres sur le domaine, déjà connus et validés. Ces travaux supposent que les ontologies de tous les pairs sont connues de tous, ou du moins accessibles dans leur totalité par tous,

ce qui, dans le contexte des systèmes P2P, est une hypothèse forte.

Ces travaux s'appuient sur les techniques implémentées dans le système GLUE (Doan *et al.* (2004)) pour l'intégration de sources de données hétérogènes. Dans ce système, deux concepts sont dits similaires si leurs ensembles d'instances se recouvrent. Soient deux concepts A et B appartenant respectivement à deux ontologies *O1* et *O2*. Les instances de A sont tout d'abord utilisées comme ensemble d'apprentissage afin de construire le classifieur d'instances de A. Ce classifieur est ensuite utilisé pour classifier les instances de B, c'est-à-dire décider si une instance de B est aussi une instance de A. L'approche utilise un modèle probabiliste.

Travaux pour lesquels les ontologies ne sont pas directement mises en relation

Travaux basés sur l'existence d'un service externe d'alignement : Afin de ne pas utiliser les ressources d'un pair donné, ni du système pair-à-pair auquel il appartient, il est possible de faire appel à un service externe d'alignement. Le serveur d'alignement est, dans ce cas, considéré comme ne faisant pas partie du réseau pair-à-pair en tant que tel.

Un tel service d'alignement peut identifier des correspondances à la demande, à l'aide d'algorithmes d'alignement d'ontologies. Il peut demander au pair à l'origine de la demande d'alignement de valider certaines opérations exécutées pour l'alignement. Il peut aussi utiliser des alignements déjà construits, qu'il aura stockés dans sa base de données. En particulier, si un pair $\mathcal{P}$ utilise une ontologie renommée trouvée sur le Web et qu'il en est de même pour d'autres pairs auxquels $\mathcal{P}$ s'est connecté, il est probable que l'alignement soit déjà calculé. Les alignements d'un tel service restent disponibles malgré l'évolution de la topologie du réseau pair-à-pair. On pourra donc toujours composer des alignements, même si un pair s'avère manquant. Jerome Euzenat dans Euzenat (2005) et dans Euzenat *et al.* (2008) propose un tel service ayant, entre autres, ces fonctionnalités. Ce service permet de construire un réseau d'ontologies alignées à partir d'ontologies isolées.

Travaux exploitant des ontologies de référence : D'autres travaux font intervenir des ontologies dites de référence avec lesquelles l'ontologie de chaque pair peut être liée, ce qui évite aussi la mise en relation directe des ontologies des pairs les unes avec les autres.

Ainsi, dans Herschel & Heese (2005) et Heese *et al.* (2005), la découverte de nouveaux mappings repose sur la consultation de super pairs spécialistes d'un domaine. Chaque pair décrit ses données à l'aide d'un schéma qui lui est propre et les consulte à l'aide d'un langage de requêtes. On suppose qu'il existe des ontologies de référence sur le réseau. Chaque pair établit des mappings entre son schéma et cet ensemble d'ontologies partagées. Aux entités des bases de données de chaque pair sont donc associées les descriptions des concepts trouvés dans les ontologies de référence. L'ensemble de tous ces concepts et leurs relations dans les ontologies forment un graphe propre à chaque pair appelé *peer graph*.

Lorsqu'un pair se connecte au PDMS, il utilise une fonction de hachage sur les URI des ontologies d'origine des concepts de son *peer graph*. Par cette fonction de hachage, il obtient les adresses des pairs experts auxquels il est associé. Une requête exprimée dans le langage de requêtes d'un pair est transformée en un *query graph*, à l'aide des mappings schéma-ontologies établis. Une fonction de hachage est appliquée sur les URI des ontologies auxquelles il est fait référence dans le *query graph*. On obtient ainsi les adresses des pairs experts à consulter. Ces derniers disposent de la liste des pairs à consulter pour répondre. Grâce à ces informations, le pair initial établit le plan de requête à exécuter.

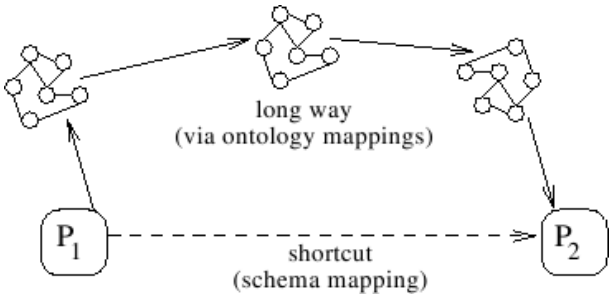

FIG. 1.9 – Utilisation des mappings schéma-ontologies pour trouver des mappings de schémas

Les plans de requête ainsi obtenus permettent d'établir des *mappings de schémas* ou raccourcis (cf. figure 1.9) entre les entités du schéma d'un pair et les entités des schémas d'autres pairs correspondant aux mêmes concepts. Ces mappings ne correspondent pas à des raccourcis de mappings au sens indiqué dans la section 3.1 mais permettent d'effectuer des raccourcis méthodologiques pour moins solliciter les super pairs : en effet, les pairs consultent d'abord leurs données locales, puis celles des pairs déjà connus. Ce n'est que lorsqu'aucune réponse n'est obtenue en local qu'une nouvelle source de données sera recherchée par utilisation des super pairs.

Travaux basés sur l'exploitation de requêtes

Travaux analysant les requêtes préalablement posées : Dans Limam *et al.* (2008), une approche à base de requêtes pour la découverte de mappings entre bases de données en pair-à-pair a été proposée. L'idée sous-jacente est que, même si des schémas de bases de données sont différents d'un point de vue structurel et syntaxique, les requêtes posées sur ces bases peuvent être similaires car elles retournent des objets sémantiquement liés. L'approche de découverte de mappings est donc de raisonner sur les requêtes posées sur différentes bases. Etant données une requête Q_i^1 sur un schéma S_1 et une requête Q_j^2 sur un schéma S_2, le raisonnement réalisé consiste à analyser les relations existant entre les attributs apparaissant dans Q_i^1 et Q_j^2 et à propager ces relations sur les autres attributs des deux schémas.

L'originalité de ce travail est d'utiliser l'analyse sémantique latente (LSA pour Latent Semantic Analysis) pour raisonner sur les requêtes posées. La LSA est un procédé de

traitement des langues naturelles qui établit de manière totalement automatique des relations entre un ensemble de documents et les termes qu'ils contiennent. Elle s'appuie sur la co-occurrence d'éléments, ici la co-occurrence d'attributs dans des requêtes, pour inférer des relations par comparaison de matrices des occurrences qui, dans ce travail, correspondent à des relations entre éléments de schémas.

Travaux basés sur l'interrogation des pairs du réseau : Dans les travaux de Castano *et al.* (2003a), la découverte de mappings est basée sur l'interrogation des pairs du réseau. Ceci signifie qu'un pair P1 qui souhaite élargir ses connaissances sur un concept C donné va rechercher les pairs dont l'ontologie contient des concepts ayant un lien sémantique avec C en posant une requête de type *probe query* sur le réseau. La requête posée contient le concept C. Les pairs du réseau peuvent répondre si leur ontologie contient un concept qui peut être mis en relation avec le concept C de la requête. Ce mécanisme est illustré en figure 1.10 : le pair A souhaite enrichir le concept *book*. Il pose alors une *probe query* contenant la description de ce concept. Cette requête est transmise à tous les pairs du réseau, notamment les pairs B et C. Chacun des pairs va alors comparer la description du concept *book* avec tous les concepts qui composent son ontologie. Le pair C identifie un concept sémantiquement proche tandis que le pair B en identifie deux. Chacun des pairs retourne ensuite au pair A la description de tous les concepts qu'il a identifiés comme étant sémantiquement proches du concept *book*.

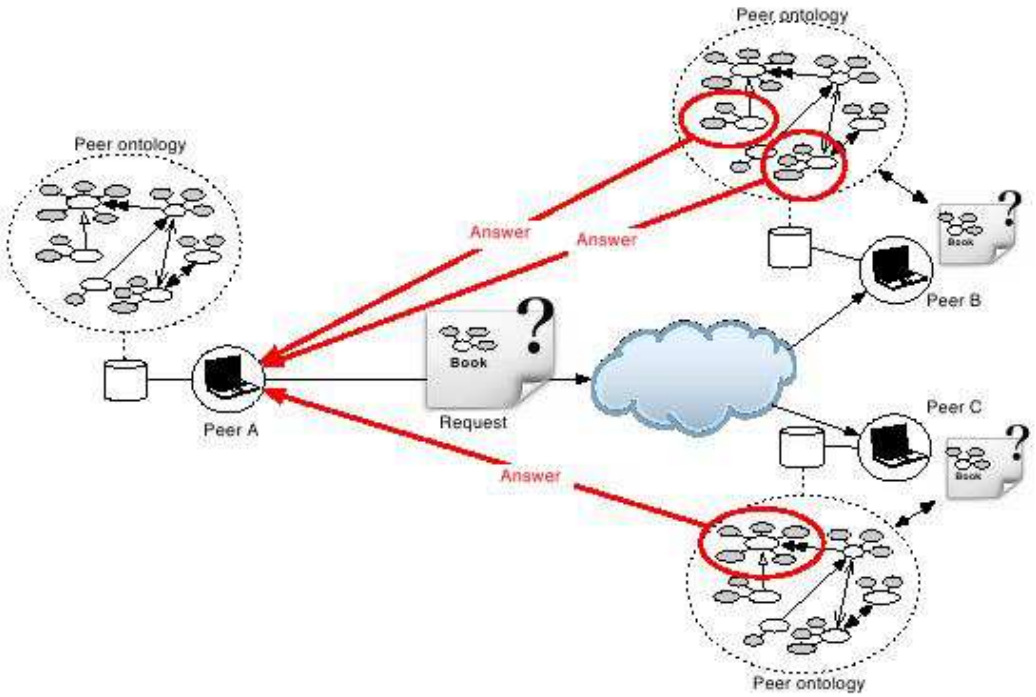

FIG. 1.10 – Exemple de *probe query* sur le concept *book*

Cela signifie que les pairs savent déterminer s'il existe une relation entre un concept d'une requête et un de leurs concepts. Cela est réalisé par la mise en œuvre de l'algorithme H-Match capable de déterminer le niveau d'affinité entre deux concepts. H-Match (Castano *et al.* (2006)) est une version étendue et enrichie des techniques développées dans le cadre du système d'intégration de sources de données hétérogènes ARTEMIS, l'objectif étant de permettre des mises en correspondance entre ontologies. Dans H-Match, chaque concept est associé à un contexte (cf. figure 1.11) donné par l'ensemble de ses propriétés et l'ensemble de ses concepts adjacents (concepts liés par une relation sémantique). Les comparaisons s'effectuent ainsi entre contextes (cf. figure 1.12).

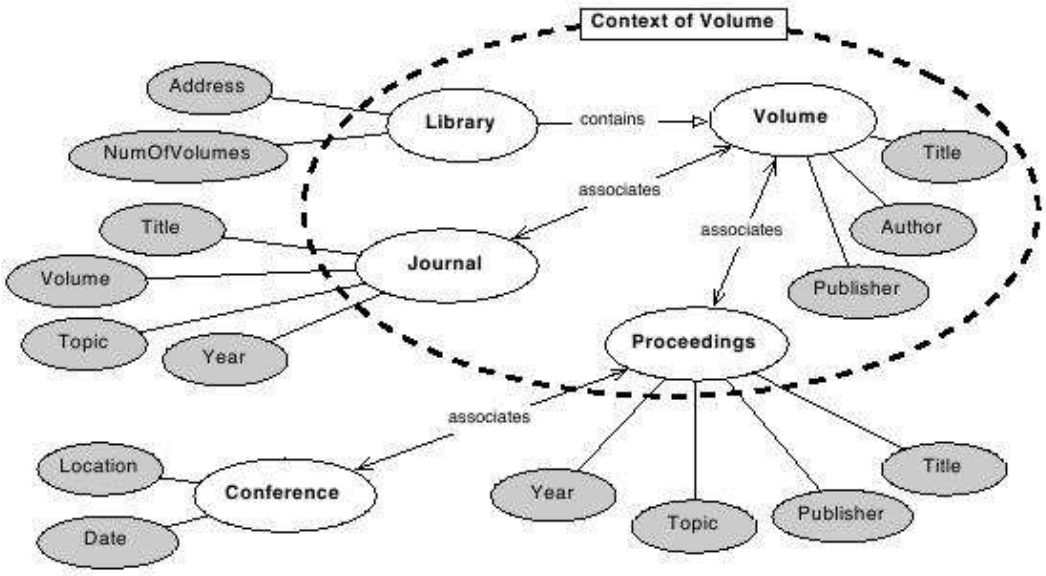

FIG. 1.11 – Exemple de contexte pour le concept "Volume" Castano *et al.* (2003a)

L'évaluation de l'affinité entre deux concepts repose sur l'évaluation de l'affinité linguistique des noms de propriétés et de concepts des différents contextes (recherche de relations terminologiques représentées dans un thesaurus) et sur l'évaluation du degré de proximité des relations sémantiques représentées dans les différents contextes (évaluation contextuelle).

Les mises en correspondance trouvées sont ainsi accompagnées d'une mesure de l'affinité sémantique existant entre les concepts. Deux stratégies de matching, plus ou moins restrictives, sont possibles : la première est basée sur la recherche de concepts similaires (mesure supérieure à un seuil de similarité) tandis que la seconde exploite la recherche de concepts équivalents (seuil d'équivalence).

L'intérêt de H-Match dans un environnement distribué repose sur sa flexibilité et son côté dynamique. En effet, le modèle de matching appliqué est choisi en fonction de la requête. De plus, l'algorithme est paramétrable et permet de donner un poids plus ou moins fort aux aspects linguistiques et contextuels et d'ignorer ou de limiter la notion de contexte associé aux concepts. Ainsi, dans le cadre d'Helios, trois modèles de matching

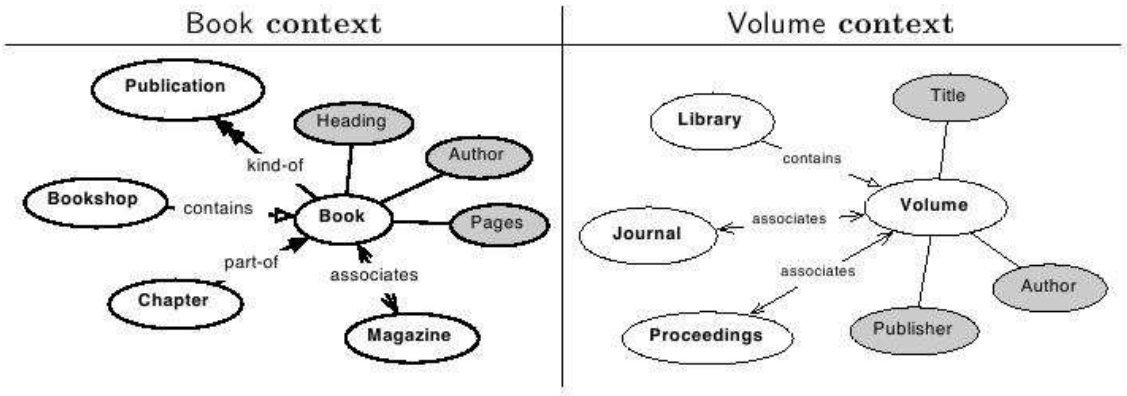

FIG. 1.12 – Exemple de comparaison des contextes des concepts *book* et *volume*

sont proposés :

Shallow : seules les informations linguistiques fournies par les noms des concepts et le thésaurus sont considérées. Le contexte est totalement ignoré. Ce modèle est appliqué lorsque la requête ne contient que des noms de concepts. Peu de calculs sont nécessaires mais les noms doivent être signifiants et précis pour obtenir de bons résultats.

Intermediate : les informations linguistiques sont étendues aux propriétés des concepts. On utilise aussi les informations contextuelles limitées à ces mêmes propriétés. Ces informations supplémentaires sont exploitées pour fournir une pertinence supérieure.

Deep : tout le contexte des concepts est pris en compte. Ce modèle exige une description complète du concept dans la requête. Il nécessite également plus de calculs mais il fournit des résultats d'une grande précision. Il est donc recommandé lorsque l'on s'intéresse plus à la pertinence qu'au temps de réponse.

L'applicabilité de l'un ou l'autre de ces modèles dépend de la richesse de la description du concept dans la requête soumise. Le pair ayant posé la requête intègre ensuite les réponses reçues (concepts proches du concept cible décrits par leurs propriétés et leurs relations avec d'autres concepts) à son ontologie grâce au même algorithme H-Match, et enregistre leur localisation (pair à l'origine du concept). Ainsi, pour découvrir les mappings de façon automatique, une solution consiste à interroger tous les pairs du réseau comme nous venons de l'illustrer précédemment.

D'autres architectures permettent de limiter le nombre de pairs interrogés. En s'appuyant sur le fait qu'une ontologie porte sur un domaine donné, il est proposé, dans Castano *et al.* (2003b), de n'envoyer les *probe queries* qu'aux pairs dont les ontologies recouvrent le domaine de la requête. Pour permettre cela, les pairs sont rassemblés en groupes d'intérêts. Un service permet à un pair de se déclarer membre d'une communauté d'intérêt et de connaître ainsi les autres pairs de cette communauté. Seules les *probe queries* n'ayant pas eu de réponses auprès de ce groupe de pairs sont envoyées à l'ensemble du réseau.

Travaux basés sur la mise en œuvre de mécanismes de raisonnement

Les techniques d'alignement sémantiques permettent, à l'aide d'une représentation formelle des concepts des ontologies et d'un moteur d'inférence, de générer automatiquement des correspondances sémantiques entre concepts de deux ontologies. Etendre et adapter l'application de ces techniques aux systèmes pair-à-pair est une solution intéressante. Ainsi dans Médini *et al.* (2007), on trouve une approche de découverte de mappings à base d'inférences en 3 étapes, utilisant des méthodes sémantiques fondées sur la logique de description. Prenons l'exemple d'un pair A souhaitant connaître ses correspondances avec un pair C et possédant un voisin B avec lequel des mappings sont déjà établis. Le processus de découverte de mappings entre A et C est le suivant : (1) Génération des axiomes : identification des correspondances existantes entre A et C en demandant éventuellement à C (absence d'hypothèse de symétrie des correspondances), identification des correspondances existantes entre A et B par un mécanisme analogue, (2) Génération des ontologies : il s'agit de définir les parties des ontologies de A et de C sur lesquelles le raisonnement sera réalisé. Ceci se fait par segmentation, (3) Découverte de correspondances : A met en œuvre la recherche de correspondances sémantiques à l'aide de son moteur d'inférences, en utilisant les axiomes identifiés et sur les parties des ontologies de A et de C concernées.

Dans ce travail, les auteurs ont eu pour objectif de minimiser la quantité d'informations fournie au moteur d'inférences lors d'une recherche de correspondance, afin de diminuer le temps de calcul. Ceci justifie le processus de segmentation permettant de réduire la taille des ontologies d'entrée, en ne communiquant pas les parties des ontologies dans lesquelles aucune correspondance ne peut être découverte. Par ailleurs, ce travail avait aussi pour ambition de distribuer les opérations à réaliser entre les différents pairs du réseau.

D'autres travaux se sont intéressés à la découverte de mappings entre les données des pairs d'un réseau et ont également proposé des solutions reposant sur l'exécution de mécanismes de raisonnement automatisés. C'est le cas du projet Hyperion (http://www.cs.toronto.edu/db/hyperion/) dont les objectifs principaux étaient : la définition d'une architecture de gestion de données en pair-à-pair, l'étude du problème d'intégration de données, le développement d'algorithmes permettant une recherche et un échange de données entre les pairs efficaces. Dans ce cadre, des outils de gestion des données facilitant la création, la maintenance et la gestion de tables de mappings ont été développés. En effet, les tables de mappings, établissant des correspondances entre des valeurs, sont considérées comme des outils bien appropriés aux systèmes pair-à-pair car ils respectent leur principe d'autonomie et permettent de lier des données d'un même domaine ou de domaines différents.

Les travaux décrits dans Kementsietsidis *et al.* (2003) proposent différentes sémantiques pour les tables de mappings, un langage de spécification formelle et des techniques automatiques de raisonnement capables d'inférer de nouveaux mappings. L'approche suivie consiste à considérer les tables de mappings comme des contraintes sur l'échange de données entre pairs. L'algorithme de raisonnement proposé pour inférer de nouveaux mappings permet d'éliminer les mappings inconsistants, c'est-à-dire ne satisfaisant pas

les contraintes selon la sémantique associée. Cet algorithme est distribué ; le processus de calcul est réparti entre différents pairs.

1.3 Positionnement de notre travail

Les travaux présentés dans cette thèse font suite à la thèse de Philippe Adjiman (Adjiman (2006)) dans laquelle SomeRDFS, un système pair-à-pair de gestion de données ou Peer to Peer Data Management System (PDMS) a été conçu. SomeRDFS exploite des mappings sémantiques définis entre les ontologies des différents pairs. Notre objectif est d'enrichir les mappings exploités par SomeRDFS.

Dans un premier temps, nous présentons SomeRDFS qui constitue notre contexte d'étude. Nous présenterons, ensuite, les particularités de notre travail par rapport aux autres travaux du domaine.

1.3.1 Description du contexte d'étude

Nous présentons dans un premier temps le modèle des données de SomeRDFS puis nous expliquerons le raisonnement mis en œuvre au sein de cette plateforme pour répondre aux requêtes.

Modèle des données de SomeRDFS

Dans le contexte de SomeRDFS, les ontologies et les mappings sont exprimés en RDF(S) tandis que les données sont représentées en RDF. Il est ainsi possible de définir des classes, des sous-classes, des propriétés, des sous-propriétés, de typer le domaine et la portée des propriétés. Les constructeurs autorisés sont l'inclusion de classes, l'inclusion de propriétés, le typage de domaine et de portée. Ce langage, restreint à ces constructeurs, appelé core-RDFS, a l'avantage d'avoir une sémantique logique claire et intuitive. Il est basé sur des relations unaires qui représentent les classes et des relations binaires qui représentent des propriétés. Nous donnons dans le tableau 1.2 la sémantique logique de RDF(S) en indiquant la notation en logique de description associée (notation LD) et la traduction en logique du 1er ordre des constructeurs correspondants.

Opérateur	Notation LD	Traduction en logique du premier ordre
Inclusion de classes	$C_1 \sqsubseteq C_2$	$\forall X, C_1(X) \Rightarrow C_2(X)$
Inclusion de propriétés	$P_1 \sqsubseteq P_2$	$\forall X \forall Y R_1(X,Y) \Rightarrow R_2(X,Y)$
Typage de domaine d'une propriété	$\exists P \sqsubseteq C$	$\forall X \forall Y, P(X,Y) \Rightarrow C(X)$
Typage de co-domaine d'une propriété	$\exists P^- \sqsubseteq C$	$\forall X \forall Y, P(X,Y) \Rightarrow C(Y)$

TAB. 1.2 – Opérateurs RDF(S)

Les ontologies des pairs sont représentées à l'aide d'expressions core-RDFS composées uniquement d'éléments appartenant au vocabulaire du pair. Le vocabulaire d'un pair comprend l'union de l'ensemble des noms de classes et de l'ensemble des noms de propriétés. Les noms de classes et de propriétés sont propres à un pair donné. Nous notons $\mathcal{P}$:e l'élément e (classe ou propriété) de l'ontologie du pair $\mathcal{P}$.

Les données des pairs sont associées à des éléments faisant partie de son vocabulaire. Ainsi, les notations en logique terminologique et en logique du 1er ordre correspondant respectivement à la déclaration d'une instance d'une classe et d'une instance d'une propriété sont $C(a)$ et $P(a,b)$, a et b étant des constantes.

Un mapping correspond à une inclusion entre classes ou propriétés appartenant aux ontologies de 2 pairs différents (cf. tableau 1.3 a et b) ou au typage d'une propriété appartenant à l'ontologie d'un pair donné avec une classe appartenant à l'ontologie d'un autre pair (cf. TAB. 1.3 c et d). Ainsi les mappings sont également des expressions RDF(S). Leur spécificité est d'être construites à partir du vocabulaire de pairs différents qui établissent ainsi des correspondances sémantiques entre eux.

Mappings	Notation LD	Traduction en logique du premier ordre
a. Inclusion de classes	$\mathcal{P}_1{:}C_1 \sqsubseteq \mathcal{P}_2{:}C_2$	$\forall X, \mathcal{P}_1{:}C_1(X) \Rightarrow \mathcal{P}_2{:}C_2(X)$
b. Inclusion de propriétés	$\mathcal{P}_1{:}P_1 \sqsubseteq \mathcal{P}_2{:}P_2$	$\forall X \forall Y, \mathcal{P}_1{:}P_1(X,Y) \Rightarrow \mathcal{P}_2{:}P_2(X,Y)$
c. Typage de domaine d'une propriété	$\exists \mathcal{P}_1{:}P \sqsubseteq \mathcal{P}_2{:}C$	$\forall X \forall Y, \mathcal{P}_1{:}P_1(X,Y) \Rightarrow \mathcal{P}_2{:}C(X)$
d. Typage de co-domaine d'une propriété	$\exists \mathcal{P}_1{:}P^- \sqsubseteq \mathcal{P}_2{:}C$	$\forall X \forall Y, \mathcal{P}_1{:}P_1(X,Y) \Rightarrow \mathcal{P}_2{:}C(Y)$

TAB. 1.3 – Mappings

Raisonnement mis en œuvre dans SomeRDFS

L'ensemble des connaissances d'un pair est traduit en clauses propositionnelles de façon à réduire le problème du calcul des réponses à des requêtes conjonctives à un calcul de conséquences implémenté dans SomeWhere via l'algorithme DEcentralized Consequence finding Algorithm (Adjiman *et al.* (2004)). Nous décrivons, dans cette section, ce processus de calcul des réponses à une requête sur un exemple.

Soient deux pairs, $\mathcal{P}_1$ et $\mathcal{P}_2$, pour lesquels les ontologies (cf. tableau 1.4), les mappings (cf. tableau 1.5) et les données (cf. tableau 1.6) sont définis.

$\mathcal{P}_1$	$\mathcal{P}_2$
$\forall X, \mathcal{P}_1{:}Painter(X) \Rightarrow \mathcal{P}_1{:}Artist(X)$	$\forall X \forall Y, \mathcal{P}_2{:}Sculpts(X,Y) \Rightarrow \mathcal{P}_2{:}Sculptor(X)$
$\forall X, \mathcal{P}_1{:}Painting(X) \Rightarrow \mathcal{P}_1{:}Artifact(X)$	$\forall X \forall Y, \mathcal{P}_2{:}Sculpts(X,Y) \Rightarrow \mathcal{P}_2{:}Sculpture(Y)$
$\forall X \forall Y, \mathcal{P}_1{:}Paints(X,Y) \Rightarrow \mathcal{P}_1{:}Creates(X,Y)$	$\forall X \forall Y, \mathcal{P}_2{:}Refersto(X,Y) \Rightarrow \mathcal{P}_2{:}Sculptor(X)$
$\forall X \forall Y, \mathcal{P}_1{:}Paints(X,Y) \Rightarrow \mathcal{P}_1{:}Painter(X)$	$\forall X, \mathcal{P}_2{:}Refersto(X,Y) \Rightarrow \mathcal{P}_2{:}Movement(Y)$
$\forall X \forall Y, \mathcal{P}_1{:}Paints(X,Y) \Rightarrow \mathcal{P}_1{:}Painting(Y)$	$\forall X, \mathcal{P}_2{:}SteelSculptor(X) \Rightarrow \mathcal{P}_2{:}Sculptor(X)$
$\forall X \forall Y, \mathcal{P}_1{:}Creates(X,Y) \Rightarrow \mathcal{P}_1{:}Artist(X)$	$\forall X, \mathcal{P}_2{:}GlassSculptor(X) \Rightarrow \mathcal{P}_2{:}Sculptor(X)$
$\forall X \forall Y, \mathcal{P}_1{:}Creates(X,Y) \Rightarrow \mathcal{P}_1{:}Artifact(Y)$	$\forall X, \mathcal{P}_2{:}WoodSculptor(X) \Rightarrow \mathcal{P}_2{:}Sculptor(X)$
$\forall X \forall Y, \mathcal{P}_1{:}Belongsto(X,Y) \Rightarrow \mathcal{P}_1{:}Artifact(X)$	

TAB. 1.4 – Ontologies de $\mathcal{P}_1$ et $\mathcal{P}_2$

$\mathcal{P}_1$	$\mathcal{P}_2$
$\forall X \forall Y, \mathcal{P}_1{:}Belongsto(X,Y) \Rightarrow \mathcal{P}_2{:}Movement(Y)$	$\forall X, \mathcal{P}_2{:}Sculptor(X) \Rightarrow \mathcal{P}_1{:}Artist(X)$
$\forall X, \mathcal{P}_2{:}Sculpture(X) \Rightarrow \mathcal{P}_1{:}Artifact(X)$	

TAB. 1.5 – Mappings de $\mathcal{P}_1$ et $\mathcal{P}_2$

Chacun des deux pairs $\mathcal{P}_1$ et $\mathcal{P}_2$ peut être sollicité à l'aide de requêtes exprimées avec son propre vocabulaire. Ainsi, si un utilisateur s'adresse au pair $\mathcal{P}_1$ pour connaître l'ensemble des artefacts connus du système, la requête suivante sera posée : $Q_1(X) \equiv$

$\mathcal{P}_1$	$\mathcal{P}_2$
$\mathcal{P}_1$:$Painter(Monet)$	$\mathcal{P}_2$:$Sculpts(Rodin, Le\ Penseur)$
$\mathcal{P}_1$:$Paints(Picasso, Les\ demoiselles\ d'Avignon)$	$\mathcal{P}_2$:$Sculptor(Cesar)$
$\mathcal{P}_1$:$Painting(La\ Joconde)$	$\mathcal{P}_2$:$Sculpture(statue\ de\ David)$
$\mathcal{P}_1$:$Refersto(Les\ demoiselles\ d'Avignon, Cubisme)$	$\mathcal{P}_2$:$Refersto(deVinci, Renaissance)$

TAB. 1.6 – Données de $\mathcal{P}_1$ et $\mathcal{P}_2$

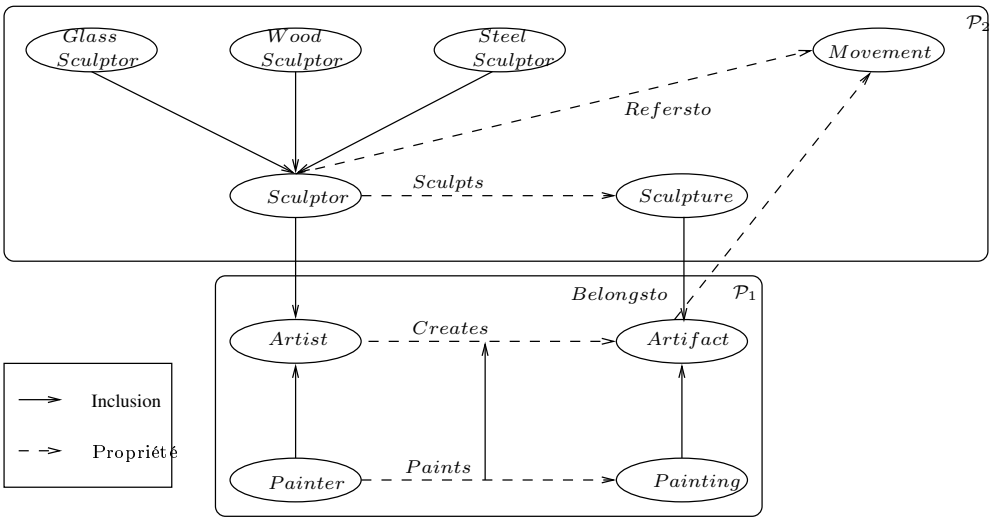

FIG. 1.13 – Représentation graphique des ontologies et des mappings des pairs $\mathcal{P}_1$ et $\mathcal{P}_2$ (PDMS $\mathcal{S}$)

$\mathcal{P}_1$:$Artifact(X)$. Le calcul des réponses à Q_1 se fait en deux temps. Q_1 est d'abord réécrite en un ensemble de requêtes la subsumant.

Le calcul des réécritures maximales de chaque atome de la requête fournit un ensemble de conjonctions de relations (classes ou propriétés). Une réécriture possible de $Q_1(X) \equiv \mathcal{P}_1$:$Artifact(X)$ est, par exemple, $\mathcal{P}_4$:$Painting(X)$, ce qui signifie qu'un moyen d'obtenir des instances de $Artifact(X)$ consiste à aller rechercher les instances de $Painting(X)$ au sein du même pair. Une autre réécriture est $\mathcal{P}_2$:$Sculpture(X)$, ce qui signifie qu'un autre moyen d'obtenir des instances de $Artifact(X)$ consiste à retenir les instances de $Sculpture(X)$ dans $\mathcal{P}_2$. Sur notre exemple, les réécritures obtenues sont :

- $RW_1(X) \equiv \mathcal{P}_1$:$Artifact(X)$
- $RW_2(X) \equiv \mathcal{P}_1$:$Painting(X)$
- $RW_3(X) \equiv \exists Y, \mathcal{P}_1$:$Belongsto(X, Y)$
- $RW_4(X) \equiv \exists Y, \mathcal{P}_1$:$Paints(Y, X)$
- $RW_5(X) \equiv \exists Y, \mathcal{P}_1$:$Creates(Y, X)$
- $RW_6(X) \equiv \exists Y, \mathcal{P}_2$:$Sculpts(Y, X)$
- $RW_7(X) \equiv \mathcal{P}_2$:$Sculpture(X)$

La réécriture RW_2 est obtenue par $\mathcal{P}_1$ en utilisant l'inclusion de classes, tandis que R_4 et R_5 sont obtenues en utilisant le typage de portée des propriétés $\mathcal{P}_1$:$Paints$ et $\mathcal{P}_1$:$Creates$. Les mappings $\forall X\ \forall Y, \mathcal{P}_1$:$Belongsto(X, Y) \Rightarrow \mathcal{P}_2$:$Movement(Y)$ et $\forall X$, $\mathcal{P}_2$:$Sculpture(X) \Rightarrow \mathcal{P}_1$:$Artifacts(X)$ permettent d'obtenir les réécritures RW_3 et RW_7. La requête $Q_1(X)$ est ensuite propagée au pair $\mathcal{P}_2$ du fait de l'existence de mappings avec des relations de $\mathcal{P}_2$. Ce dernier transmettra à $\mathcal{P}_1$ les réécritures qu'il obtient, i.e. RW_6.

Ces réécritures sont ensuite évaluées afin d'obtenir les données associées. La réponse à Q_1 est alors :

$$Q_1(\mathcal{S}) = \underbrace{\emptyset}_{RW_1(\mathcal{S})} \bigcup \underbrace{\{LaJoconde\}}_{RW_2(\mathcal{S})} \bigcup \underbrace{\emptyset}_{RW_3(\mathcal{S})} \bigcup \underbrace{\{lesdemoisellesd'Avignon\}}_{RW_4(\mathcal{S})} \bigcup \underbrace{\emptyset}_{RW_5(\mathcal{S})} \bigcup \underbrace{\{lePenseur\}}_{RW_6(\mathcal{S})} \bigcup \underbrace{\{statuedeDavid\}}_{RW_7(\mathcal{S})}$$

1.3.2 Situation de notre travail par rapport à l'existant

Notre travail a été réalisé dans le contexte de SomeRDFS et les choix qui ont été réalisés en dépendent complètement. Nous retiendrons trois caractéristiques importantes pour notre travail à propos du PDMS SomeRDFS.

La première caractéristique concerne l'architecture associée qui est une architecture totalement décentralisée composée de pairs totalement équivalents en termes de capacités et de fonctionnalités d'échange d'information.

La seconde caractéristique concerne le modèle de données de SomeRDFS. Les ontologies et les mappings sont exprimés en RDF(S), les données sont représentées en RDF. RDF(S) est un langage de représentation des connaissances fournissant les éléments de base pour la définition d'ontologies ou de vocabulaires destinés à structurer des ressources RDF. Son expressivité est toutefois limitée, ce qui explique l'existence du langage ontologique OWL, intégrant les composants principaux de RDF(S) et plus expressif. Le modèle RDF(S) contraint les connaissances ontologiques représentées sur lesquelles les techniques d'alignement peuvent prendre appui. Celles-ci se limitent à la représentation de classes, de propriétés, d'inclusions de classes et de propriétés, du typage de domaine et de co-domaine des propriétés.

La troisième caractéristique porte sur les mécanismes de raisonnement associés au système. Le raisonnement associé à un PDMS est une source de richesse pour découvrir des mappings entre ontologies. L'analyse des réponses obtenues à une requête posée à un pair peut, par exemple, permettre de s'apercevoir de l'absence de participation de certains pairs et donc de connaître les classes (concepts) et propriétés pour lesquelles il serait intéressant de rechercher des mappings si on souhaite des réponses plus larges. Ces mécanismes de raisonnement ont été conçus par Philippe Adjiman dans sa thèse (Adjiman (2006)). Nous les avons implémentés, ce qui a permis de les exploiter pour la découverte de mappings entre ontologies.

Ces caractéristiques sont importantes. Elles ont un certain nombre de conséquences qui ont contribué à définir la particularité du problème de découverte de mappings auquel nous nous attaquions. Nous distinguons deux types de conséquences, celles inhérentes à tout système distribué et celles spécifiques aux systèmes distribués dont l'architecture est totalement décentralisée.

Dans tout système distribué, le nombre de pairs composant le réseau est inconnu et

potentiellement très important. Ainsi, le nombre d'ontologies pour lesquelles nous souhaitons étudier l'existence de mises en correspondance est lui aussi important, de même que le nombre d'éléments de ces diverses ontologies potentiellement alignables. Ceci a une influence sur la mise en oeuvre de l'étape 2 du processus d'alignement (cf. figure 1.2) correspondant à la sélection des couples à comparer. Dans notre contexte, cette étape ne peut correspondre à une sélection de couples parmi un ensemble de couples possibles connu a priori. La prise en considération de ce problème a été primordiale dans cette thèse. Etant donné un pair, l'ensemble des couples d'éléments entre lesquels l'alignement est étudié ne peut être que *construit* en sélectionnant, d'une part les éléments de ce pair pour lesquels il est *intéressant* de rechercher des mappings, d'autre part les éléments des pairs distants qu'il est *pertinent* d'aligner avec ceux du pair local préalablement retenus. Un gros travail a donc consisté à définir les *bons* critères de sélection.

L'aspect dynamique est également un point singulier des systèmes distribués. Un pair peut, à un moment donné, décider de se déconnecter du réseau. A tout moment, de nouveaux pairs peuvent rejoindre le réseau. Dans cette thèse, nous avons également tenu compte de cette dynamicité. Ainsi, nous proposons dans le chapitre 3 un mécanisme de découverte de mappings pouvant être déduits par inférence à partir des mappings existants. Ces raccourcis de mappings limitent les pertes de connaissances occasionnées par la déconnexion d'un ou plusieurs pairs du réseau.

SomeRDFS est un système distribué dont l'architecture est totalement décentralisée. Aucun pair n'a une vision globale des connaissances du PDMS. Chaque pair ne connaît que les éléments qu'il a définis. Cette architecture totalement décentralisée a beaucoup influencé l'approche de découverte de mappings que nous proposons dans cette thèse. Elle explique que cette approche prend appui, au maximum, sur les mécanismes de raisonnement mis en œuvre dans SomeRDFS, ceux-ci permettant notamment de découvrir les éléments composant les ontologies de pairs distants. Les mécanismes de raisonnement sont, de façon plus générale, une constante dans notre approche, utilisés aussi bien pour la phase de sélection des couples à étudier (chapitre 4) que pour la phase consistant à étudier l'existence d'une relation entre les éléments de ces couples (chapitre 5). L'utilisation de ces mécanismes rapproche nos travaux de ceux basés sur l'exploitation de requêtes, les requêtes auxquelles nous nous intéressons étant celles des utilisateurs déjà posées ou des requêtes générées pour les besoins spécifiques du processus de découverte de mappings.

Du point de vue techniques d'alignement, ce contexte nous a naturellement conduit à adopter des techniques sémantiques, syntaxiques et également à étudier l'adaptation des techniques exploitant des ressources externes, le réseau lui-même pouvant être considéré comme une ressource externe. Les techniques syntaxiques permettent de comparer des éléments par comparaison de leur nom. Les relations de mises en correspondance recherchées étant, dans notre contexte, essentiellement des relations de subsomption, les techniques d'inclusion de noms ont été les techniques syntaxiques que nous avons privilégiées. Nous avons éliminé l'étude des méthodes extensionnelles, bien qu'en théorie applicables dans notre contexte, du fait de la masse des données pouvant être représentées dans les PDMS.

Enfin, nos travaux se situent dans le cadre des approches semi-automatiques. Des pro-

positions de mappings sont faites à un administrateur qui doit les valider. Nous assistons l'administrateur dans la vérification de la correction des mappings générés automatiquement. Notons également que l'administrateur peut être mis à contribution au départ du processus de découverte de mappings. En effet, un ensemble de mappings initiaux est nécessaire, même s'il peut être limité. Cet ensemble peut être défini via des échanges entre les administrateurs des pairs concernés.

Conclusion

Ce chapitre a eu pour objectif de dresser un état de l'art sur les travaux effectués en alignement d'ontologies ainsi que sur les travaux portant sur la découverte de mappings en environnement distribué. Nous avons ensuite positionné notre travail de thèse par rapport à cet existant en mettant en avant les problématiques centrales qui nous ont intéressé et auxquelles nous avons apporté des solutions. Ce positionnement a été précédé d'une description de SomeRDFS, correspondant à notre contexte de travail. Les chapitres suivants décrivent notre approche.

Chapitre 2

Propriétés des mappings recherchés

Sommaire

Introduction

La recherche de mappings est un processus coûteux. Il n'est donc pas raisonnable de vouloir ajouter tous les mappings qu'il serait possible de trouver au sein d'un système composé de très nombreux pairs. Il est alors nécessaire de se focaliser sur un sous-ensemble de mappings présentant certaines propriétés. Certaines de ces propriétés, comme l'optimalité des mappings décrite en section 2.1, sont liées au système et permettent d'identifier les mappings qui ne sont pas indispensables car ils ne permettent pas d'enrichir les réponses aux requêtes. D'autres propriétés dépendent de choix de l'administrateur. Elles sont prises en compte au sein de *stratégies*. Nous décrivons plusieurs de ces *stratégies* en section 2.2.

2.1 Notion de mapping optimal

Afin d'enrichir les réponses aux requêtes, nous souhaitons représenter tous les liens de généralisation et de spécialisation qui peuvent être trouvés entre deux éléments appartenant aux ontologies de deux pairs distincts. Un mapping correspond à une telle représentation. Cependant, certains mappings ne permettent pas d'enrichir les réponses aux requêtes. Au contraire, ils alourdissent le processus de réécriture exploité par les mécanismes de raisonnement mis en œuvre dans SomeRDFS, et introduisent de la confusion dans l'interprétation, par les utilisateurs, des éléments des différentes ontologies. Nous ne

souhaitons donc pas représenter tous les liens de généralisation et de spécialisation par des mappings. Par exemple, dans la figure 2.1, le mapping $\mathcal{P}_2$:$Peintre \Rightarrow \mathcal{P}_1$:$Profession$ peut amener les utilisateurs à penser que $\mathcal{P}_2$:$Peintre$ désigne les peintres en bâtiment et non les artistes contrairement à ce qu'indique le mapping $\mathcal{P}_2$:$Peintre \Rightarrow \mathcal{P}_1$:$Artiste$.

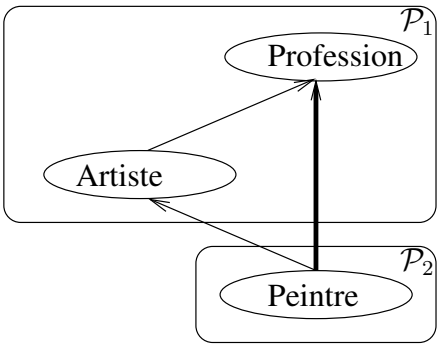

FIG. 2.1 – Exemple de mappings liés au concept de Peintre

De plus, $\mathcal{P}_2$:$Peintre$ est représenté deux fois comme un spécialisant de $\mathcal{P}_1$:$Profession$. Les mécanismes de raisonnement de SomeRDFS sont prévus pour traiter ce type de redondance qui apparaît nécessairement dans un système pair-à-pair. Cependant, plus il y a de redondance, plus ce traitement est coûteux. Le lien entre $\mathcal{P}_2$:$Peintre$ et $\mathcal{P}_1$:$Profession$ étant représenté par la composition du mapping $\mathcal{P}_2$:$Peintre \Rightarrow \mathcal{P}_1$:$Artiste$ et du lien ontologique $\mathcal{P}_1$:$Artiste \Rightarrow \mathcal{P}_1$:$Profession$, il n'est pas nécessaire de le représenter par le mapping $\mathcal{P}_2$:$Peintre \Rightarrow \mathcal{P}_1$:$Profession$. Ce mapping est dit non optimal par opposition au mapping $\mathcal{P}_2$:$Peintre \Rightarrow \mathcal{P}_1$:$Artiste$ qui est dit optimal.

Les mappings optimaux représentent les liens de généralisation et de spécialisation les plus précis au sens où si $M \equiv E_2 \Rightarrow E_1$ est un mapping optimal entre les ontologies $\mathcal{O}_1$ de $\mathcal{P}_1$ et $\mathcal{O}_2$ de $\mathcal{P}_2$, E_2 est l'élément le plus général de $\mathcal{O}_2$ spécialisant E_1 et E_1 est l'élément le plus spécifique de $\mathcal{O}_1$ généralisant E_2. Les mappings optimaux sont les seuls à enrichir les réponses aux requêtes.

Définition 6 (mapping optimal)
Étant donnés deux éléments E_1 et E_2 appartenant respectivement aux ontologies $\mathcal{O}_1$ et $\mathcal{O}_2$, le mapping $E_2 \Rightarrow E_1$ est un mapping optimal ssi $\nexists E_3 \in \mathcal{O}_1 \cup \mathcal{O}_2$ tq ($E_2 \Rightarrow E_3$ et $E_3 \Rightarrow E_1$).

2.2 Notion de stratégie

L'administrateur peut également choisir de limiter sa recherche de mappings pour se focaliser sur des mappings qui présentent un intérêt particulier pour lui. Il peut ainsi ne rechercher des mappings qu'avec certains pairs comme nous le décrivons en section 2.2.1. Il peut également, comme nous le présentons en section 2.2.2, rechercher uniquement un certain type de mappings. Enfin, l'administrateur est capable de préciser quels sont les manques dans la description de la connaissance du pair qu'il administre. Les mappings

recherchés ont alors pour objectif, comme nous le montrons en section 2.2.3, de combler ces manques.

2.2.1 Stratégies sur les pairs

Le problème du nombre d'éléments à aligner est directement lié au nombre de pairs dans le système. Une solution consiste à ne rechercher des mappings qu'avec certains pairs choisis par l'administrateur. La première étape est de préciser quels sont les pairs, appelés *pairs d'intérêt*, avec lesquels des mappings doivent être trouvés. Le choix de ces pairs constitue une stratégie de l'administrateur. Dans ce qui suit, nous décrivons différentes stratégies, leurs différents niveaux d'application possibles et leur mise en œuvre.

Description des stratégies sur les pairs

Une stratégie sur les pairs est la caractérisation des pairs choisis par l'administrateur.

Définition 7 (pair d'intérêt) *Soient un pair $\mathcal{P}$, une propriété P sur les pairs $\mathcal{P}$:$P(\mathcal{P})$ et une stratégie S. $\mathcal{P}$ est un pair d'intérêt selon S si $P(\mathcal{P})$ est vérifiée.*

Nous proposons quatre caractérisations :

- stratégie extentionnelle : l'administrateur précise un ensemble de pairs avec lesquels il souhaite trouver des mappings.
- stratégie par défaut : l'administrateur souhaite trouver des mappings avec tous les pairs quels qu'ils soient.
- stratégie de rupture de l'isolement : l'administrateur souhaite trouver des mappings avec des pairs qui ne partagent pas encore de liens sémantiques avec le pair local. Plus précisément, dans cette stratégie, l'administrateur souhaite ne pas chercher de mappings avec des pairs qui partagent déjà des liens sémantiques avec le pair local.
- stratégie de rapprochement : l'administrateur souhaite trouver des mappings avec des pairs qui partagent déjà des liens sémantiques avec le pair local.

S'il le souhaite, l'administrateur peut suivre une stratégie extentionnelle en donnant une liste de pairs d'intérêt. La propriété vérifiée par les pairs d'intérêt est alors d'appartenir à cette liste.

Par défaut, les mappings sont recherchés avec tous les pairs.

Une liste statique de pairs d'intérêt est mal adaptée au contexte pair-à-pair qui évolue de façon continue. Il est ainsi préférable de choisir une stratégie intentionnelle reposant sur une propriété pour laquelle les pairs ne sont pas invariants. C'est le cas des stratégies de rupture de l'isolement et de rapprochement qui exploitent le nombre de liens sémantiques liant les différents pairs à un moment donné.

Lorsque l'administrateur recherche de nouveaux mappings, il précise la stratégie qu'il souhaite suivre. Il est possible à l'administrateur de choisir une stratégie différente à chaque nouvelle recherche de mappings.

Niveau d'application d'une stratégie sur les pairs

Bien qu'une stratégie soit choisie pour un pair, les ensembles de pairs d'intérêt peuvent être précisés à plusieurs niveaux :

- ensemble global de pairs d'intérêt : un unique ensemble de pairs d'intérêt est défini au niveau du pair pour tous les éléments de son ontologie.
- ensemble local de pairs d'intérêt : un ensemble spécifique de pairs d'intérêt est défini pour chaque élément de l'ontologie. Les mappings sont donc recherchés avec des pairs différents selon l'élément local considéré.
- ensemble hybride de pairs d'intérêt : un ensemble des pairs d'intérêt est défini globalement pour un sous-ensemble donné des éléments de l'ontologie.

Les stratégies extentionnelles sont principalement appliquées aux niveaux global et hybride. En effet, la donnée de listes locales de pairs d'intérêt par l'administrateur est difficilement envisageable à l'échelle de l'ontologie d'un pair. En revanche, les stratégies intentionnelles sont applicables aux trois niveaux. La propriété sur laquelle repose une stratégie est alors vérifiée lors de l'application de la stratégie. Par exemple, pour les stratégies intentionnelles que nous avons décrites, on vérifie l'existence de liens sémantiques entre les pairs à partir de tous les éléments de l'ontologie pour trouver l'ensemble global de pairs d'intérêt, à partir d'un seul élément pour trouver son ensemble local de pairs d'intérêt et à partir d'un sous-ensemble des éléments de l'ontologie pour trouver un ensemble hybride de pairs d'intérêt.

Mise en œuvre

Dans le cas de la stratégie extentionnelle, les pairs d'intérêt sont donnés par l'administrateur. Dans le cas de la stratégie par défaut, tous les pairs sont des pairs d'intérêt. Dans le cas des stratégies intentionnelles, les pairs d'intérêt dépendent de la stratégie précise et du niveau d'application de la stratégie. Dans le cas des stratégies de rupture de l'isolement et de rapprochement, les pairs d'intérêt sont identifiés en fonction du nombre de liens sémantiques partagés avec le pair local. Pour trouver ces liens, deux types de connaissances peuvent être exploitées :

- les connaissances du pair composées de son ontologie et de ses mappings. Un lien sémantique entre deux pairs $\mathcal{P}_1$ et $\mathcal{P}_2$ correspond à un mapping de spécialisation liant un élément de l'ontologie de $\mathcal{P}_1$ et un élément de l'ontologie de $\mathcal{P}_2$. Ce qui se traduit formellement par : $\exists \mathcal{P}_1{:}E_i\ \exists \mathcal{P}_2{:}E_j$ tel que $\mathcal{P}_2{:}E_j \Rightarrow \mathcal{P}_1{:}E_i$ est un mapping.
- les connaissances obtenues en exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS. Il existe un lien sémantique entre deux pairs $\mathcal{P}_1$ et $\mathcal{P}_2$ si un élément de l'ontologie de $\mathcal{P}_2$ est un spécialisant d'un élément de l'ontologie de $\mathcal{P}_1$. La pose d'une requête unitaire permet de connaître tous les spécialisants de l'élément composant cette requête. De façon formelle, on a : $\exists \mathcal{P}_1{:}E_i\ \exists \mathcal{P}_2{:}E_j \in RW$ où RW est l'ensemble des réécritures de la requête $Q \equiv \mathcal{P}_1{:}E_i$

Pour ces deux stratégies intentionnelles, on identifie d'abord les pairs connus. Cette identification dépend du niveau d'application de la stratégie et des connaissances exploi-

tées :

- Pour une stratégie globale exploitant les connaissances locales, les éléments distants connus sont extraits des mappings. Un pair est connu si l'un de ces éléments distants appartient à l'ontologie de ce pair.
- Pour une stratégie globale exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS, il est nécessaire de poser une requête pour chaque élément de l'ontologie locale puis de vérifier les pairs qui participent aux réécritures de cette requête. Notons que pour limiter le nombre de requêtes posées, il est possible de rechercher, dans un premier temps, les éléments les plus généraux de l'ontologie locale. Il n'est alors nécessaire de poser des requêtes que pour ces éléments.
- Pour une stratégie locale exploitant les connaissances locales, on identifie, pour un élément local e, l'ensemble de ses spécialisants locaux. On sélectionne ensuite les mappings de spécialisation faisant intervenir e ou un de ses spécialisants. On extrait de ces mappings les éléments distants. Un pair est connu localement de e si l'un de ces éléments distants appartient à l'ontologie de ce pair.
- Pour une stratégie locale exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS, il est nécessaire de poser la requête $Q \equiv e$. Les pairs qui participent aux réécritures de Q sont des pairs connus localement de e.
- Pour une stratégie hybride exploitant les connaissances locales, les pairs connus de chaque élément de l'ensemble considéré sont identifiés localement en exploitant les connaissances locales. L'ensemble hybride de pairs connus est alors l'union des ensembles locaux de pairs connus.
- Pour une stratégie hybride exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS, les pairs connus de chaque élément de l'ensemble considéré sont identifiés localement en exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS. L'ensemble hybride de pairs connus est alors l'union des ensembles locaux de pairs connus.

Une fois les pairs connus identifiés, on trouve les pairs d'intérêt. Dans le cas de la stratégie de rapprochement, l'ensemble des pairs d'intérêt est l'ensemble des pairs connus. Pour la stratégie de rupture de l'isolement, les pairs d'intérêt sont les pairs qui n'appartiennent pas à l'ensemble des pairs connus.

Notons que, dans le cas des stratégies de rapprochement et de rupture de l'isolement, les pairs d'intérêt sont, respectivement, des pairs connus et des pairs inconnus. L'introduction d'un seuil permet de raffiner ces stratégies de sorte que les pairs d'intérêt soient, respectivement, des pairs suffisamment connus et des pairs insuffisamment connus. L'ensemble hybride de pairs d'intérêt obtenu à partir de tous les éléments de l'ontologie n'est alors pas nécessairement équivalent à l'ensemble global des pairs d'intérêt pour une même stratégie.

2.2.2 Stratégies sur les types de mappings recherchés

Du fait du formalisme utilisé dans SomeRDFS, trouver un mapping entre deux éléments $\mathcal{P}_1{:}E_1$ et $\mathcal{P}_2{:}E_2$ revient à vérifier la validité d'au moins une des deux implications logiques $\mathcal{P}_1{:}E_1 \Rightarrow \mathcal{P}_2{:}E_2$ ou $\mathcal{P}_2{:}E_2 \Rightarrow \mathcal{P}_1{:}E_1$. Lorsqu'on cherche des mappings pour les

éléments de l'ontologie de $\mathcal{P}_1$, il est possible de s'intéresser uniquement aux mappings de spécialisation ou, au contraire, uniquement aux mappings de généralisation. Le choix d'un type de mapping relève d'une stratégie de l'administrateur. Nous proposons donc les trois stratégies suivantes :

- Stratégie par défaut. L'administrateur souhaite trouver à la fois des mappings de généralisation et des mappings de spécialisation.
- Stratégie orientée mappings de spécialisation. L'administrateur souhaite trouver uniquement des mappings de spécialisation. Pour un pair $\mathcal{P}_1$ donné, ces mappings sont de la forme $\mathcal{P}_{i\neq 1}{:}E_j \Rightarrow \mathcal{P}_1{:}E_k$.
- Stratégie orientée mappings de généralisation. L'administrateur souhaite trouver uniquement des mappings de généralisation. Pour un pair $\mathcal{P}_1$ donné, ces mappings sont de la forme $\mathcal{P}_1{:}E_k \Rightarrow \mathcal{P}_{i\neq 1}{:}E_j$.

En appliquant la stratégie orientée mappings de spécialisation ou orientée mappings de généralisation, trouver un mapping entre deux éléments $\mathcal{P}_1{:}E_1$ et $\mathcal{P}_2{:}E_2$ revient alors à vérifier la validité d'un seule des deux implications précédentes. Ces stratégies permettent de réduire le nombre d'alignements à effectuer sans nécessairement modifier le nombre d'éléments à aligner.

2.2.3 Stratégies sur la qualité de connaissance représentée

Nous faisons l'hypothèse que la description de la connaissance d'un pair (ontologie et mappings) est correcte. On suppose, cependant, que cette description est incomplète. La question que l'on se pose, ici, est de distinguer les cas où la description est incomplète des cas où la description est complète. La connaissance est scindée en deux parties : l'ontologie et les mappings. L'ontologie est gérée localement tandis que les mappings font intervenir des connaissances externes qui ne sont pas aussi bien maîtrisées. Nous proposons donc de considérer séparément ces deux parties de la représentation. Nous traitons d'abord des mappings. Dans ce cas, déterminer si la description de la connaissance est complète correspond à identifier les mappings optimaux et les mappings non optimaux. Nous traitons ensuite de l'ontologie. Déterminer si la description de la connaissance est complète correspond alors au problème d'extension conservative.

Optimalité des mappings existants

Lorsque la description de la connaissance est incomplète, la découverte de mappings permet de la compléter. Cependant, lorsque cette description est déjà complète, la découverte de mappings introduit de la redondance. Dans la section 2.1, nous avons montré que les mappings non optimaux pouvaient être la cause de cette redondance. Cette redondance étant néfaste pour le système, tous les mappings identifiés comme non optimaux doivent être supprimés. Ces mappings ne doivent donc pas non plus être ajoutés. Or, lorsqu'il existe deux mappings de généralisation ou de spécialisation d'un élément donné avec deux éléments d'une même ontologie liés par un lien de subsomption, l'un des deux mappings est non optimal. Par exemple, sur la figure 2.1, $Profession$ et $Artiste$ sont deux éléments du pair $\mathcal{P}_1$ liés par le lien de subsomption $\mathcal{P}_1{:}Artiste \Rightarrow \mathcal{P}_1{:}Profession$.

L'élément *Peintre* du pair $\mathcal{P}_2$ est lié aux éléments $Profession$ et *Artiste* de $\mathcal{P}_1$ par les deux mappings $\mathcal{M}_1 \equiv \mathcal{P}_2{:}Peintre \Rightarrow \mathcal{P}_1{:}Artiste$ et $\mathcal{M}_2 \equiv \mathcal{P}_2{:}Peintre \Rightarrow \mathcal{P}_1{:}Profession$. Or, pour $\mathcal{P}_1$, $\mathcal{M}_1$ et $\mathcal{M}_2$ sont deux mappings de spécialisation. L'un des deux n'est donc pas optimal. Dans ce cas, $\mathcal{M}_1$ est le mapping optimal car $\mathcal{P}_1{:}Artiste$ est le généralisant de $\mathcal{P}_2{:}Peintre$ le plus spécifique dans l'ontologie de $\mathcal{P}_1$.

Lors de la découverte de mappings, deux cas sont possibles :

- l'ajout d'un mapping $\mathcal{M}_1$ rend non optimal un autre mapping $\mathcal{M}_2$. Cet autre mapping $\mathcal{M}_2$ doit donc être supprimé.
- on suppose qu'un mapping $\mathcal{M}_2$ est optimal. Il est alors inutile de chercher tout mapping $\mathcal{M}_1$ dont l'ajout rendrait $\mathcal{M}_2$ non optimal.

Par exemple, sur la figure 2.2, $\mathcal{M}_2 \equiv \mathcal{P}_2{:}Peintre \Rightarrow \mathcal{P}_1{:}Profession$ peut être un mapping optimal. Dans ce cas, $\mathcal{P}_2{:}Peintre$ désigne des peintres en bâtiment. Le mapping $\mathcal{M}_1 \equiv \mathcal{P}_2{:}Peintre \Rightarrow \mathcal{P}_1{:}Artiste$ dont l'ajout rendrait $\mathcal{M}_2$ non optimal n'est pas valide. En revanche, si $\mathcal{P}_2{:}Peintre$ désigne des artiste-peintres, $\mathcal{M}_2$ n'est pas optimal et on souhaite le remplacer par $\mathcal{M}_1$.

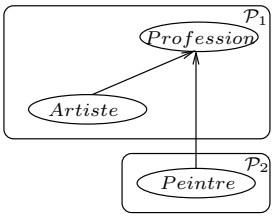

FIG. 2.2 – Exemple d'ambiguïté sur l'optimalité d'un mapping

Cependant, l'optimalité d'un mapping est définie par rapport à la représentation de la connaissance. Dans la plupart des cas, il est impossible de s'appuyer sur cette représentation pour supposer de l'optimalité du mapping $\mathcal{M}_2$. Nous proposons donc que cette supposition soit faite par l'administrateur en lui proposant trois stratégies :

- stratégie par défaut : l'administrateur ne fait aucune supposition sur l'optimalité des mappings existants.
- stratégie d'optimisation des mappings existants : l'administrateur suppose que les mappings existants sont non optimaux et souhaite trouver uniquement des mappings plus précis que les mappings existants afin de les remplacer.
- stratégie de préservation des mappings existants : l'administrateur suppose que les mappings existants sont optimaux et ne souhaite pas aligner deux éléments si cet alignement conduit à la proposition d'un mapping dont l'ajout nécessite la suppression d'un mapping existant.

Extension conservative

Lors de la mise en commun des connaissances des pairs d'un système d'inférence pair-à-pair, de nouvelles connaissances peuvent émerger. Ces nouvelles connaissances peuvent être des liens sémantiques entre des éléments d'un même ontologie. Lorsque ce phénomène d'extension non conservative Abdallah & Goasdoué (2008) est détecté, il doit être résolu

soit par l'ajout de la nouvelle connaissance dans l'ontologie, soit par la suppression d'un des mappings ayant provoqué l'apparition de la nouvelle connaissance. Par exemple, dans la figure 2.3, aucun lien sémantique n'est représenté dans l'ontologie de $\mathcal{P}_1$ entre les classes $\mathcal{P}_1$:*Homme* désignant les individus mâles adulte de l'espèce humaine, et $\mathcal{P}_1$:*Animal* désignant les animaux. Du fait des mappings $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_2$:*Adulte* et $\mathcal{P}_2$:*Adulte* $\Rightarrow$ $\mathcal{P}_1$:*Animal*, le système impose à $\mathcal{P}_1$ la connaissance $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_1$:*Animal*. Si, dans l'ontologie de $\mathcal{P}_1$, les être humains sont des animaux particuliers alors cette nouvelle connaissance doit être représentée. Si, au contraire, dans l'ontologie de $\mathcal{P}_1$, les être humains sont considérés par opposition aux animaux alors $\mathcal{P}_2$:*Adulte* ne peut être à la fois un généralisant de $\mathcal{P}_1$:*Homme* et un spécialisant de $\mathcal{P}_1$:*Animal*.

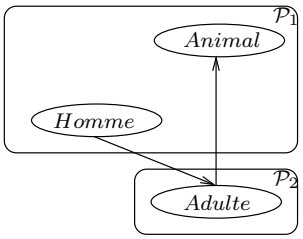

FIG. 2.3 – Exemple d'extension non conservative

Sur notre exemple, le lien entre $\mathcal{P}_1$:*Homme* et $\mathcal{P}_1$:*Animal* est ambiguë dans l'ontologie de $\mathcal{P}_1$: l'absence de représentation d'un lien sémantique entre ces deux classes peut signifier soit qu'il n'existe pas de lien soit que ce lien n'est pas représenté.

Pour traiter ces différents cas nous proposons trois stratégies :

- stratégie par défaut. L'administrateur souhaite décider de la résolution de chaque détection d'extension non conservative individuellement.
- stratégie d'enrichissement de l'ontologie. L'administrateur suppose qu'il existe des liens sémantiques entre des éléments de l'ontologie du pair local qui ne sont pas représentés. Il souhaite trouver des mappings provoquant une extension non conservative afin de compléter l'ontologie du pair local.
- stratégie de préservation de l'ontologie. L'administrateur suppose que tout lien entre deux éléments de l'ontologie du pair local est représenté. Lorsque le phénomène d'extension non conservative est détecté, il est systématiquement résolu par la suppression de mappings.

Conclusion

Dans ce chapitre, nous avons présenté un ensemble de notions permettant d'orienter la recherche de mappings. Dans la mesure où il est impossible de découvrir tous les mappings, notre objectif est d'aider l'administrateur à se focaliser sur la découverte de mappings qui l'intéressent et qui n'encombrent pas le système. Nous ferons référence à ces notions au sein des différentes méthodes de découverte de mappings présentées dans cette thèse.

Chapitre 3

Méthode de sélection des mappings inférables

Sommaire

Introduction

Nous nous intéressons, dans ce chapitre, à la découverte de mappings qui peuvent être déduits par inférence à partir des mappings existants. Nous les appelons *raccourcis de mappings*.

Dans un premier temps, nous précisons la définition de *raccourcis de mappings* ainsi que les particularités de ce type de mappings. Nous discutons, entre autres, de l'intérêt de ces mappings dans notre contexte de travail et de la nécessité de sélectionner ceux qui doivent être ajoutés au système. Nous proposons une méthode en deux étapes permettant d'effectuer cette sélection. La première étape de sélection, décrite en section 3.2, exploite les interactions entre les utilisateurs et un pair $\mathcal{P}$ donné pour proposer un ensemble restreint de *raccourcis de mappings*. En faisant la distinction, parmi les requêtes des utilisateurs, entre les demandes de réécriture et les demandes d'évaluation, l'étude des interactions permet de trouver les mappings utiles pour les utilisateurs. La seconde étape

de sélection, présentée en section 3.3, s'appuie sur des critères de filtrage qui peuvent différer d'un pair à l'autre. Nous présentons un ensemble de critères qui nous semblent pertinents et une mesure exploitant ces critères pour décider automatiquement des *raccourcis de mappings* à retenir.

3.1 Définition d'un raccourci de mappings

3.1.1 Définition

Les *raccourcis de mappings* sont des mappings qui peuvent être déduits par inférence dans le système. Ce sont des raccourcis au sens où ils relient directement des éléments qui sont déjà liés de manière indirecte. Par exemple, dans figure 3.1, le *raccourcis de mappings* $\mathcal{P}_1$:*Pianist* $\Rightarrow$ $\mathcal{P}_2$:*Artist* est équivalent à la composition des mappings $\mathcal{P}_1$:*Pianist* $\Rightarrow$ $\mathcal{P}_3$:*Musician* et $\mathcal{P}_3$:*Musician* $\Rightarrow$ $\mathcal{P}_2$:*Artist*.

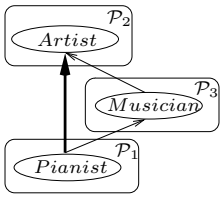

FIG. 3.1 – Exemple de *raccourci de mappings*

Les *raccourcis de mappings* sont des mappings. Par conséquent, un *raccourci de mappings* relie un élément de l'ontologie locale et un élément de l'ontologie d'un autre pair. Un *raccourci de mappings* est une composition de liens sémantiques (mappings et liens ontologiques) faisant intervenir au moins un mapping.

Définition 8 (*raccourcis de mappings*)
Soient deux pairs $\mathcal{P}_i$ et $\mathcal{P}_k$. Soient E_j et E_m deux éléments appartenant respectivement aux ontologies de $\mathcal{P}_i$ et de $\mathcal{P}_k$. Un mapping $\mathcal{P}_i$:E_j $\Rightarrow$ $\mathcal{P}_k$:E_m est un raccourci de mappings s'il existe un élément $\mathcal{P}_{n\neq k}$:E_q tel que $\mathcal{P}_n$:E_q $\Rightarrow$ $\mathcal{P}_k$:E_m est un mapping et que $\mathcal{P}_i$:E_j $\Rightarrow$ $\mathcal{P}_n$:E_q est une connaissance du système (un mapping, un lien ontologique ou une composition de plusieurs de ces relations).

Bien que tout *raccourci de mappings* soit une composition de mappings, toute composition de mappings ne correspond pas nécessairement à un *raccourci de mappings*. Ainsi, une composition de mappings peut permettre d'établir un lien sémantique entre deux éléments dont aucun n'appartient à l'ontologie locale. De même, une composition de mappings peut permettre d'établir un lien sémantique entre des éléments d'une même ontologie. Dans ces deux cas, le lien sémantique représenté par la composition de mappings ne correspond pas à la définition d'un *raccourci de mappings* puisqu'il ne s'agit pas d'un mapping. Par exemple, dans la figure 3.1, si on se place dans le pair $\mathcal{P}_3$, les deux mappings $\mathcal{P}_3$:*Musician* $\Rightarrow$ $\mathcal{P}_2$:*Artist* et $\mathcal{P}_1$:*Pianist* $\Rightarrow$ $\mathcal{P}_3$:*Musician* permettent de déduire le mapping $\mathcal{P}_1$:*Pianist* $\Rightarrow$ $\mathcal{P}_2$:*Artist*. Ce mapping est défini entre un élément de $\mathcal{P}_1$ et un élément de $\mathcal{P}_2$, il ne correspond pas à un mapping de $\mathcal{P}_3$.

3.1.2 Propriétés

Les *raccourcis de mappings* sont des mappings particuliers dans la mesure où ils sont inférables, où ils introduisent de la redondance et où ils rendent certaines connaissances persistantes.

Ces mappings sont inférables à partir des connaissances du système pair-à-pair. Ils peuvent donc être validés automatiquement. Une validation automatique consiste à vérifier l'existence, dans le système, d'un raisonnement construisant le lien sémantique décrit par le mapping à valider. Un *raccourci de mappings* étant une composition de liens sémantiques, ce raisonnement suit donc les liens sémantiques dont il est la composition.

Les *raccourcis de mappings* introduisent de la redondance dans le système. Ainsi, lorsqu'un *raccourci de mappings* $\mathcal{P}_1{:}E_1 \Rightarrow \mathcal{P}_2{:}E_2$ est ajouté au système, il existe alors deux liens entre $\mathcal{P}_1{:}E_1$ et $\mathcal{P}_2{:}E_2$. Le premier lien est le *raccourci de mappings*, le second lien est la composition de mappings à laquelle le *raccourci de mappings* est équivalent.

Cette redondance est encombrante car elle surcharge le système lors du processus de réponse aux requêtes. En effet, les mécanismes de réponse aux requêtes exploitent séparément chaque mapping et les *raccourcis de mappings* permettent d'obtenir des réponses déjà obtenues grâce aux mappings qui les composent.

Cette redondance peut toutefois être intéressante. Un premier intérêt vient de la nécessité de connaître le vocabulaire d'un pair pour l'interroger. On suppose qu'un utilisateur n'interagit qu'avec un seul pair. Quand il pose ses requêtes, il ne peut donc utiliser que des éléments de l'ontologie de ce pair. Lorsque le vocabulaire d'un pair est étendu à ses mappings, les *raccourcis de mappings* présentent un intérêt car ils peuvent permettre de poser des requêtes plus précises en utilisant un élément de l'ontologie d'un autre pair qui correspond plus exactement à ce que cherche l'utilisateur que n'importe quel élément de l'ontologie du pair local. Par exemple, dans la figure 3.1, le *raccourci de mappings* $\mathcal{P}_1{:}Pianist \Rightarrow \mathcal{P}_2{:}Artist$ permet aux utilisateurs de $\mathcal{P}_2$ de poser une requête portant sur des pianistes, c'est à dire utilisant la classe $\mathcal{P}_1{:}Pianist$ pour laquelle il n'existe aucun équivalent dans l'ontologie de $\mathcal{P}_2$.

Enfin, une autre caractéristique des *raccourcis de mappings* provient du fait que les pairs sont autonomes et peuvent se déconnecter à n'importe quel moment. Lorsqu'un pair se déconnecte, les données et les connaissances qu'il apporte au système disparaissent avec lui. Les *raccourcis de mappings* permettent de conserver une partie des connaissances de ce pair malgré sa déconnexion. Par exemple, dans la figure 3.1, si le pair $\mathcal{P}_3$ se déconnecte, en l'absence de *raccourcis de mappings*, le pair $\mathcal{P}_2$ ne peut obtenir de données correspondant à $\mathcal{P}_2{:}Artist$ ni par $\mathcal{P}_3$ puisqu'il est déconnecté, ni par $\mathcal{P}_1$ car la déconnexion de $\mathcal{P}_3$ a fait disparaître le lien entre $\mathcal{P}_2{:}Artist$ et $\mathcal{P}_1{:}Pianist$. Le *raccourci de mappings* $\mathcal{P}_1{:}Pianist \Rightarrow \mathcal{P}_2{:}Artist$ rétablit ce lien et seules les données de $\mathcal{P}_3$ restent inaccessibles tant que ce pair est déconnecté.

Les *raccourcis de mappings* sont intéressants, comme nous venons de le voir, car ils permettent d'augmenter le vocabulaire mis à la disposition des utilisateurs pour poser leurs requêtes et permettent également de limiter les pertes de connaissances occasionnées

par la déconnexion d'un ou plusieurs pairs du réseau. Cependant, une génération systématique de tous les *raccourcis de mappings* est néfaste pour le système dans la mesure où ils surchargent le processus de réponse aux requêtes. Par ailleurs, si les *raccourcis de mappings* sont trop nombreux, il peut devenir difficile, pour les utilisateurs, d'identifier dans le vocabulaire mis à leur disposition, les éléments pertinents pour leurs requêtes. Pour ces raisons, il est nécessaire de sélectionner les *raccourcis de mappings* qui doivent être ajoutés.

3.2 Méthode de sélection des raccourcis de mappings

Selon la définition donnée dans la section précédente, la découverte de *raccourcis de mappings* peut être réalisée en calculant l'ensemble des compositions de mappings à partir des connaissances du système puis à sélectionner celles qui correspondent effectivement à des *raccourcis de mappings*, c'est à dire à des mappings permettant d'enrichir le vocabulaire mis à disposition des utilisateurs d'un pair et/ou permettant de conserver des connaissances malgré la déconnexion de pairs dans le réseau.

Pour réaliser le calcul de l'ensemble des compositions de mappings possibles, une optimisation est proposée dans Halevy *et al.* (2003). Cette approche repose sur l'hypothèse implicite que tous les mappings sont connus. Cette hypothèse ne peut être vérifiée dans le cadre de SomeRDFS dans lequel il n'existe pas de vue sur l'ensemble des pairs du système et selon lequel un pair ne connaît que ses propres mappings. Il est donc impossible de calculer l'ensemble des compositions de mappings à partir des mappings du système. Le calcul de tous les raccourcis de mappings d'un pair $\mathcal{P}$ ne correspond pas non plus à ce que l'on recherche. Certains, en effet, peuvent correspondre à des compositions faisant intervenir deux mappings de $\mathcal{P}$. Il établissent des liens entre des éléments dont aucun n'appartient à $\mathcal{P}$ ou entre des éléments appartenant tous à $\mathcal{P}$. Il ne s'agit donc pas, dans les deux cas, de mappings de $\mathcal{P}$. D'autres *raccourcis de mappings* qui sont des compositions d'un mapping de $\mathcal{P}$ et de connaissances ontologiques de $\mathcal{P}$ ne possèdent pas les propriétés des *raccourcis de mappings* que nous recherchons. En effet, le vocabulaire apporté est déjà disponible pour les utilisateurs de $\mathcal{P}$ car ces *raccourcis* sont construits à partir des connaissances du pair $\mathcal{P}$. Par ailleurs, ils ne rendent aucune connaissance persistante malgré la déconnexion d'un pair du réseau. Chacun de ces *raccourcis* représente, en effet, un lien avec un élément d'une ontologie distante pour lequel il existe déjà un mapping. Par exemple, en figure 3.2, les éléments *Artist*, *Musician*, *Percussionist* et *Drummer* sont représentés dans les ontologies des pairs $\mathcal{P}_1$ et $\mathcal{P}_2$. Supposons que $\mathcal{P}_1$:$Musician \Rightarrow \mathcal{P}_1$:$Artist$ et $\mathcal{P}_2$:$Drummer \Rightarrow \mathcal{P}_2$:$Percussionist$ soient des connaissances ontologiques de $\mathcal{P}_1$ et $\mathcal{P}_2$, respectivement et que $\mathcal{M} \equiv \mathcal{P}_2$:$Percussionist \Rightarrow \mathcal{P}_1$:$Musician$ soit un mapping connu des deux pairs. $\mathcal{M}$ ajoute respectivement $\mathcal{P}_2$:$Percussionist$ et $\mathcal{P}_1$:$Musician$ aux vocabulaires de $\mathcal{P}_1$ et de $\mathcal{P}_2$.

Du point de vue de $\mathcal{P}_1$, il est possible d'inférer le *raccourci de mappings* $\mathcal{M}_1 \equiv \mathcal{P}_2$:$Percussionist \Rightarrow \mathcal{P}_1$:$Artist$. De même, du point de vue de $\mathcal{P}_2$ il est possible d'inférer le *raccourci de mappings* $\mathcal{M}_2 \equiv \mathcal{P}_2$:$Drummer \Rightarrow \mathcal{P}_1$:$Musician$. $\mathcal{M}_1$ et $\mathcal{M}_2$ ne correspondent pas aux *raccourcis de mappings* que nous recherchons car ils ne permettent pas

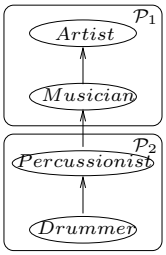

FIG. 3.2 – Exemple de *raccourci de mappings* non pertinent

d'enrichir les vocabulaires respectifs de $\mathcal{P}_1$ et $\mathcal{P}_2$. De plus, les liens sémantiques décrits par $\mathcal{M}_1$ et $\mathcal{M}_2$ n'ont pas pour rôle de permettre la persistance de connaissances.

En revanche, les mécanismes de raisonnement mis à disposition dans SomeRDFS permettent de calculer, à partir d'un élément e donné, toutes les compositions de mappings correspondant à des *raccourcis de mappings* spécialisant e. Une solution naïve consiste alors à interroger le système pour chaque élément de l'ontologie du pair étudié. Cependant, cette solution est coûteuse. De plus, nous ne souhaitons pas ajouter tous les *raccourcis de mappings* calculables mais uniquement ceux qui sont utiles. Nous proposons alors d'exploiter les interactions des utilisateurs avec le système, autrement dit les requêtes des utilisateurs correspondant soit à des demandes de réécriture soit à des demandes d'évaluation, pour limiter le processus de découverte des *raccourcis de mappings* à ceux qui sont intéressants pour les utilisateurs. Un *raccourci de mappings* est intéressant pour les utilisateurs s'il permet de pallier un manque dans le vocabulaire mis à disposition des utilisateurs. Dans SomeRDFS, les requêtes des utilisateurs sont sous la forme d'une conjonction d'éléments. Nous présentons, dans un premier temps, une version simplifiée de notre méthode en ne considérant que des requêtes composées d'un seul élément. Dans un deuxième temps, nous complétons cette présentation en l'étendant aux requêtes composées de plusieurs éléments. Enfin, nous présentons les limites de cette méthode.

3.2.1 Cas des requêtes unitaires

Le processus de réponses aux requêtes, dans SomeRDFS, se décompose en deux étapes : le calcul des réécritures maximales de chaque élément de la requête et l'évaluation des réécritures (cf. 1.3.1). Ces deux étapes peuvent être ou non dissociées. La dissociation des deux étapes est intéressante lorsqu'un utilisateur ne trouve pas, au sein du vocabulaire du pair auquel il s'adresse, l'élément lui permettant de définir précisément les données qu'il recherche, et qu'il est dans l'obligation d'indiquer un autre élément. Cet autre élément peut être plus spécifique. Dans ce cas, toutes les réponses attendues par l'utilisateur ne seront pas obtenues. Il peut être plus général. Dans cet autre cas, la requête posée permettra d'obtenir toutes les réponses attendues mais contiendra également des réponses ne le satisfaisant pas. Dissocier le calcul des réécritures de leur évaluation permet à l'utilisateur de sélectionner les réécritures qui l'intéressent et dont il veut l'évaluation. Par exemple, dans la figure 3.3, un utilisateur à la recherche de pianistes et s'adressant à $\mathcal{P}_2$ ne peut obtenir directement ce qu'il recherche car l'élément *Pianist* n'appartient pas au vocabulaire de $\mathcal{P}_2$. Dans l'ontologie de $\mathcal{P}_2$, l'élément le plus proche de sa recherche est

Artist. L'utilisateur peut obtenir les instances qu'il recherche en demandant l'évaluation d'*Artist* mais il obtient alors également des instances qui ne sont pas des pianistes. Si l'utilisateur dissocie les deux étapes de réponse aux requêtes, il demande, dans un premier temps, la réécriture de $\mathcal{P}_2$:*Artist*. Il obtient, entre autres, $\mathcal{P}_1$:*Pianist* qui correspond exactement à sa recherche. En demandant l'évaluation de cet unique élément il obtiendra les instances précisément recherchées et uniquement celles-là.

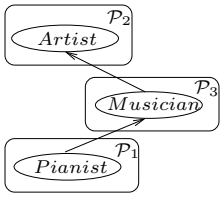

FIG. 3.3 – Scénario de découverte de *raccourcis de mappings*

L'analyse des interactions entre les utilisateurs et le système permet d'identifier les éléments distants qui sont intéressants pour les utilisateurs. Ainsi, dans le scénario décrit précédemment, l'élément que recherche l'utilisateur n'appartient pas au vocabulaire du pair interrogé. S'il existe un élément distant correspondant à la recherche, l'utilisateur en demande l'évaluation. L'ajout d'un mapping vers cet élément permet d'enrichir le vocabulaire du pair interrogé et dispense les utilisateurs de suivre à nouveau le même scénario, lors de recherches futures identiques.

Notons qu'il est possible que l'utilisateur demande l'évaluation d'un élément distant puis se rende compte que cet élément ne correspond pas à ce qu'il recherchait. Dans cette éventualité, une vérification de la satisfaction de l'utilisateur peut être mise en place lors de la présentation des résultats des évaluations demandées. En section 3.3, nous proposons une solution alternative exploitant le nombre d'interactions conduisant à un même *raccourci de mappings*.

L'élément distant évalué e_d est obtenu par réécriture d'un élément local e_l. Cela signifie qu'il existe un raisonnement exploitant les mappings et les ontologies du système pair-à-pair permettant de déduire que e_l est un généralisant de e_d. $e_d \Rightarrow e_l$ est donc un *raccourci de mappings*.

3.2.2 Cas des requêtes conjonctives

Pour obtenir les réponses à une requête conjonctive, il est nécessaire de calculer l'intersection des réponses aux requêtes unitaires portant sur chaque élément qui compose cette requête. L'utilisateur pose une requête conjonctive lorsqu'il ne trouve pas, dans le vocabulaire mis à sa disposition, les éléments décrivant précisément ce qu'il recherche. Sa requête correspond à la définition de ce qu'il recherche. Dans le cas des requêtes conjonctives, le manque de vocabulaire peut avoir deux causes. La première est, comme pour les requêtes unitaires, l'absence d'un élément dans le vocabulaire. Par exemple, supposons que la figure 3.4 schématise les connaissances d'un pair interrogé par un utilisateur qui souhaite obtenir des peintres. En l'absence d'une classe $\mathcal{P}$:*Painter*(X) ou

d'une propriété $\mathcal{P}$:$paints(X, Y)$ dans la connaissance du pair interrogé, l'utilisateur le définit comme un artiste dont les créations sont des peintures. Il pose alors la requête $Q(X) \equiv \mathcal{P}$:$Artist(X) \wedge \mathcal{P}$:$creates(X, Y) \wedge \mathcal{P}$:$Painting(Y)$.

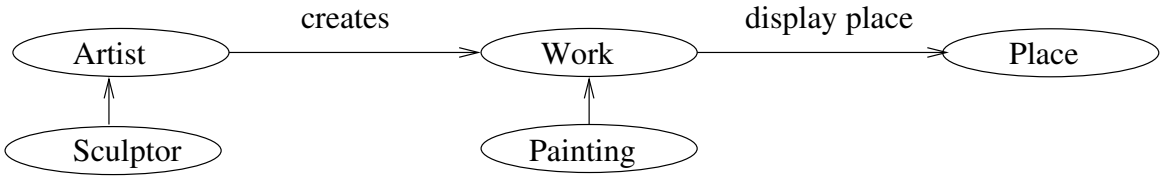

FIG. 3.4 – Exemple de connaissances d'un pair

La seconde cause de manque de vocabulaire provient des contraintes du langage. En effet, RDF(S) ne permet de définir que des classes (relations unaires) et des propriétés (relations binaires). Il est donc impossible de définir des éléments correspondant à des relations n-aires avec n $>$ 2. Par exemple, il est impossible de définir un élément associant des artistes, leurs œuvres et les lieux où ces dernières sont exposées.

Notons que, du fait du manque de vocabulaire, les requêtes conjonctives peuvent être imprécises. Dans ce cas, l'utilisateur est amené à demander, comme dans le cas des requêtes unitaires, la séparation des deux étapes du processus de réponse aux requêtes. Cela lui permet alors de préciser sa requête en sélectionnant les seules réécritures dont il veut l'évaluation.

Dans ce qui suit, nous distinguons deux sous-cas en présence de requêtes conjonctives. Dans le premier cas, l'utilisateur recherche un élément qui correspond précisément à la définition donnée par sa requête. Dans le second cas, il est impossible d'obtenir des éléments d'ontologies correspondant exactement à la requête.

Cas de requêtes conjonctives correspondant à des définitions d'éléments recherchés

L'utilisateur peut parfois vouloir définir précisément un élément recherché (classe ou propriété) par une requête conjonctive. Cependant, certains éléments ne peuvent être définis très précisément et certaines des réponses obtenues ne correspondent alors pas à ce qui est recherché. Par exemple, dans la figure 3.4, lorsque l'utilisateur souhaite obtenir des sculptures, il définit ce qu'il recherche à l'aide d'une requête conjonctive $Q_1(X) \equiv \mathcal{P}$:$\text{Œ}uvre(X) \wedge \mathcal{P}$:$create(Y, X) \wedge \mathcal{P}$:$Sculptor(Y)$. Une sculpture est alors définie comme une œuvre créée par un sculpteur. Cette requête permet bien d'obtenir des sculptures, mais toutes les réponses ne seront pas des sculptures. En effet, si un artiste est à la fois peintre et sculpteur, les peintures de cet artiste figureront également parmi les réponses. Nous nous situons donc dans le contexte où l'utilisateur doit demander la séparation des deux phases de réponse aux requêtes afin de sélectionner les seules réécritures unitaires correspondant à l'élément recherché.

Pour qu'une requête possède des réécritures unitaires il est nécessaire, d'une part, que la tête de cette requête contienne au plus deux variables et, d'autre part, que les autres

variables du corps de la requête ne soient en relation qu'avec une seule variable chacune. Ainsi, par exemple, la requête Q_1 peut admettre une réécriture unitaire. Au contraire la requête $Q_2(X, Y, Z) \equiv creates(X, Z) \wedge displayplace(Z, Y)$ permettant de connaître les artistes, leurs œuvres et les lieux où ces dernières sont exposées ne peut être réécrite de façon unitaire car il est impossible, dans le formalisme de RDF(S), de définir un tel élément. De même, la requête $Q_3(X, Y) \equiv creates(X, Z) \wedge displayplace(Z, Y)$ permettant de connaître les artistes et les lieux où leurs œuvres sont exposées ne peut être réécrite de façon unitaire. Dans ce dernier cas, la raison est que le lien entre les variables demandées n'est pas un spécialisant des éléments utilisés dans le corps de la requête. Or les mécanismes de raisonnement exploitent exclusivement la relation de spécialisation.

Si on se restreint aux requêtes produisant des réécritures unitaires, les demandes d'évaluation par l'utilisateur ne concerneront que ces réécritures unitaires. Pour chaque évaluation d'une réécriture unitaire RW demandée, un *raccourci de mappings* $RW \Rightarrow E_i$ sera alors ajouté pour chaque élément E_i tel que $Q \equiv \wedge_i E_i$. En effet, lorsqu'une telle réécriture RW existe, cela signifie que $\forall E_i$, RW est une réécriture de la requête unitaire $Q_i \equiv E_i$. Il existe donc un raisonnement exploitant les mappings et les ontologies du système pair-à-pair permettant de déduire que chacun des E_i est un généralisant de RW.

L'étape de réécriture peut ne pas produire de réécritures unitaires car le fait qu'une requête possède les propriétés nécessaires citées dans le paragraphe précédent, ne garantit pas que l'élément décrit par cette requête existe dans la connaissance du système. Par exemple, s'il n'existe pas de propriété $\mathcal{P}$:*sculpts* qui vérifie à la fois $\mathcal{P}:sculpts(X, Y) \Rightarrow \mathcal{P}:create(X, Y)$ et $\mathcal{P}:sculpts(X, Y) \Rightarrow \mathcal{P}:sculptor(X)$, alors la requête $Q(X) \equiv \mathcal{P}:Œuvre(X) \wedge \mathcal{P}:create(Y, X) \wedge \mathcal{P}:Sculptor(Y)$ ne possède pas de réécriture unitaire. Dans ce cas, nous proposons d'étendre la méthode présentée dans le paragraphe précédent aux réécritures dont la taille est inférieure à celle de la requête.

Définition 9 (taille d'une requête/réécriture)
Soit une requête Q (ou une réécriture RW) telle que Q (ou RW) $\equiv \wedge_{i=1}^{n} E_i$, alors n est appelé la taille de Q (ou de RW).

À défaut de pouvoir trouver l'élément défini par la requête parmi les réécritures obtenues, on propose d'en identifier les généralisants les plus spécifiques. Ces éléments appartiennent aux réécritures dont la taille est la plus petite. Les demandes d'évaluation par l'utilisateur ne concerneront que ces plus petites réécritures. Contrairement aux réécritures unitaires, l'identification des *raccourcis de mappings*, dans ce cas, n'est pas triviale. Il est nécessaire de déterminer quel(s) élément(s) E_i de la requête $Q \equiv \wedge_i E_i$ est un généralisant d'un élément E'_j donné de la réécriture $RW \equiv \wedge_j E'_j$ dont l'évaluation est demandée.

Cette identification peut se faire par l'étude des réécritures de la même taille que la requête. Par construction, il est possible de trouver, pour chaque élément E_j de RW, au moins une réécriture de même taille n que Q dont $(n-1)$ éléments E_i sont identiques à la requête Q et où l'élément E'_j remplace le n^e élément E_i dont il est un spécialisant. L'algorithme 1 exploite cette propriété pour identifier les *raccourcis de mappings*.

Algorithme 1 : Identification de *raccourcis de mappings*($Q \equiv \wedge_i\{E_i\}$, RW, $\{R\}$)

Entrées : une requête Q, l'ensemble de ses réécritures $\{R\}$ et RW une réécriture particulière

Sorties : l'ensemble RM des *raccourcis de mappings* déductibles à partir de Q et RW

```
1 RM ← ∅
2 pour chaque E'_j de RW faire
3 |   pour chaque R réécriture de Q telle que R ≡ ∧_i({E_i} − E_k + E'_j) faire
4 |   |   RM ← (E'_j ⇒ E_k)
  |   fin
  fin
```

Dans cet algorithme, les réécritures R qui nous intéressent en ligne 3 sont obtenues par remplacement d'un élément E_k de Q par un élément E_j de la réécriture RW. E'_j est donc obtenu par réécriture de E_k et $E'_j \Rightarrow E_k$ est un des *raccourcis de mappings* recherchés.

Une autre méthode consisterait à ne pas poser la requête conjonctive mais uniquement les requêtes unitaires portant sur chacun de ses éléments. On n'a alors aucune difficulté à identifier les *raccourcis de mappings*. Cette méthode est moins coûteuse uniquement si les algorithmes de réponse aux requêtes ne sont pas optimisés. Dans le cas contraire, si le calcul du produit cartésien est optimisé, la première méthode présentée est préférable.

Cas de requêtes conjonctives ne correspondant pas à des définitions d'éléments recherchés

On sait distinguer, comme nous l'avons vu, les requêtes pour lesquelles une réécriture unitaire peut être trouvée de celles pour lesquelles le formalisme ou les mécanismes de raisonnement ne le permettent pas. Dans ce dernier cas, la demande de séparation des deux phases de réponse aux requêtes par l'utilisateur nous indique qu'il ne souhaite pas toutes les réponses à sa requête mais qu'il souhaite préciser sa requête. Cela signifie que l'utilisateur n'a pas pu trouver tous les éléments pour formuler sa requête et que certains des éléments de la requête posée sont plus généraux que ceux qu'il aurait souhaités.

Le traitement proposé dans le cas précédent en l'absence de réécriture unitaire est techniquement envisageable. Cependant, on ne sait pas, cette fois, ce qui est précisément recherché par l'utilisateur. On ne peut donc pas proposer à l'utilisateur uniquement les réécritures de plus petite taille. Or, les réécritures sont obtenues en faisant le produit cartésien des réécritures des requêtes unitaires portant sur chaque élément de la requête conjonctive. Le nombre de réécritures est donc polynomial par rapport à la taille de la requête. Il n'est pas envisageable de laisser l'utilisateur rechercher toutes les réécritures dont il souhaite l'évaluation. Du fait du nombre important de réécritures, certaines évaluations, bien que pertinentes pour l'utilisateur, pourront ne pas être demandées. En effet, il est possible que l'utilisateur ne demande pas l'évaluation de certaines réécritures pertinentes simplement parce que l'étude des très nombreuses réécritures est fastidieuse et que les réécritures pertinentes peuvent être nombreuses.

Nous proposons, dans ce cas, de permettre à l'utilisateur de préciser quelle partie de

la requête est trop générale. Il devient alors possible d'étudier indépendamment les sous-requêtes correspondantes. Chaque sous-requête correspond à la définition précise d'un élément. On applique alors la méthode décrite précédemment. Le nombre de réécritures de l'ensemble des sous-requêtes est inférieur au nombre de réécritures de la requêtes. En effet, la somme des réécritures des sous-requêtes est inférieure à leur produit cartésien. Leur étude par l'utilisateur est donc facilitée. L'application de l'algorithme 1 permet alors de trouver les *raccourcis de mappings* pour chaque sous-requête.

La découverte de *raccourcis de mappings* peut, dans ce cas, être complétée d'une étape de reformulation de requête. En effet, contrairement aux cas des requêtes unitaires et des requêtes pouvant posséder des réécritures unitaires, l'utilisateur n'a toujours pas obtenu, à l'issue du processus de découverte de *raccourcis de mappings*, les réponses recherchées. Pour les obtenir, il lui sera possible de poser des requêtes supplémentaires contruites à partir des *raccourcis de mappings* identifiés. Cette tâche peut être automatisée en remplaçant successivement, dans la requête initiale, chaque sous-requête par l'ensemble des réécritures dont l'utilisateur avait précédemment demandé l'évaluation.

3.2.3 Limites de la méthode

Cette méthode de sélection des *raccourcis de mappings* exploite uniquement la trace des interactions entre les utilisateurs et le système. Le coût de cette méthode est minime car elle ne nécessite pas de poser des requêtes supplémentaires à celles des utilisateurs. En revanche, cette méthode peux générer de "faux" *raccourcis de mappings*. Nous avons vu qu'un utilisateur pouvait demander l'évaluation d'une réécriture ne correspondant pas à sa recherche et nous avons proposé la mise en place d'un système d'enquête de satisfaction pour contrôler ce type d'erreurs. Un autre type d'erreur, moins perceptible par l'utilisateur, concerne le choix des requêtes posées. Les *raccourcis de mappings* générés font automatiquement intervenir les éléments de ces requêtes. Or, s'il est pertinent de supposer qu'un utilisateur pose les requêtes les plus précises possibles, cette hypothèse n'est pas toujours vérifiée. Il arrive donc que des *raccourcis de mappings* non minimaux (cf. 2.1) soient trouvés. Pour contrôler ce type d'erreurs, nous proposons d'utiliser des mécanismes similaires à ceux utilisés pour la découverte de nouveaux mappings qui seront décrits en section 4.3.

Par ailleurs, les *raccourcis de mappings* ainsi trouvés témoignent du manque de vocabulaire dans un pair pour un utilisateur à un moment donné. On justifie l'ajout de ces mappings par le fait que, dans le futur, l'utilisateur risque de rencontrer à nouveau ce manque. Cependant, il est possible que la requête à l'origine d'un *raccourci de mappings* soit ponctuelle et que le *raccourci de mappings* ajouté ne soit plus jamais exploité lors de requêtes futures. Un *raccourci de mappings* peut donc s'avérer inutile pour les utilisateurs, comme nous venons de le décrire, tout en étant utile pour le système en limitant les risques de partitionnement du réseau provoqué par la déconnexion d'un ou plusieurs pairs. Enfin, certains *raccourcis de mappings* ne doivent pas être ajoutés car ils affaiblissent le système en y ajoutant inutilement de la redondance. Pour distinguer ces trois types de *raccourcis de mappings* (utiles pour les utilisateurs, utiles pour le système et

inutiles), nous proposons, en section 3.3, une *fonction d'utilité* permettant d'automatiser la sélection des *raccourcis de mappings* à ajouter.

3.3 Filtrage des *raccourcis de mappings*

La méthode de découverte des *raccourcis de mappings* que nous proposons ne se résume pas à ajouter au coup par coup les *raccourcis de mappings* à chaque fois qu'un utilisateur en exprime le besoin. Un *raccourci de mappings* n'est ajouté que s'il se révèle utile pour les utilisateurs ou pour le système. Nous décrivons, dans un premier temps, un ensemble de critères permettant d'estimer l'utilité d'un *raccourci de mappings*. Dans un deuxième temps, nous proposons une *fonction d'utilité* permettant de décider automatiquement si un *raccourci de mappings* donné doit être ajouté ou non.

3.3.1 Critères d'estimation de l'utilité d'un raccourci de mappings

L'utilité d'un *raccourci de mappings* pour les utilisateurs ne dépend pas directement du mapping mais des conditions dans lesquelles la méthode décrite dans la section précédente a permis de trouver ce mapping. Un *raccourci de mappings* est trouvé suite à une interaction entre un utilisateur et le système. Le nombre et les caractéristiques des utilisateurs à l'origine d'un *raccourci de mappings* sont un indice pour juger de l'intérêt de ce mapping.

A contrario, l'utilité d'un *raccourci de mappings* pour le système est inhérente à ce mapping. Elle dépend uniquement des éléments mis en jeu. L'ajout d'un mapping donné est favorisé si les éléments qu'il relie sont en accord avec la stratégie de découverte de mappings définie par le pair.

Nous détaillons ces critères en nous focalisant d'abord sur les critères reflétant l'utilité pour les utilisateurs, puis les critères se rapportant à l'utilité pour le système.

Critère d'utilité pour les utilisateurs

Nous avons vu qu'un *raccourci de mappings* peut être trouvé chaque fois qu'un utilisateur demande la séparation des deux étapes du mécanisme de réponse aux requêtes. Cette condition est nécessaire mais non suffisante pour démontrer l'utilité, pour les utilisateurs, des *raccourcis de mappings* trouvés. Cependant, lorsque plusieurs interactions d'utilisateurs avec le système permettent de trouver un même *raccourci de mappings*, il est probable que ce mapping traduise un besoin durable des utilisateurs. L'ajout de ce *raccourci de mappings* est alors justifié.

Par ailleurs, les interactions avec le système de tous les utilisateurs ne sont pas toutes aussi représentatives les unes des autres. Selon la maîtrise que l'utilisateur a des connaissances du pair interrogé et du fonctionnement du système, il est pertinent de minorer ou de majorer l'impact de ses interactions avec le système. Par exemple, un utilisateur qui découvre le système a tendance à se tromper. De plus, lorsqu'il se trompe, il ne s'en

aperçoit pas nécessairement et peut persister dans son erreur. Les interactions répétées de ce type d'utilisateur peuvent fausser la détection des *raccourcis de mappings* utiles pour les utilisateurs. Au contraire, on suppose que l'administrateur d'un pair connaît bien le système et qu'il ne provoque la découverte de *raccourcis de mappings* que parce qu'ils sont utiles. On souhaite alors forcer l'ajout de ces mappings. D'autres catégories d'utilisateurs peuvent être distinguées comme les utilisateurs intensifs, les utilisateurs sur le long terme... Pour chaque catégorie d'utilisateurs, on souhaite pouvoir modifier l'impact de ses interactions avec le système sur le processus de découverte de *raccourcis de mappings*.

Critère d'utilité pour le système

Un *raccourci de mappings* est un mapping. Il met donc en jeu un élément de l'ontologie locale et un élément d'une ontologie distante. Il peut être intéressant de favoriser les mappings faisant intervenir certains éléments particuliers. Ces éléments sont définis par l'administrateur du pair local. Cette définition peut être extensionnelle. Dans ce cas l'administrateur précise un par un les éléments pour lesquels il souhaite favoriser la découverte de *raccourcis de mappings*. Cependant, la définition extensionnelle est peu adaptée au contexte pair-à-pair. Une définition intentionnelle est mieux adaptée. Nous proposons, ici, plusieurs exemples de définitions.

Concernant les éléments distants, peu d'informations sont disponibles. Toutefois, on peut favoriser la découverte d'un *raccourci de mappings* lorsque l'élément distant qu'il fait intervenir appartient à un (ensemble de) pair(s) particulier(s). Ce type de sélection correspond à la mise en œuvre de stratégies particulières sur les pairs présentées en 2.2.

Concernant les éléments locaux, plusieurs critères peuvent être combinés. Il peut être intéressant de favoriser la découverte de *raccourcis de mappings* pour les éléments à certains niveaux dans les hiérarchies de classes ou de propriétés. Par exemple, les mappings trouvés pour les éléments les plus spécifiques dans l'ontologie sont utilisés pour répondre à plus de requêtes que ceux trouvés pour les éléments plus généraux. A contrario, les mappings trouvés pour les éléments les plus généraux dans l'ontologie permettent des réponses plus variées aux requêtes. Selon que l'on souhaite favoriser l'un ou l'autre de ces aspects, il convient de favoriser l'un ou l'autre type d'éléments.

On peut également favoriser la découverte de *raccourcis de mappings* pour les propriétés par rapport aux classes ou inversement. Les classes sont plus simples à manipuler pour les utilisateurs que les propriétés. Par conséquent, les *raccourcis de mappings* sont plus souvent trouvés à partir des classes qu'à partir des propriétés. Pour contrebalancer cette tendance, l'administrateur doit favoriser la découverte de *raccourcis de mappings* à partir de propriétés. Toutefois, si les propriétés sont plus difficiles à manipuler pour les utilisateurs, la probabilité d'erreur d'interprétation par les utilisateurs augmente. Le choix de l'administrateur peut alors être de défavoriser ce type de *raccourcis de mappings*.

Enfin, on peut favoriser la découverte de *raccourcis de mappings* pour des éléments pour lesquels il existe peu de mappings. De tels éléments sont particulièrement sensibles aux évolutions du réseau (connexion et déconnexions de pairs) et la découverte de *raccourcis de mappings* permet de réduire cette sensibilité. La définition et la sélection de ces éléments est décrite en 4.1.

Cette liste de définitions intentionnelles possibles n'est pas exhaustive. Par ailleurs, les cas décrits ne tiennent chacun compte que d'une caractéristique des éléments mis en jeu. D'autres définitions, plus complexes, peuvent être obtenues en combinant plusieurs de ces caractéristiques. Le choix d'une définition relève de la notion de stratégie présentée en 2.2.

3.3.2 Mise en œuvre des critères d'utilité

Nous avons présenté un ensemble de critères à prendre en compte pour mesurer l'utilité d'un *raccourci de mappings*. Nous présentons maintenant une *fonction d'utilité*[1] permettant la mise en œuvre de ces critères dans le but d'automatiser la décision d'ajout des *raccourcis de mappings* trouvés.

Pour tenir compte de l'utilité d'un *raccourci de mappings* pour le système, nous proposons de pondérer les éléments locaux. Le poids $W(E)$ d'un élément est donné en fonction de la stratégie choisie pour le pair. Le poids des éléments est normalisé : si la découverte d'un mapping mettant en jeu cet élément est peu utile pour le système, son poids est nul. Si la découverte d'un mapping mettant en jeu cet élément est très utile pour le système, son poids vaut 1. Les mappings faisant intervenir des éléments distants ne correspondant pas à la stratégie sur les pairs ne sont simplement pas considérés.

Pour tenir compte des différences entre les utilisateurs, nous proposons l'utilisation d'un système d'identification lors de l'utilisation du PDMS. L'administrateur donne à chaque utilisateur un poids $W(U)$ relatif à son importance. Pour simplifier l'attribution des poids, les utilisateurs peuvent être assignés à des groupes et le poids d'un utilisateur est alors calculé à partir des groupes auxquels il appartient. Le poids des utilisateurs est normalisé : un utilisateur peu important a un poids nul et un utilisateur important a un poids de 1.

La fonction d'utilité que nous proposons prend en compte ces deux poids qui reflètent chacun un ensemble de critères. Cette fonction prend également en compte le nombre de fois qu'un *raccourci de mappings* est trouvé. Dans le tableau 3.1, nous donnons la valeur de la fonction d'utilité d'un *raccourci de mappings* m_j lorsque ce mapping a été trouvé n fois parmi M *raccourcis de mappings* trouvés. $W(U_{i,j})$ est le poids de l'utilisateur pour la i^{eme} occurence du mapping m_j. $W(E_j)$ est le poids de l'élément local du mapping m_j.

n : # d'occurrences de m_j	valeur de la fonction d'utilité
$n \geq 80\% \times M$	1
$50\% \times M \leq n < 80\% \times M$	$\text{Max}\left(0.5, \frac{\sum_{i=1}^{n} W(U_{i,j}) + n \times W(R_j)}{2n}\right)$
$n < 50\% \times M$	$\frac{\sum_{i=1}^{n} W(U_{i,j}) + n \times W(R_j)}{2n}$

TAB. 3.1 – Valeur de la fonction d'utilité du mapping m_j ayant n occurrences

[1] Le terme de fonction d'utilité fait référence à l'utilité d'un mapping pour le système. Il n'y a aucun lien avec la notion de fonction d'utilité en économie.

On favorise les *raccourcis de mappings* trouvés de très nombreuses fois dans l'échantillon M (plus de 80%). Ils doivent être ajoutés quels que soient les utilisateurs qui en sont à l'origine et quels que soient les éléments locaux mis en jeu. Lorsque les *raccourcis de mappings* sont trouvés un nombre moindre mais toujours relativement important de fois, la valeur de la fonction d'utilité dépend de la moyenne des poids pour chacun des critères, mais elle est au moins de 0.5. Lorsque les *raccourcis de mappings* sont trouvés un petit nombre de fois dans M (moins de 50%), la valeur de la fonction d'utilité dépend uniquement de la moyenne des poids pour chacun des critères.

Conclusion

Nous avons décrit, dans ce chapitre, une méthode de découverte et de sélection des *raccourcis de mappings*. Cette méthode exploite les mécanismes de raisonnement mis à disposition dans les systèmes pair-à-pair sémantiques pour calculer les *raccourcis de mappings* puis exploite la trace des interactions entre les utilisateurs et le système pour sélectionner, selon des critères précisés par l'administrateur, les *raccourcis de mappings* à ajouter.

La sélection des *raccourcis de mappings* à ajouter est principalement nécessaire du fait de la surcharge du processus de réponse aux requêtes qui serait occasionnée par l'ajout de tous les *raccourcis de mappings*. Une autre alternative serait la modification de du processus de réponse aux requêtes de façon à ce que l'exploitation des mappings soit conditionnelle. Ces perspectives n'ont pas été étudiées plus avant car elles nécessitent la modification des algorithmes de réécriture mis en œuvre dans SomeRDFS.

Chapitre 4

Méthode de sélection des éléments à aligner

Sommaire

Introduction

Dans le chapitre précédent, nous avons proposé une méthode de découverte de mappings inférables. Nous nous intéressons, maintenant, à la découverte de *nouveaux* mappings. Nous proposons, pour cela, une méthode basée à la fois sur l'exploitation des mécanismes de raisonnement mis à disposition dans le système SomeRDFS et sur des techniques d'alignement adaptées au contexte pair-à-pair. Notre méthode comporte deux phases : une phase de sélection et une phase d'alignement. La phase d'alignement, que nous présentons dans le chapitre 5, consiste à étudier des couples d'éléments issus d'ontologies distinctes pour déterminer comment ils sont liés sémantiquement. Dans le contexte pair-à-pair dans lequel nous nous situons, il est impossible d'appliquer ce traitement à

tous les couples d'éléments issus d'ontologies distinctes du système. La phase de sélection, que nous décrivons dans ce chapitre, a pour objectif de déterminer quels couples d'éléments constitueront l'entrée du processus d'alignement.

Dans un contexte pair-à-pair, une sélection par élimination n'est pas envisageable car elle suppose, en premier lieu, de construire l'ensemble de tous les couples d'éléments issus d'ontologies distinctes. Nous proposons une sélection par construction : dans un premier temps, nous définissons les éléments pour lesquels il est *intéressant* de trouver des mappings. Cette étape est réalisée en étudiant la trace des interactions entre les utilisateurs et le système. Nous exploitons, dans un deuxième temps, le contexte d'interprétation de ces éléments pour construire l'ensemble des couples d'éléments qu'il est *pertinent* d'aligner. Cette seconde étape repose sur l'exploitation des mécanismes de raisonnement mis en œuvre dans SomeRDFS. Enfin, dans la mesure où cette sélection peut s'avérer insuffisante, nous présentons un ensemble de critères de filtrage basés sur les stratégies présentées en 2.2 permettant de contraindre l'exploitation des mécanismes de raisonnement et limiter ainsi le nombre de couples d'éléments en entrée du processus d'alignement.

4.1 Sélection des éléments pour lesquels il est intéressant de trouver des mappings

Nous proposons de construire l'ensemble des couples d'éléments qui seront fournis en entrée du processus d'alignement à partir d'un ensemble d'éléments, appelés *éléments cibles*, pour lesquels nous considérons qu'il est *intéressant* de trouver de nouveaux mappings. Nous proposons successivement dans cette section, une définition de ces éléments et une méthode permettant leur identification.

4.1.1 Définition des éléments cibles

La découverte de nouveaux mappings permet au système de fournir des réponses plus riches aux requêtes qui lui sont posées. Tout nouveau mapping est, par définition, intéressant. A priori, il est donc intéressant de chercher à établir des mappings à partir de tous les éléments de l'ensemble des ontologies des pairs du système. Cependant, le nombre de ces éléments pouvant être important, nous avons choisi d'en sélectionner certains en tenant compte de critères liés, d'une part, aux utilisateurs et, d'autre part, au système pair-à-pair lui même (les connaissances détenues par les différents pairs et les mécanismes de raisonnement mis en œuvre). Les éléments ainsi sélectionnés sont appelés *éléments cibles*. On peut caractériser trois types d'éléments cibles : les *éléments cibles élémentaires*, les *points de fort intérêt* et les *points de fort intérêt probable*.

Définition 10 (élément cible)
Un élément $\mathcal{P}$:E du pair $\mathcal{P}$ est un élément cible si $\mathcal{P}$:E est un élément cible élémentaire ou si $\mathcal{P}$:E est un point de fort intérêt ou si $\mathcal{P}$:E est un point de fort intérêt probable.

Les éléments cibles sont des éléments pour lesquels la recherche de mappings est intéressante à la fois du point de vue des utilisateurs d'un pair et du point de vue du

système. Les différents types d'éléments cibles s'obtiennent différemment. L'obtention des *éléments cibles élémentaires* passe par l'étude des spécialisants des éléments de l'ontologie locale. L'obtention des *points de fort intérêt* passe par la prise en compte de la satisfaction des utilisateurs par rapport aux réponses obtenues. Enfin, l'obtention des *points de fort intérêt probable* passe par la caractérisation des requêtes susceptibles de ne pas produire suffisamment de réponses. Les éléments composant ces requêtes sont des *points de fort intérêt probable*.

Éléments cibles élémentaires

Pour définir les éléments cibles élémentaires, nous proposons de définir indépendamment les éléments, appelés *points d'intérêt*, pour lesquels la recherche de mappings est intéressante du point de vue des utilisateurs puis les éléments, appelés *points de blocage*, pour lesquels la recherche de mappings est intéressante du point de vue du système.

Définition 11 (élément cible élémentaire)
Un élément $\mathcal{P}$:E du pair $\mathcal{P}$ est un élément cible élémentaire si et seulement si $\mathcal{P}$:E est un point d'intérêt et un point de blocage de $\mathcal{P}$.

Nous définissons, d'abord, les éléments pour lesquels la recherche de mappings est intéressante du point de vue des utilisateurs. Les éléments de l'ontologie et des mappings d'un pair $\mathcal{P}$ constituent le vocabulaire à partir duquel les utilisateurs de $\mathcal{P}$ peuvent formuler leurs requêtes. Cependant, le vocabulaire n'est pas nécessairement exploité dans son intégralité. La recherche de mappings est primordiale pour les éléments très souvent utilisés dans les requêtes. L'hypothèse sous-jacente est que le nombre de requêtes régulièrement posées à partir d'un pair $\mathcal{P}$ est généralement restreint et n'exploite qu'un sous ensemble du vocabulaire de $\mathcal{P}$ appelé *vocabulaire effectif* de $\mathcal{P}$. La définition d'éléments du *vocabulaire effectif* se formalise de la façon suivante :

Définition 12 (élément du vocabulaire effectif)
Soit $\mathcal{P}$:E un élément du pair $\mathcal{P}$. Soit $\mathcal{Q} \equiv Q_1, \ldots, Q_n$ un ensemble de requêtes posées au pair $\mathcal{P}$ et soit t un seuil défini par l'administrateur du pair $\mathcal{P}$. Soit n le nombre de requêtes Q de $\mathcal{Q}$ telles que $Q \equiv \mathcal{P}$:E $\wedge \{\mathcal{P}_i$:$E_j\}$. $\mathcal{P}$:E est un élément du vocabulaire effectif de $\mathcal{P}$ si $t < n$.

Ainsi, chercher des mappings pour les éléments de ce sous-ensemble du vocabulaire de $\mathcal{P}$ sera pour nous une priorité en ce sens qu'ils viendront enrichir les réponses aux requêtes qui seront très certainement posées dans le futur si les utilisateurs ne modifient pas leur comportement. Pour enrichir ces mêmes réponses, il est également pertinent de chercher des mappings pour les spécialisants des éléments du vocabulaire effectif de $\mathcal{P}$. On définit alors, de la façon suivante, les éléments, appelés *points d'intérêt*, pour lesquels il est intéressant de trouver des mappings du point de vue des utilisateurs :

Définition 13 (point d'intérêt)
Soit $\mathcal{P}$:E un élément du pair $\mathcal{P}$. $\mathcal{P}$:E est un point d'intérêt s'il est un élément du vocabulaire effectif de $\mathcal{P}$ ou s'il existe un élément $\mathcal{P}$:E$'$ du vocabulaire effectif de $\mathcal{P}$ tel que $\mathcal{P}$:E $\Rightarrow \mathcal{P}$:E$'$.

Lorsqu'un pair est interrogé par plusieurs utilisateurs, nous proposons de mettre en place un système de pondération affectant des poids aux requêtes considérées selon les utilisateurs qui les ont posées, comme pour la méthode de découverte de *raccourcis de mappings* (cf. 3.3). Un tel système permet alors de se focaliser sur un sous-ensemble des requêtes et de favoriser celles de certains utilisateurs uniquement.

Nous définissons maintenant les éléments pour lesquels la recherche de mappings est intéressante du point de vue du système. Dans un système pair-à-pair sémantique, les mappings permettent la propagation du raisonnement entre les différents pairs du système. Une insuffisance de mappings se traduit par une propagation limitée du raisonnement. Lorsque le raisonnement est exploité pour répondre à des requêtes, l'objectif est d'obtenir des réponses contruites collectivement par les pairs du système. S'il existe un nombre insuffisant de mappings, le nombre de pairs participant aux réponses est réduit. Dans des cas extrêmes, il est possible que seul le pair interrogé participe aux réponses aux requêtes. Par exemple, considérons la requête $Q(X) \equiv \mathcal{P}_1{:}Artist(X)$ posée au pair $\mathcal{P}_1$ et dont les réécritures sont $\mathcal{P}_1{:}Painter(X)$, $\exists Y, \mathcal{P}_1{:}paints(X,Y)$, $\mathcal{P}_1{:}Sculptor(X)$, $\exists Y, \mathcal{P}_1{:}sculpts(X,Y)$. Toutes les réécritures sont construites à partir d'éléments de $\mathcal{P}_1$, ce qui signifie qu'il n'existe pas de mapping de spécialisation pour $\mathcal{P}_1{:}Artist$. De tels éléments constituent des points de blocage pour le processus de réponse aux requêtes. Il est intéressant d'identifier de tels points de blocage et de générer des mappings pour les supprimer. Dans notre exemple, l'élément $\mathcal{P}_1{:}Artist$ ne possède aucun spécialisant appartenant à l'ontologie d'un pair autre que $\mathcal{P}_1$. De façon plus générale, un point de blocage peut être défini comme un élément n'ayant pas assez de spécialisants provenant des ontologies de pairs distants. Cette définition peut être formalisée de la façon suivante :

Définition 14 (point de blocage)
Soit $\mathcal{P}$:E un élément du pair $\mathcal{P}$ ayant n spécialisants appartenant aux ontologies de pairs autres que $\mathcal{P}$ considérés et soit t un seuil défini par l'administrateur du pair $\mathcal{P}$. $\mathcal{P}$:E est un point de blocage si $n < t$.

Les points de blocage sont donc définis de manière automatique par l'étude des spécialisants des éléments de l'ontologie locale. De même, les points d'intérêt sont définis automatiquement par l'étude des requêtes des utilisateurs. Les éléments cibles élémentaires sont donc définis de manière totalement automatique.

Points de fort intérêt

Lorsqu'une requête est posée au système, il est possible qu'elle possède un nombre relativement important de réécritures mais que le nombre de réponses obtenues ne satisfasse pas l'auteur de la requête. Nous proposons de donner la possibilité aux utilisateurs de signaler au système que sa requête ne lui a pas permis d'obtenir toutes les réponses qu'il attendait. Les éléments composant cette requête sont particulièrement intéressants car en plus d'être des éléments du vocabulaire effectif, on apprend des utilisateurs qu'ils possèdent un nombre insuffisant de mappings les spécialisant. Nous définissons ces éléments ainsi :

Définition 15 (point de fort intérêt)
Soit $\mathcal{P}$:E un élément du pair $\mathcal{P}$ et soit $\mathcal{Q} \equiv Q_1, \ldots, Q_n$ l'ensemble des requêtes posées au

pair $\mathcal{P}$ considérées. $\mathcal{P}$:E est un point de fort intérêt s'il existe une requête $Q \equiv \mathcal{P}$:$E \wedge \{\mathcal{P}_i$:$E_j\}$ de $\mathcal{Q}$ dont les réponses ne satisfont pas les utilisateurs de $\mathcal{P}$ ou s'il existe un élément $\mathcal{P}$:E' tel que $\mathcal{P}$:$E \Rightarrow \mathcal{P}$:$E'$ et qu'il existe une requête $Q \equiv \mathcal{P}$:$E' \wedge \{\mathcal{P}_i$:$E_j\}$ de $\mathcal{Q}$ dont les réponses ne satisfont pas les utilisateurs de $\mathcal{P}$.

Contrairement aux points d'intérêt pour lesquels il est nécessaire de vérifier qu'ils sont également des points de blocage, les points de fort intérêt sont des éléments cibles parce que les utilisateurs ont identifié un manque de mappings les concernant mais ils ne sont pas nécessairement des points de blocage.

Point de fort intérêt probable

L'hypothèse de noms uniques pour les ressources (cf. 1.3.1) suppose qu'une ressource est identifiée de manière unique par tous les pairs du système. Lorsque cette hypothèse n'est pas vérifiée, il est plus probable d'obtenir des réponses à une requête conjonctive si elle possède des réécritures composées d'éléments appartenant à l'ontologie d'un même pair. Par exemple, considérons une réécriture $RW(X) \equiv \mathcal{P}_1$:$create(X, Y) \wedge \mathcal{P}_2$:$Painting(Y)$ correspondant à des peintres. Si l'évaluation de $\mathcal{P}_1$:$create(X, Y)$, d'une part, nous donne, entre autres, $(DeVinci, La\_Joconde)$ et que l'évaluation de $\mathcal{P}_2$:$Painting(Y)$, d'autre part, nous donne, entre autres $Lajoconde$, alors l'évaluation de la réécriture RW complète risque d'être vide du fait de l'hétérogénéité des données entre $\mathcal{P}_1$ et $\mathcal{P}_2$. En revanche, si on considère la réécriture $RW(X) \equiv \mathcal{P}_1$:$create(X, Y) \wedge \mathcal{P}_1$:$Painting(Y)$, on est assuré que les données seront homogènes. L'évaluation de la réécriture ne sera donc pas vide avec les mêmes données que précédemment. Une telle réécriture est dite *homogène*.

Définition 16 (réécriture homogène)
Une réécriture $R \equiv \wedge\{\mathcal{P}_i$:$E_j\}$ est homogène si et seulement si $\forall \mathcal{P}_i$:$E_j \in R$, $\forall \mathcal{P}_k$:$E_m \in R$, $i = k$

Dans le cas des requêtes conjonctives, si l'on ne considère pas l'hypothèse de noms uniques, il est pertinent d'identifier les requêtes qui possèdent peu de réécritures homogènes. On se trouve alors dans un cas proche du manque de réponses signalé par les utilisateurs. Pour les raisons déjà énoncées précédemment, on s'intéressera donc aux éléments composant ces requêtes. Nous appelons ces éléments des *point de fort intérêt probable*.

Définition 17 (point de fort intérêt probable)
Soit $\mathcal{P}$:E un élément du pair $\mathcal{P}$, soit $\mathcal{Q} \equiv \{Q_1, \ldots, Q_n\}$ l'ensemble des requêtes posées au pair $\mathcal{P}$ considérées et soit t un seuil défini par l'administrateur du pair $\mathcal{P}$. Soit $f(Q)$ la fonction qui, à une requête Q associe le nombre de réécritures homogènes qu'elle possède. $\mathcal{P}$:E est un point de fort intérêt probable s'il existe une requête $Q \equiv \mathcal{P}$:$E \wedge \{\mathcal{P}_i$:$E_j\}$ de $\mathcal{Q}$ telles que $f(Q) < t$ ou s'il existe un élément $\mathcal{P}$:E' tel que $\mathcal{P}$:$E \Rightarrow \mathcal{P}$:$E'$ et qu'il existe une requête $Q \equiv \mathcal{P}$:$E' \wedge \{\mathcal{P}_i$:$E_j\}$ de $\mathcal{Q}$ telle que $f(Q) < t$.

Contrairement aux points de fort intérêt, le manque de mappings n'est pas explicitement déclaré pour les points de fort intérêt probable. Par ailleurs, les points de fort intérêt probable ne sont pas nécessairement des points de blocage. Cependant, ils font

partie de requêtes pour lesquelles on a constaté un manque de réécritures homogènes. Il existe donc, pour ces requêtes, un manque supposé de réponses. Une solution pour corriger ce problème est l'ajout de mappings, c'est pourquoi nous proposons d'inclure ces éléments dans les éléments cibles.

Remarque : Dans SomeRDFS, l'hypothèse de noms uniques est supposée vraie. Dans ce cas, le manque de réécritures homogène ne permet pas de conclure à un manque supposé de réponses. Les points de fort intérêt probable ne sont alors pas considérés comme des éléments cibles.

4.1.2 Méthode d'identification des éléments cibles

Après avoir défini les éléments cibles, nous décrivons maintenant une méthode en trois étapes permettant de les identifier.

Tout d'abord, on se focalise sur les éléments constituant le vocabulaire d'un pair donné. On élimine d'emblée tous les éléments appartenant à des ontologies de pairs distants. En effet, le lien sémantique entre un de ces éléments et l'ontologie du pair local est déjà représenté par un mapping. De plus, les mappings entre un tel élément et l'ontologie d'un autre pair ne peuvent être ajoutés aux connaissances du pair local.

Dans un deuxième temps, nous proposons d'étudier la trace des requêtes posées par les utilisateurs pour obtenir l'ensemble des points d'intérêt. Cette trace contient l'ensemble des requêtes posées par les utilisateurs durant une session d'observation. On peut donc en extraire les éléments du vocabulaire effectif.

On conserve, dans la trace, l'identité de l'utilisateur à l'origine de chaque requête. On peut alors associer le poids de cet utilisateur à la requête posée. Lorsque l'utilisateur est important, son poids est supérieur à 1 et chacune de ses requête compte pour plusieurs. Les éléments qui composent ces requêtes deviennent plus rapidement des éléments du vocabulaire effectif. Au contraire, un utilisateur peu important possède un poids quasi nul. Ses requêtes ont donc peu d'influence sur les éléments du vocabulaire effectif.

À partir des réécritures contenues dans la trace, on extrait également les spécialisants des éléments du vocabulaire effectif appartenant à l'ontologie locale. On identifie ainsi tous les points d'intérêt

La trace contient également la satisfaction des utilisateurs par rapport aux réponses obtenues. Les requêtes ayant de telles réponses sont donc facilement identifiables. Il est également possible d'identifier des requêtes particulières telles que les requêtes ayant peu de réécritures homogènes. Les points d'intérêt trouvés à partir de ces deux types de requêtes sont respectivement des points de fort intérêt et des points de fort intérêt probable.

Enfin, pour connaître les points de blocage, il est nécessaire de compter le nombre de spécialisants de chaque élément dans les ontologies de pairs distants. Pour cela on exploite les connaissances disponibles dans le système pair-à-pair. Nous distinguons deux méthodes de comptage. La première exploite uniquement les connaissances du pair local (son ontologie et ses mappings), la seconde exploite les mécanismes de raisonnement mis à disposition dans le système.

Lorsqu'on exploite les connaissances locales, les éléments trouvés qui appartiennent

à des ontologies de pairs distants, apparaissent au sein de mappings. On compte alors le nombre de mappings spécialisant chaque élément local. L'avantage de cette méthode de comptage est que le nombre de spécialisants d'un élément qui appartiennent à des ontologies de pairs distants est constant tant qu'il n'y a pas ajout ou suppression de mappings. Le comptage du nombre de spécialisants des éléments de l'ontologie locale peut donc se faire à n'importe quel moment. Notons, toutefois, qu'il est inutile de vérifier si un élément est un point de blocage si cet élément n'est pas un point d'intérêt. Ce comptage sera donc réalisé en fin de session d'observation. L'inconvénient de cette méthode est que de nombreux éléments appartenant à des ontologies de pairs distants ne sont pas comptabilisés.

L'exploitation des mécanismes de raisonnement mis en œuvre dans le système pair-à-pair SomeRDFS nécessite de poser des requêtes. On pose une requête unitaire construite sur un élément pour connaître tous les spécialisants de cet élément. On compte ensuite le nombre de spécialisants appartenant à des ontologies de pairs distants. Contrairement à la première méthode, tous les éléments appartenant à des ontologies de pairs distants sont bien comptabilisés. Cependant, dans ce cas, le résultat obtenu peut varier sans qu'il y ait ajout ou suppression de mappings dans les connaissances du pair $\mathcal{P}$. En effet, les connaissances du système peuvent évoluer sans que les connaissances du pair $\mathcal{P}$ étudié n'évoluent (connexion/déconnexion de pairs distants, ajout/suppression de connaissances dans des pairs distants). Le calcul du nombre de spécialisants doit tenir compte de cette évolution possible. Nous proposons une méthode de calcul en deux temps : tout d'abord l'identification des points d'intérêt à partir de la trace des requêtes utilisateurs puis la pose de requêtes sur les points d'intérêt à intervalle régulier. Le nombre de spécialisants sera alors une fonction du nombre de spécialisants observés à chaque pose de requêtes (minimum, maximum ou moyenne, par exemple). L'algorithme 2, *ExtractElementaryTargets*, réalise les différentes étapes nécessaires à la construction de l'ensemble des éléments cibles élémentaires.

Cet algorithme est appliqué à l'ensemble de la trace des requêtes des utilisateurs. On sélectionne d'abord les points d'intérêt. Pour cela on liste tous les éléments de toutes les requêtes puis on compte le nombre de requêtes dont fait partie chacun de ces éléments (la procédure *incNbQueries* de la ligne 5 initialise puis incrémente un compteur). Lorsqu'un élément fait partie du vocabulaire effectif, la fonction *isRepresentative* de la ligne 6 renvoie vrai et on ajoute à PI le couple composé de la requête sur cet élément et des réécritures obtenues.

L'algorithme 3, *ExtractNonElementaryTargets*, réalise les différentes étapes nécessaires à la construction des ensembles de points de fort intérêt et de points de fort intérêt probable.

Cet algorithme est appliqué à l'ensemble de la trace des requêtes des utilisateurs.Pour les éléments cibles non élémentaires, la fonction *isRepresentative* de la ligne 4 est appliquée à une requête. Elle permet de déterminer si la requête a été posée un nombre suffisant de fois par les différents utilisateurs. La fonction *notEnoughAnswers* (c.f. ligne 5) vérifie si les utilisateurs ont signalé un manque de réponses pour la requête passée en paramètre. Enfin, la fonction *notEnoughHomogeneousRW* teste si un ensemble de réécritures passées

Algorithme 2 : ExtractElementaryTargets($\{(Q, RW)\}$)

Entrées : Un ensemble de couples composés d'une requête Q et de l'ensemble de ses réécritures RW

Sorties : L'ensemble ETE des éléments cibles élémentaires

```
1  ETE ⇐ ∅
2  PI ⇐ ∅
3  pour tous les (Q) faire
4      pour tous les E_i de Q faire
5          incNbQueries(E_i, poids(Q))
6          si isRepresentative(E_i) alors
7              PI ⇐ (Q' ≡ E_i, Rewrite(Q'))
           fin
       fin
   fin
8  pour tous les (Q, RW de PI) faire
9      pour tous les RW_i de RW faire
10         pour tous les E_j de RW faire
11             si isLocal(E_j) alors
12                 si notEnoughDistantSpec(E_j) alors
13                     ETE ⇐ E_j
                   fin
               fin
           fin
       fin
   fin
```

en paramètre contient un nombre suffisant de réécritures homogènes.

4.2 Sélection des couples d'éléments qu'il est pertinent d'aligner

Dans la section précédente, nous avons sélectionné les *éléments cibles* pour lesquels la découverte de nouveaux mappings est *intéressante*. Nous décrivons maintenant un processus de sélection permettant d'aboutir à un ensemble de couples d'éléments entre lesquels la recherche de mappings est *pertinente*. Pour cela, nous exploitons la notion de contexte d'interprétation que nous présentons, dans un premier temps, de façon simplifiée. Dans un deuxième temps, nous présentons comment prendre en compte les contraintes du formalisme RDF(S) afin de définir, de façon plus précise, le contexte d'interprétation.

Algorithme 3 : ExtractNonElementaryTargets($\{(Q, RW)\}$)

Entrées : Un ensemble de couples composés d'une requête Q et de l'ensemble de ses réécritures RW

Sorties : Deux ensembles d'éléments : SPI les points de fort intérêt et $PSPI$ les points de fort intérêt probable

```
1  SPI ⇐ ∅
2  PSPI ⇐ ∅
3  pour tous les (Q, RW) faire
4      si isRepresentative(Q) alors
5          si notEnoughAnswers(Q) alors
6              pour tous les RW_i de RW faire
7                  pour tous les E_j de RW_i faire
8                      si isLocal(E_j) alors
9                          SPI ⇐ E_j
                       fin
                   fin
               fin
           fin
10         si notEnoughHomogeneousRW(RW) alors
11             pour tous les RW_i de RW faire
12                 pour tous les E_j de RW_i faire
13                     si isLocal(E_j) alors
14                         PSPI ⇐ E_j
                       fin
                   fin
               fin
           fin
       fin
   fin
```

4.2.1 Contexte d'interprétation

Définition

Nous définissons le contexte d'interprétation d'un élément comme l'ensemble des éléments avec lesquels il partage un lien. Cette notion est utilisée pour étudier la proximité sémantique entre deux éléments.

Définition 18 (contexte d'interprétation)
Le contexte d'interprétation d'un élément E est l'ensemble des éléments E_i tels que $E_i \Rightarrow e$ ou $E \Rightarrow E_i$

Nous considérons qu'il est pertinent d'étudier l'existence d'un mapping entre des éléments appartenant à des ontologies distinctes s'ils ont un contexte d'interprétation commun.

Définition 19 (contexte d'interprétation commun)
Deux éléments E_1 et E_2 ont un contexte d'interprétation commun s'il existe un élément E_3 tel que ($E_1 \Rightarrow E_3$ et $E_2 \Rightarrow E_3$) ou ($E_3 \Rightarrow E_1$ et $E_3 \Rightarrow E_2$).

En effet, un contexte d'interprétation commun entre deux éléments peut refléter l'existence d'une proximité sémantique entre ces deux éléments. Ce rapprochement peut ne pas conduire à un mapping dans le système pair-à-pair. De même, l'absence d'un contexte d'interprétation commun entre deux éléments ne signifie pas absence de proximité sémantique. Mais l'absence de proximité sémantique implique l'absence d'un contexte d'interprétation commun.

Exploitation

Le contexte d'interprétation d'un élément est exploité pour trouver des éléments appartenant à des ontologies de pairs distants avec lesquels il est pertinent de rechercher un mapping. La construction des couples d'éléments à aligner découle de l'un des deux scénarios présentés figures 4.1 et 4.2. Dans les deux cas, nous nous plaçons du point de vue du pair $\mathcal{P}_1$ et nous cherchons des éléments qu'il serait pertinent d'aligner avec $\mathcal{P}_1$:E_1.

Dans le premier scénario, présenté en figure 4.1, nous considérons trois pairs $\mathcal{P}_1$, $\mathcal{P}_2$ et $\mathcal{P}_3$, leurs éléments respectifs E_1, E_2 et E_3 ainsi que les mappings $\mathcal{M}_1 \equiv \mathcal{P}_1$:$E_1 \Rightarrow \mathcal{P}_3$:$E_3$ connu de $\mathcal{P}_1$ et $\mathcal{M}_2 \equiv \mathcal{P}_2$:$E_2 \Rightarrow \mathcal{P}_3$:$E_3$ connu de $\mathcal{P}_3$. Si $\mathcal{P}_1$:E_1 est un élément pour lequel il est intéressant de rechercher de nouveaux mappings, il est possible de poser la requête $Q \equiv \mathcal{P}_3$:E_3. Cette requête permet d'obtenir tous les éléments ayant $\mathcal{P}_3$:E_3 comme généralisant commun. On obtient $\mathcal{P}_1$:E_1 et $\mathcal{P}_2$:E_2. $\mathcal{P}_3$:E_3 est leur contexte d'interprétation commun. Nous en déduisons que $\mathcal{P}_2$:E_2 est un élément qu'il serait pertinent d'aligner avec $\mathcal{P}_1$:E_1.

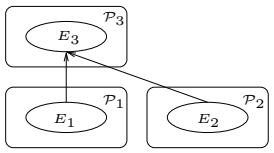

FIG. 4.1 – Scénario 1

Dans ce scénario, il est uniquement nécessaire que les éléments E_1 et E_2 appartiennent à des ontologies de deux pairs distincts $\mathcal{P}_1$ et $\mathcal{P}_2$. L'élément E_3 peut appartenir soit à l'ontologie de $\mathcal{P}_1$, soit à l'ontologie de $\mathcal{P}_2$. De même, le mapping $\mathcal{M}_2$ peut être remplacé par une composition de mappings et de liens ontologiques. Le mapping $\mathcal{M}_1$ peut également être remplacé par un composition de liens ontologiques et de mappings mais cette composition ne doit comporter au plus qu'un mapping et tous les liens ontologiques doivent appartenir à l'ontologie de $\mathcal{P}_1$.

Dans le second scénario, présenté en figure 4.2, nous considérons deux pairs $\mathcal{P}_1$ et $\mathcal{P}_2$ ainsi que les éléments $\mathcal{P}_1$:E_1, $\mathcal{P}_2$:E_2 et $\mathcal{P}_1$:E_3. $\mathcal{P}_1$:$E_3 \Rightarrow \mathcal{P}_1$:$E_1$ est un lien dans l'ontologie

de $\mathcal{P}_1$ et $\mathcal{M} \equiv \mathcal{P}_1{:}E_3 \Rightarrow \mathcal{P}_2{:}E_2$ est un mapping de $\mathcal{P}_1$. Si $\mathcal{P}_1{:}E_1$ est un élément pour lequel il est intéressant de rechercher de nouveaux mappings, il est *pertinent* d'étudier le lien sémantique entre $\mathcal{P}_1{:}E_1$ et $\mathcal{P}_2{:}E_2$ qui généralisent tous les deux $\mathcal{P}_1{:}E_3$.

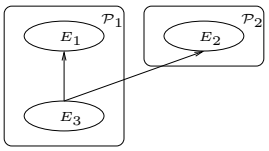

FIG. 4.2 – Scénario 2

Dans ce scénario le lien ontologique entre E_1 et E_3 peut être remplacé par une composition de liens ontologiques et le mapping $\mathcal{M}$ par une composition de liens ontologiques de $\mathcal{P}_1$ et d'un mapping.

Dans les deux scénarios, on conclut qu'il est *pertinent* de rechercher un mapping entre $\mathcal{P}_1{:}E_1$ et $\mathcal{P}_2{:}E_2$ qui partagent respectivement un généralisant commun dans le scénario 1 et un spécialisant commun dans le scénario 2.

En appliquant ces deux scénarios aux éléments sélectionnés par la méthode décrite en section précédente, nous construisons un ensemble de couples d'éléments entre lesquels la recherche de mappings est à la fois *intéressante* et *pertinente*.

Mise en œuvre

Nous avons décrit deux scénarios permettant de trouver des éléments ayant un contexte d'interprétation commun. Nous proposons, ci-dessous, deux algorithmes qui construisent, chacun, un ensemble d'éléments à partir d'un *élément cible*. Ces deux algorithmes (algorithmes 4 et 5) servent de base à la construction de l'ensemble des couples d'éléments qu'il est pertinent d'aligner selon les deux scénarios que nous avons présentés précédemment. L'algorithme de construction de cet ensemble (algorithme 6) est présenté ensuite.

L'algorithme 4 construit un ensemble d'éléments partageant un généralisant commun avec un élément donné.

Algorithme 4 : $SPEC\_MC(\mathcal{P}_{loc}{:}E_j)$

Entrées : Un élément cible E_j du pair $\mathcal{P}_j$
Sorties : Un ensemble $SPEC\_MC$ d'éléments ayant un généralisant commun avec E_j

1 $SPEC\_MC(\mathcal{P}_{loc}{:}E_j) \Leftarrow \emptyset$
2 **pour tous les** $\mathcal{P}_i{:}E_m$ *élément plus général que* $\mathcal{P}_{loc}{:}E_j$ *dans la connaissance de* $\mathcal{P}_{loc}$ **faire**
3 | Add Rewrite($\mathcal{P}_i{:}E_m$) to $SPEC\_MC(\mathcal{P}_{loc}{:}E_j)$
fin

Cet algorithme appliqué à un élément cible $\mathcal{P}_{loc}{:}E_j$ permet de construire l'ensemble $SPEC\_MC(\mathcal{P}_{loc}{:}E_j)$ des éléments partageant un généralisant avec $\mathcal{P}_{loc}{:}E_j$ en exploitant

d'abord les connaissances locales (cf. ligne 2) puis les mécanismes de réécriture (cf. ligne 3)

Cet ensemble permettra, ultérieurement, de générer un ensemble de couples d'éléments *pertinents* à aligner de la forme ($\mathcal{P}_{loc}$:E_j, $\mathcal{P}_{dist \neq loc}$:$E_k$) avec $\mathcal{P}_{dist}$:$E_k \in SPEC\_MC(\mathcal{P}_{loc}$:$E_j)$.

L'algorithme 5 construit un ensemble d'éléments ayant un élément donné pour spécialisant commun.

Algorithme 5 : $GEN\_MC(\mathcal{P}_{loc}$:$E_j)$

Entrées : Un élément cible E_j d'un pair $\mathcal{P}_{loc}$
Sorties : Un ensemble GEN_MC d'éléments ayant comme spécialisant commun E_j

1 $GEN\_MC(\mathcal{P}_{loc}$:$E_j) \Leftarrow \emptyset$
2 **pour tous les** $\mathcal{P}_k$:E_m *généralisant de* $\mathcal{P}_{loc}$:E_j *dans la connaissance de* $\mathcal{P}_{loc}$ **faire**
3 | Add $\mathcal{P}_k$:E_m to $GEN\_MC(\mathcal{P}_{loc}$:$E_j)$
fin

Cet algorithme appliqué à un élément cible $\mathcal{P}_{loc}$:E_j permet de construire l'ensemble $GEN\_MC(\mathcal{P}_{loc}$:$E_j)$ des éléments ayant pour spécialisant $\mathcal{P}_{loc}$:E_j en exploitant uniquement les connaissances locales (cf. ligne 2).

Cet ensemble permettra, ultérieurement, de générer un ensemble de couples d'éléments *pertinents* à aligner de la forme ($\mathcal{P}_{loc}$:E_m, $\mathcal{P}_{dist \neq loc}$:$E_k$) avec $\mathcal{P}_{loc}$:E_m et $\mathcal{P}_{dist}$:$E_k \in GEN\_MC(\mathcal{P}_{loc}$:$E_j)$.

Ces deux algorithmes permettent de construire des ensembles d'éléments partageant un point commun. Dans les deux algorithmes, les ensembles construits sont indexés par l'élément cible qui a permis de les construire. Cependant, il est pertinent de connaître, pour un élément donné, tous les éléments distants avec lesquels il partage un contexte d'interprétation commun quel qu'il soit (généralisant ou spécialisant). L'algorithme 6 réunit les algorithmes 4 et 5 afin de regrouper tous les éléments ayant un contexte d'interprétation commun avec un élément E_j dans un même ensemble MC(E_j).

Algorithme 6 : MC($\mathcal{P}_{loc}$:E_j)

Entrées : Un élément cible E_j d'un pair $\mathcal{P}_{loc}$
Sorties : Un ensemble MC d'éléments qu'il est *pertinent* d'aligner avec E_j

1 $MC(\mathcal{P}_{loc}{:}E_j) \Leftarrow \emptyset$
2 **pour tous les** $\mathcal{P}_i{:}E_m$ *élément plus spécifique que* $\mathcal{P}_{loc}{:}E_j$ **faire**
3 | **pour tous les** $\mathcal{P}_{dist\neq loc}{:}E_k$ *élément plus général que* $\mathcal{P}_{loc}{:}E_m$ *dans les connaissances de* $\mathcal{P}_{loc}$ **faire**
4 | | Add $\mathcal{P}_dist{:}E_k$ to $MC(\mathcal{P}_{loc}{:}E_j)$
| | **fin**
| **fin**
5 **pour tous les** $\mathcal{P}_i{:}E_m$ *élément plus général que* $\mathcal{P}_{loc}{:}E_j$ **faire**
6 | **pour tous les** $\mathcal{P}_{dist\neq loc}{:}E_k$ *de Rewrite*($\mathcal{P}_i{:}E_m$) **faire**
7 | | Add $\mathcal{P}_{dist}{:}E_k$ to $MC(\mathcal{P}_{loc}{:}E_i)$
| | **fin**
| **fin**

On retrouve, dans l'algorithme 6, les algorithmes 4 et 5 respectivement aux lignes 3 à 4 et 5 à 7. L'algorithme 5 est encapsulé, ligne 2, dans une boucle pour obtenir tous les éléments ayant un spécialisant commun avec l'élément E_j pour lequel on construit l'ensemble MC(E_j). En effet, l'algorithme 5 permet d'obtenir l'ensemble des éléments partageant un même spécialisant donné en paramètre. Pour obtenir tous les éléments ayant un spécialisant commun avec E_j, on applique l'algorithme 5 à chaque spécialisant de E_j. Comme l'algorithme 5 utilise la connaissance du pair, il est nécessaire qu'il ne soit appliqué que sur les spécialisants locaux de E_j. Il n'est pas nécessaire d'encapsuler l'algorithme 4 qui permet d'obtenir l'ensemble des éléments ayant un généralisant commun avec un élément passé en paramètre.

Notons que l'algorithme 6 ne se contente pas d'encapsuler les algorithmes 4 et 5. L'ensemble MC(E_j) construit par cet algorithme contient tous les éléments qui partagent un généralisant ou un spécialisant avec E_j. Or, cet ensemble est construit dans le but de trouver de nouveaux mappings. Il n'est donc pas pertinent de conserver les éléments appartenant au même vocabulaire que E_j. On vérifie alors l'origine de chaque élément rapproché aux lignes 3 et 6 pour ne garder que des éléments d'un vocabulaire autre que celui auquel appartient E_j.

Le contexte d'interprétation permet, comme nous l'avons montré, une sélection de couples d'éléments entre lesquels la recherche de mapping est *pertinente*. Dans cette section, nous avons présenté une version abstraite de la notion de contexte d'interprétation commun. Cette description correspond à la réalité tant que les ontologies des pairs sont composées uniquement de classes ou uniquement de propriétés. Lorsque les ontologies comportent à la fois des classes et des propriétés, il est possible de détecter des couples d'éléments pour lesquels la recherche de mapping n'est pas pertinente. La méthode générale présentée reste valable mais elle doit être complétée pour détecter les couples d'éléments qui ne pourront pas se traduire par un mapping du fait du formalisme utilisé dans SomeRDFS. Dans la section suivante, nous présentons comment réaliser cette identification en prenant en compte les contraintes du formalisme RDF(S).

4.2.2 Prise en compte des contraintes du formalisme RDF(S)

Il est *pertinent* de rechercher des mappings, comme nous l'avons présenté dans la section précédente, entre des éléments ayant un contexte d'interprétation commun. On s'assure ainsi qu'il existe un lien sémantique entre les éléments rapprochés. Dans le meilleur des cas, il existe, entre deux éléments rapprochés, un lien sémantique jusqu'alors non représenté. Par exemple, dans la figure 4.3, les éléments *Compositeur* et *Musicien* sont rapprochés du fait de leur généralisant commun *Artist*. Or, nous savons, de manière informelle, que *Compositeur* est un spécialisant de *Musicien*. L'alignement de ces deux éléments, dont le processus est décrit dans le chapitre 5, doit alors permettre de trouver le mapping $Compositeur(X) \Rightarrow Musicien(X)$.

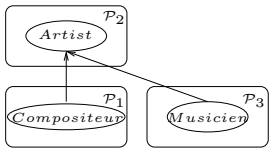

FIG. 4.3 – Exemple de contexte d'interprétation commun pertinent

Cependant, lorsque les ontologies des pairs sont constituées à la fois de classes et de propriétés, la méthode précédemment décrite peut conduire à rapprocher des éléments bien qu'il soit impossible de trouver un mapping entre eux. Par exemple, dans la figure 4.4, les éléments *compose_par* et *a_compose* seraient proposés à un rapprochement du fait de leur généralisant commun *Musicien*. Cependant, ces deux éléments sont des propriétés inverses. Il n'est pas possible de traduire ce lien sémantique par un mapping dans SomeRDFS qui ne peut représenter qu'un lien de subsomption, un typage de domaine ou un typage de portée.

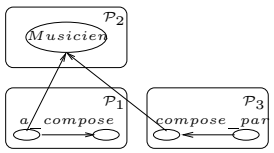

FIG. 4.4 – Exemple de contexte d'interprétation commun non pertinent

La confusion des différents types de liens de spécialisation (subsomption, typage de domaine et typage de portée) est à l'origine de la perte de précision de la méthode exploitant les contextes d'interprétation. Pour y remédier, il est nécessaire d'étudier plus précisément les liens entre les différents éléments mis en jeu. Pour construire les couples d'éléments à aligner, on s'appuie sur les deux scénarios présentés en 4.2.1 et correspondant aux cas où le contexte d'interprétation est soit un généralisant soit un spécialisant commun. Dans les deux cas, on manipule trois éléments qui, chacun, peuvent être soit des classes (relations unaires), soit des propriétés (relations binaires). On peut donc théoriquement distinguer 16 variantes différentes à partir des deux scénarios de départ. Les résultats de l'étude de ces 16 variantes sont résumés dans le tableau 4.1.

E_j, l'élément local	E_k, l'élément distant	E_m, le point commun entre E_j et E_k	Conclusion
classe	classe	classe généralisante	OK
classe	classe	classe spécialisante	OK
classe	classe	propriété généralisante	cas impossible
classe	classe	propriété spécialisante	OK si : • 2 typages de domaine • 2 typages de portée
classe	propriété	classe généralisante	OK • mapping de typage de domaine si E_m type le domaine de E_k • mapping de typage de portée si E_m type la portée de E_k
classe	propriété	classe spécialisante	cas impossible
classe	propriété	propriété généralisante	cas impossible
classe	propriété	propriété spécialisante	OK • mapping de typage de domaine si E_j type le domaine de E_m • mapping de typage de portée si E_j type la portée de E_m
propriété	classe	classe généralisante	OK • mapping de typage de domaine si E_m type le domaine de E_j • mapping de typage de portée si E_m type la portée de E_j
propriété	classe	classe spécialisante	cas impossible
propriété	classe	propriété généralisante	cas impossible
propriété	classe	propriété spécialisante	OK • mapping de typage de domaine si E_k type le domaine de E_m • mapping de typage de portée si E_k type la portée de E_m

E_j, l'élément local	E_k, l'élément distant	E_m, le point commun entre E_j et E_k	Conclusion
propriété	propriété	classe généralisante	OK si : • E_m type les domaines de E_j et de E_k et que $\exists E_n$ type les portées de E_j et de E_k • E_m type les portées de E_j et de E_k et que $\exists E_n$ type les domaines de E_j et de E_k
propriété	propriété	classe spécialisant	cas impossible
propriété	propriété	propriété généralisante	OK
propriété	propriété	propriété spécialisante	OK

TAB. 4.1: Synthèse de l'étude sur les contextes d'interprétation communs

On distingue plusieurs catégories de conclusions :

- OK sans condition. Ces combinaisons de types d'éléments n'imposent aucune contrainte sur les mappings associés. On obtient cette conclusion pour toutes les variantes dans lesquelles tous les éléments sont du même type (classes ou propriétés). Ces variantes correspondent exactement aux cas décrits dans la section précédente. La figure 4.3 correspond à un exemple basé sur le généralisant commun et dont tous les éléments sont des classes.
- OK avec condition. Ces combinaisons de types d'éléments imposent des contraintes sur l'orientation des mappings associés. Il peut s'agir de contraintes dues au formalisme (orientation du mapping) ou de contraintes dues à la faible probabilité que les mappings puissent être pertinents. Par exemple, le formalisme impose qu'un mapping liant une classe et une propriété soit uniquement un typage de domaine ou de portée. Dans les deux cas, il s'agit d'une spécialisation de la classe par la propriété. Lorsque les deux éléments E_1 et E_2 rapprochés sont respectivement une classe et une propriété, il est ainsi inutile de rechercher un mapping représentant une spécialisation de E_2 par E_1. Dans le cas où une propriété est le spécialisant commun des classes typant respectivement son domaine et sa portée, des mappings de subsomption peuvent être représentés entre ces deux classes. Cependant il est peu probable qu'il existe un lien de subsomption entre une classe typant le domaine et une autre classe typant la portée d'une même propriété. Lorsque deux classes partagent une propriété comme spécialisant commun, ces deux classes ne seront rapprochées que si elles typent toutes les deux le domaine ou toutes les deux la portée de la même propriété.
- Cas impossible. Ces combinaisons de types d'éléments ne sont pas permises par le formalisme RDF(S). Par exemple, une propriété ne peut être un généralisant d'une classe, elle ne peut donc pas être un généralisant commun d'une classe et d'un autre élément.

L'algorithme 6 doit être adapté pour prendre en compte les types des éléments mis en jeu et les contraintes supplémentaires qui s'appliquent aux différentes variantes. Ainsi, plusieurs tests sont ajoutés. Un test sur les types des éléments mis en jeu permet de déterminer si la recherche de mappings entre les deux éléments rapprochés est pertinente ou non. Un autre test permet de déterminer quel(s) type(s) de mapping(s) (généralisation, spécialisation, typage de domaine ou typage de portée) il est pertinent de rechercher. L'algorithme 7 correspond à l'ajout de ces tests dans l'algorithme 6.

Algorithme 7 : MC($\mathcal{P}_{loc}$:E_j) avec liaison de variables

Entrées : Un élément cible, E_j du pair $\mathcal{P}_{loc}$
Sorties : L'ensemble des éléments qu'il est pertinent d'aligner avec E_j

```
1  MC(P_loc:E_j) ⇐ ∅
2  pour tous les P_loc:E_m spécialisant direct de P_loc:E_j faire
3  |  pour tous les P_dist≠loc:E_k généralisant direct de P_loc:E_m faire
4  |  |  si isReallyCommon( P_loc:E_j, P_dist:E_k, P_loc:E_m) alors
5  |  |  |  AddToMC(P_n:E_k, P_loc:E_j, P_loc:E_m)
   |  |  fin
   |  fin
   fin
6  pour tous les P_i:E_m généralisant de P_loc:E_j faire
7  |  pour tous les P_dist≠loc:E_k de Rewrite(P_i:E_m) faire
8  |  |  si isReallyCommon( P_loc:E_j, P_dist:E_k, P_i:E_m) alors
9  |  |  |  AddToMC (P_dist:E_k, P_loc:E_j, P_i:E_m)
   |  |  fin
   |  fin
   fin
```

Le test ajouté aux lignes 4 et 8 conduit à ce que seuls les éléments pour lesquels il est pertinent de rechercher un mapping avec l'élément cible soient ajoutés à l'ensemble $MC(\mathcal{P}_{loc}{:}E_j)$. Pour ce test, on suppose que les ontologies et les mappings sont bien formés. Cette hypothèse nous permet de ne pas avoir à traiter les cas dits impossibles. Ce test vérifie les types des trois éléments E_j, E_k et E_m et retourne vrai si la recherche d'un mapping entre E_j et E_k est pertinente (lorsque la conclusion de l'étude est OK sans condition ou OK avec condition et que la condition est vérifiée) et retourne faux sinon (lorsque la conclusion de l'étude est OK avec condition et que la condition n'est pas vérifiée). Nous supprimons également de $MC(\mathcal{P}_{loc}{:}E_j)$ les éléments pour lesquels le mapping recherché existe déjà ou peut être déduit dans les connaissances du pair local.

Par ailleurs, pour permettre de préciser quel mapping doit être recherché entre l'élément cible E_j et un élément de $MC(\mathcal{P}_{loc}{:}E_j)$, chaque ensemble $MC(\mathcal{P}_{loc}{:}E_j)$ est partitionné en quatre sous-ensembles $MC_{SUP}(\mathcal{P}_{loc}{:}E_j)$, $MC_{SUB}(\mathcal{P}_{loc}{:}E_j)$, $MC_{DOM}(\mathcal{P}_{loc}{:}E_j)$ et $MC_{RAN}(\mathcal{P}_{loc}{:}E_j)$, chacun de ces sous-ensembles correspondant à un type de mapping (généralisation, spécialisation, typage de domaine et typage de portée). La procédure *AddToMC* utilisée aux lignes 5 et 9 permet de construire ces quatre sous-ensembles en suivant les conclusions du tableau 4.1. Ainsi, lorsque les deux éléments rapprochés sont du même type, le mapping recherché est une subsomption et lorsque les deux éléments sont de types différents, le mapping recherché est un typage. La nature du typage (domaine ou

portée) est donné par le contexte d'interprétation. La procédure d'ajout comporte également un test ayant pour but de vérifier que le mapping étudié n'est pas inférable à partir des connaissances propres du pair. Ce test est effectué en toute fin de l'algorithme 7 afin de permettre de transformer un mapping de subsomption en un mapping d'équivalence. Les algorithmes *isReallyCommon* et *AddToMC* sont détaillés dans l'annexe A.

4.3 Critères supplémentaires

Nous avons proposé, dans la section précédente, une méthode de sélection des couples d'éléments *pertinents* à aligner. Cependant, lorsque les pairs du système sont nombreux et qu'ils partagent de nombreux mappings, cette sélection peut être insuffisante. Dans ce cas, il est nécessaire de choisir parmi les couples d'éléments construits par notre méthode, ceux qui seront finalement à aligner. Ce choix est arbitraire car tous les couples sélectionnés ont, a priori, la même pertinence. Toutefois, bien qu'arbitraire, ce choix peut avoir pour objectif la réalisation d'une des stratégies présentées en 2.2. En particulier, il peut être intéressant de ne sélectionner que les couples d'éléments conduisant à la découverte de mappings avec certains pairs. Il peut également être intéressant de se focaliser uniquement sur certains types de mappings. Enfin, l'ajout d'un mapping peut modifier les connaissances du système pair-à-pair. Il est intéressant d'anticiper ces modifications en choisissant de n'ajouter que les mappings qui modifient les connaissances du système ou, au contraire, uniquement les mappings qui ne modifient pas les connaissances du système. La stratégie à réaliser est choisie, pour chaque pair, par son administrateur. L'ajout de critères de sélection supplémentaires s'applique aux différentes étapes du processus de sélection des couples d'éléments qu'il est pertinent d'aligner.

Dans ce qui suit, nous présentons la mise en œuvre des différentes stratégies présentées en 2.2. Tout d'abord, nous nous intéressons aux stratégies visant la découverte de mappings uniquement avec certains pairs. Dans un deuxième temps, nous proposons de sélectionner les couples d'éléments *pertinents* à aligner en fonction d'un type précis de mapping recherché. Enfin, nous proposons d'adapter la sélection des couples d'éléments *pertinents* à aligner à la qualité de la représentation des connaissances du pair.

4.3.1 Stratégies relatives aux pairs avec lesquels des mappings sont recherchés

Nous avons présenté, en 2.2.1, différentes stratégies relatives aux pairs avec lesquels des mappings sont recherchés. Ces stratégies consistent à ne considérer qu'un sous-ensemble des pairs du système pour la rechercher de mappings. Ces pairs particuliers sont appelés pairs d'intérêt (c.f. 2.2.1). Appliquée à la méthode décrite dans ce chapitre, les stratégies modifient d'une part l'ensemble des éléments cibles et, d'autre part, l'ensemble des couples d'éléments *pertinents* à aligner obtenu par exploitation des contextes d'interprétation d'un élément donné comme le montre la figure 4.5.

Concrètement, l'application d'une stratégie de cette nature se traduit par l'ajout d'une étape d'identification des pairs d'intérêt. Cette étape peut être effectuée à différents moments du processus. Cependant, les modifications apportées au processus de sélection des éléments à aligner diffèrent selon le moment où cette étape est effectuée. Nous proposons

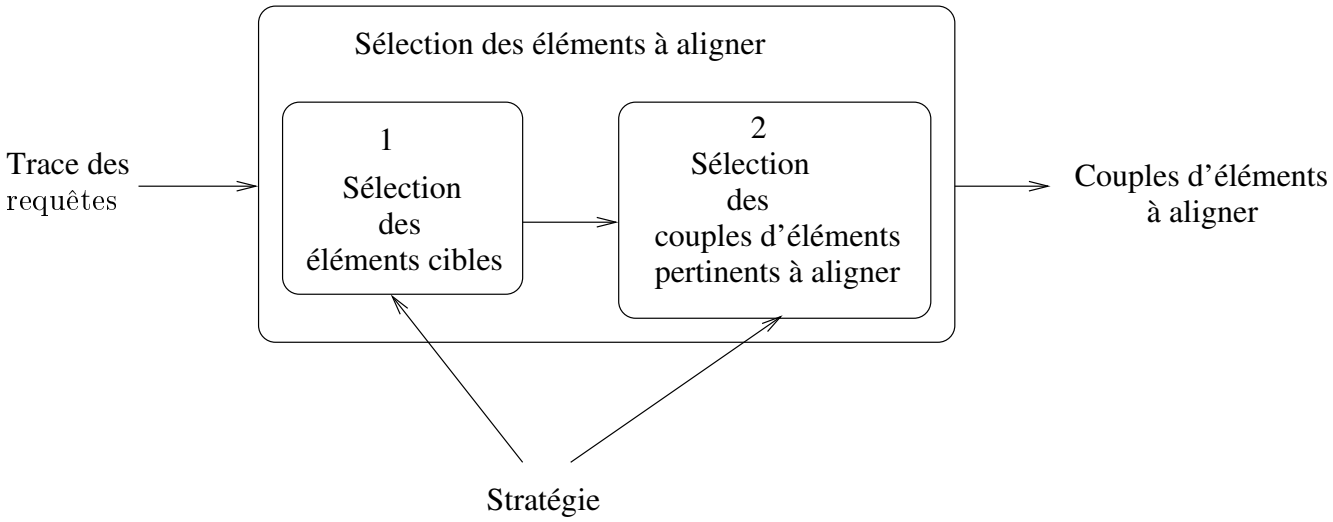

FIG. 4.5 – Décomposition du processus de sélection des éléments à aligner

donc deux méthodes qui correspondent, respectivement, aux modifications apportées au processus dans le cas où l'identification des pairs d'intérêt est réalisée avant la sélection des éléments cibles et dans le cas où la sélection des éléments cibles est réalisée avant l'identification des pairs d'intérêt.

Première méthode

La première méthode suppose les pairs d'intérêt préalablement connus. Leur identification peut être réalisée par l'administrateur ou, pour les stratégies intentionnelles, par un processus indépendant ayant analysé les connaissances du pair local. Les points de blocage sont alors définis par rapport à ce sous-ensemble des pairs du système. Dans un deuxième temps, les couples d'éléments qu'il est *pertinent* d'aligner sont filtrés pour ne retenir que ceux comportant un élément d'un pair d'intérêt.

Seconde méthode

La seconde méthode suppose que les pairs d'intérêt ne sont pas préalablement connus. Leur identification constitue une des étapes de cette méthode qui fait dépendre les points de blocage directement de la stratégie choisie. Selon la stratégie choisie, les points de blocage sont caractérisés différemment.

Cas de la stratégie par défaut : les points de blocages sont des éléments n'ayant pas assez de spécialisants appartenant aux ontologies de pairs distants. Il est donc nécessaire de connaître le nombre de ces spécialisants.

Cas de la stratégie de rupture de l'isolement : les points de blocages sont des éléments dont les spécialisants appartiennent à un nombre réduit d'ontologies de pairs différents. Il est donc nécessaire de connaître le nombre de pairs distincts qui se partagent ces spécialisants.

Cas de la stratégie rapprochement : les points de blocage sont des éléments dont les spécialisants appartiennent à un grand nombre d'ontologies de pairs différents. Il est donc nécessaire de connaître le nombre de ces spécialisants ainsi que le nombre de pairs distincts qui se partagent ces spécialisants. On s'intéresse alors à une mesure prenant en compte ces deux nombres.

Pour permettre cette diversité de caractérisation, nous proposons une nouvelle définition des points de blocage qui intègre une fonction de comptage définie spécifiquement pour chaque stratégie et qui consiste à calculer soit le nombre de spécialisants soit le nombre de pairs distincts qui se partagent ces spécialisants, soit le nombre de spécialisants par pair distinct. Cette nouvelle définition est une généralisation de celle donnée en 4.1 qui correspond, en fait, à la définition des points de blocage pour la stratégie par défaut.

Définition 20 (point de blocage stratégique)
Soient $\mathcal{P}$:E un élément du pair $\mathcal{P}$, t un seuil défini par l'administrateur du pair et f une fonction de comptage. $\mathcal{P}$:E est un point de blocage stratégique si $f(\mathcal{P}:E) < t$.

Notons que la définition des pairs d'intérêt selon les différentes stratégies peut se faire par rapport à deux types de connaissances : les connaissances locales ou les connaissances obtenues en exploitant les mécanismes de raisonnement mis en œuvre dans SomeRDFS. Les fonctions de comptage permettant de définir les points de blocages traduisent des stratégies. Elles doivent donc également tenir compte de cette distinction. Dans le tableau 4.2, nous proposons une définition de la fonction de comptage pour l'élément E_1 du pair $\mathcal{P}_1$ par rapport à la stratégie choisie pour ce pair et la méthode de comptage utilisée (prise en compte uniquement des connaissances locales ou également des connaissances obtenues via les mécanismes de raisonnement mis en œuvre dans SomeRDFS).

Reprenons les trois stratégies mentionnées précédemment pour un élément $\mathcal{P}_1$:E_1 donné et décrivons la fonction de comptage selon la stratégie et la méthode de comptage utilisée :

- Stratégie par défaut : la fonction de comptage évalue le nombre de spécialisants de $\mathcal{P}_1$:E_1. Lorsque cette fonction de comptage exploite les connaissances locales, il s'agit du nombre de mappings de la forme $E_i \Rightarrow E_j$ où E_j est soit $\mathcal{P}_1$:E_1 soit un spécialisant local de $\mathcal{P}_1$:E_1. Lorsque la fonction de comptage exploite les mécanismes de raisonnement mis en œuvre dans SomeRDFS, il s'agit du nombre d'éléments obtenus comme réécriture de la requête $Q \equiv \mathcal{P}_1$:E_1.

- Stratégie de rupture de l'isolement : la fonction de comptage évalue le nombre de pairs distincts se partageant les spécialisants de $\mathcal{P}_1$:E_1. Ces spécialisants sont obtenus comme décrit pour la stratégie par défaut. Ils sont ensuite indexés par le pair possédant l'ontologie dont ils sont issus. Le résultat de la fonction de comptage est

le nombre d'indices utilisés.

- Stratégie de rapprochement : la fonction de comptage évalue, pour chaque pair, le nombre de spécialisants de $\mathcal{P}_1$:E_1 que son ontologie contient. Ces spécialisants sont obtenus et indexés de la même façon que pour la stratégie de rupture de l'isolement. La mesure prenant en compte le nombre de spécialisants et le nombre de pairs se partageant ces spécialisants est comparée au seuil donné par l'administrateur. Dans le tableau 4.2, nous proposons deux mesures. Étant donnés n pairs se partageant les spécialisants de $\mathcal{P}_1$:E_1, nous avons un nombre de spécialisants de $\mathcal{P}_1$:E_1 par pair donc n nombres. La première mesure compare le plus petit nombre au seuil donné par l'administrateur et la seconde mesure compare le plus grand nombre à ce seuil. Ces deux mesures correspondent chacune à une sous-stratégie différente. En prenant la valeur maximale, un point de blocage est défini comme un élément pour lequel aucun pair n'a suffisamment de spécialisants de $\mathcal{P}_1$:E_1 dans son ontologie. Par exemple, si $\mathcal{P}_1$:E_1 possède 6 spécialisants répartis comme suit entre trois pairs : $\mathcal{P}_2$ (1 spécialisant), $\mathcal{P}_3$ (2 spécialisants) et $\mathcal{P}_4$ (3 spécialisants). Pour que $\mathcal{P}_1$:E_1 soit un point de blocage, il est nécessaire que le seuil défini par l'administrateur soit supérieur à 3. Dans ce cas, aucun des trois pairs $\mathcal{P}_2$, $\mathcal{P}_3$ ou $\mathcal{P}_4$ ne possède suffisamment de spécialisants de $\mathcal{P}_1$:E_1. Au contraire, en prenant la valeur minimale, un point de blocage est défini comme un élément pour lequel au moins un pair n'a pas suffisamment de spécialisants de $\mathcal{P}_1$:E_1 dans son ontologie. Sur le même exemple, $\mathcal{P}_1$:E_1 est un point de blocage car le seuil défini par l'administrateur est nécessairement supérieur à 1 (la stratégie de rapprochement ne s'intéresse qu'aux pairs distants avec lesquels un lien sémantique est déjà partagé). Avec un seuil de 1, $\mathcal{P}_2$ n'a pas suffisamment de spécialisants de $\mathcal{P}_1$:E_1 mais $\mathcal{P}_3$ et $\mathcal{P}_4$ en possèdent suffisamment.

Nous avons défini les points de blocage relativement aux stratégies sans avoir recours aux pairs d'intérêt. Cependant, l'application d'une stratégie nécessite d'identifier les pairs d'intérêt pour la réalisation de l'étape 2 du processus de sélection des éléments à aligner. Pour être sûr de la cohérence entre les différentes étapes du processus de sélection des couples d'éléments à aligner, les pairs d'intérêt et les points de blocage doivent être définis l'un par rapport à l'autre. Dans cette méthode, les pairs d'intérêt sont donc identifiés à partir des points de blocage. L'ensemble des pairs d'intérêt ne peut alors pas être global car il est obtenu uniquement à partir des pairs d'intérêt et non à partir de l'ensemble des connaissances du pair. Cet ensemble est donc nécessairement local ou hybride comme nous le décrivons en 2.2.1.

Dans le cas particulier des éléments cibles non élémentaires, on recherche également l'ensemble hybride des pairs d'intérêt obtenu pour les éléments composant une requête posée par un utilisateur. L'identification des éléments cibles non élémentaires étant liée à un manque (probable ou révélé) de réponses à des requêtes, les pairs d'intérêt spécifiques identifiés localement à partir des stratégies ne correspondent alors pas exactement aux pairs recherchés. Lorsque l'identification des éléments cibles fait suite à une requête conjonctive Q, nous avons supposé que le manque de réponses était dû à un manque de réécritures homogènes de Q. L'ajout de mappings doit alors permettre de trouver au

	C_1 (par rapport à la connaissance de $\mathcal{P}_1$)	C_2 (basé sur les réécritures)
Stratégie par défaut	$\lvert\{\mathcal{P}_{i\neq1}{:}E_j$ / $[\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_1]$ ou $[\exists\mathcal{P}_1{:}E_k$ tel que $\mathcal{P}_1{:}E_k \Rightarrow \mathcal{P}_1{:}E_1$ peut être inféré et $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_k]\}\rvert$	$\lvert\{\mathcal{P}_{i\neq1}{:}E_j \in RW\}\rvert$ où RW est l'ensemble des réécritures de la requête $Q \equiv \mathcal{P}_1{:}E_1$
Stratégie de rupture de l'isolement	$\lvert\{\mathcal{P}_{i\neq1}$ / $\exists\mathcal{P}_i{:}E_j$ tel que $[\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_1]$ ou $[\exists\mathcal{P}_1{:}E_k$ tel que $\mathcal{P}_1{:}E_k \Rightarrow \mathcal{P}_1{:}E_1$ peut être inféré et $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_k]\}\rvert$	$\lvert\{\mathcal{P}_{i\neq1}$ / $\mathcal{P}_i{:}E_j \in RW\}\rvert$ où RW est l'ensemble des réécritures de la requête $Q \equiv \mathcal{P}_1{:}E_1$
Stratégie de rapprochement 1	$\min_i(\lvert\{\mathcal{P}_{i\neq1}{:}E_j$ / $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_1$ ou $[\exists\mathcal{P}_1{:}E_{k\neq1})$ tel que $\mathcal{P}_1{:}E_k \Rightarrow \mathcal{P}_1{:}E_1$ peut être inféré et $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_k]\}\rvert)$	$\min_i \lvert\{\mathcal{P}_{i\neq1}{:}E_j \in RW\}\rvert$ où RW est l'ensemble des réécritures de la requête $Q \equiv \mathcal{P}_1{:}E_1$
Stratégie de rapprochement 2	$\max_i(\lvert\{\mathcal{P}_{i\neq1}{:}E_j$ / $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_1$ ou $[\exists\mathcal{P}_1{:}E_{k\neq1})$ tel que $\mathcal{P}_1{:}E_k \Rightarrow \mathcal{P}_1{:}E_1$ peut être inféré et $\mathcal{P}_i{:}E_j \Rightarrow \mathcal{P}_1{:}E_k]\}\rvert)$	$\max_i \lvert\{\mathcal{P}_{i\neq1}{:}E_j \in RW\}\rvert$ où RW est l'ensemble des réécritures de la requête $Q \equiv \mathcal{P}_1{:}E_1$

TAB. 4.2 – Définition de f($\mathcal{P}_1{:}E_1$)

moins une nouvelle réécriture homogène utilisant les éléments de l'ontologie d'un pair $\mathcal{P}_i$ donné. L'ajout de mappings doit alors permettre de trouver, pour chaque élément de Q, au moins un spécialisant appartenant à l'ontologie de $\mathcal{P}_i$. Si les mappings ajoutés représentent des liens avec l'ontologie de $\mathcal{P}_i$, on est assuré qu'ils permettent de trouver au moins une nouvelle réécriture homogène. Dans le meilleur des cas, il existe dans l'ontologie de $\mathcal{P}_i$, des éléments représentés comme spécialisants pour chaque les éléments de Q sauf un. Il est alors nécessaire de trouver un seul mapping. Dans le pire des cas, aucun élément de $\mathcal{P}_i$ n'est représenté comme un spécialisant d'un élément de Q. Il est donc nécessaire de trouver un mapping de spécialisation pour chacun des éléments de Q. Notre objectif est de limiter le nombre de couples d'éléments à aligner. Un pair $\mathcal{P}$ ne peut être un pair d'intérêt spécifique pour les éléments de Q que s'il participe à au moins une réécriture de Q.

Enfin, comme pour la première méthode, les couples d'éléments qu'il est *pertinent* d'aligner sont filtrés dans l'étape 2 du processus de sélection des éléments à aligner pour ne retenir que ceux comportant un élément d'un pair d'intérêt.

4.3.2 Stratégie sur le type de mappings recherchés

Nous avons présenté, en 2.2.2, différentes stratégies consistant à rechercher uniquement des mappings de spécialisation ou uniquement des mappings de généralisation. Par défaut, on s'intéresse aux deux types de mappings. Cependant, lorsque le nombre d'éléments à comparer est trop important, le choix d'une de ces deux stratégies permet de réduire considérablement le nombre d'implications à vérifier. Dans la mesure où les mécanismes de raisonnement mis en œuvre dans SomeRDFS exploitent uniquement les mappings de spécialisation, la stratégie orientée mappings de spécialisation est la plus in-

diquée pour obtenir un enrichissement immédiat des réponses aux requêtes. Cependant, il est possible que la recherche de spécialisants distants ne puisse aboutir en l'état des connaissances d'un pair. Un point de blocage pour lequel aucun mapping de spécialisation n'est trouvé sera de nouveau identifié comme un point de blocage lors d'une future recherche de mappings. À défaut de lever le blocage, la découverte d'un nouveau mapping de généralisation constitue un nouveau contexte d'interprétation qui pourra être exploité lors de cette future recherche de mappings puisque la méthode que nous proposons exploite également les mappings de généralisation comme nous le décrivons en section 4.2. De plus, les mappings de généralisation, s'ils ne permettent pas des réponses plus riches aux requêtes, permettent d'enrichir le vocabulaire du pair local. Ils facilitent alors la tâche d'alignement en permettant une meilleure connaissance des ontologies des pairs distants. Nous présentons successivement la mise en œuvre des stratégies orientée mappings de spécialisation et orientée mappings de généralisation.

Application de la stratégie orientée mappings de spécialisation

Les éléments cibles sont caractérisés par un manque de mappings de spécialisation. Ce sont donc des mappings de spécialisation des éléments cibles qui intéressent l'administrateur. On a vu, en 4.2.2, que lorsque les deux éléments à aligner ne sont pas du même type, alors seul un sens est possible pour ce mapping. En particulier, lorsque l'élément cible est une propriété, si on trouve un mapping, il s'agit d'un mapping de généralisation. Or, seuls les mappings de spécialisation sont recherchés. Il n'est donc pas nécessaire d'aligner ces deux éléments. L'algorithme 8 a pour but de limiter la recherche aux mappings de spécialisation.

Algorithme 8 : MoreSpecificFilter($\mathcal{P}_{loc}$:E_j)

Entrées : Un élément E_j, du pair $\mathcal{P}_{loc}$.
Résultat : Suppression des éléments associés à $\mathcal{P}_{loc}$:E_j pour lesquels il ne peut y avoir de mapping de spécialisation.
$MC\_Gen(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$
si $\mathcal{P}_{loc}$:E_j *est une propriété* **alors**
| $filtered\_MC\_DOM(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$ $filtered\_MC\_RAN(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$
fin

Nous avons construit, pour chaque élément cible $\mathcal{P}_{loc}$:E_j, quatre ensembles d'éléments qu'il est pertinent d'aligner avec $\mathcal{P}_{loc}$:E_j correspondant chacun à type de mapping. Cet algorithme vide les ensembles ne correspondant pas à des mappings de spécialisation. Il s'agit de l'ensemble MC_Gen pour tous les éléments cibles mais également les ensembles filtered_MC_DOM et filtered_MC_RAN des propriétés cibles. En effet, pour les propriétés cibles, ces deux ensembles ne contiennent que des classes distantes. Le seul mapping pouvant être représenté est le typage du domaine ou de la portée de la propriété par la classe. Un tel lien correspond à un mapping de généralisation.

Application de la stratégie orientée mappings de généralisation

La méthode décrite dans ce chapitre exploite non seulement les mécanismes de raisonnement mis en œuvre dans SomeRDFS mais également les connaissances du pair local. En particulier, cette méthode exploite les mappings de généralisation. Trouver de nouveaux mappings de généralisation ne permet pas de pallier immédiatement le manque de mappings de spécialisation des éléments cibles mais cela permet de trouver des nouveaux contextes d'interprétation pour les éléments cibles. Ces nouveaux contextes d'interprétation permettront de trouver de nouveaux éléments partageant un contexte d'interprétation commun avec les éléments cibles, lors d'une future recherche de mappings. L'algorithme 9 réalise ce filtrage de manière similaire à l'algorithme 8.

Algorithme 9 : MoreGeneralFilter($\mathcal{P}_{loc}$:E_j)

Entrées : Un élément E_j, du pair $\mathcal{P}_{loc}$.
Résultat : Suppression des éléments associés à $\mathcal{P}_{loc}$:E_j pour lesquels il ne peut y avoir de mapping de généralisation.

$MC\_Spec(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$
si $\mathcal{P}_{loc}$:E_j *est une classe* **alors**
  $filtered\_MC\_DOM(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$
  $filtered\_MC\_RAN(\mathcal{P}_{loc}:E_j) \Leftarrow \emptyset$
fin

Les algorithmes 8 et 9 vident les ensembles ne correspondant pas au type de mapping recherché. Ces deux filtres agissent sur des ensembles complémentaires, il doivent donc être utilisés de façon exclusive.

4.3.3 Stratégies sur la qualité des mappings existants

Nous avons présenté, en 2.2.3, différentes stratégies reposant sur l'optimalité des mappings faisant partie des connaissances du pair local. La stratégie d'optimisation des mappings existants a pour hypothèse que les mappings existants ne sont pas optimaux et consiste à rechercher des mappings plus précis. Au contraire, la stratégie de préservation des mappings existants suppose que les mappings existants sont optimaux et consiste à ne pas aligner deux éléments si cet alignement conduit à la proposition d'un mapping dont l'ajout nécessite la suppression d'un mapping existant. Par défaut, aucune hypothèse n'est faite sur l'optimalité des mappings existant et aucun filtrage n'est appliqué.

La méthode proposée dans ce chapitre permet de constituer des couples d'éléments *pertinents* à aligner. L'application de l'une ou l'autre de ces stratégies permet de choisir pour quels couples d'éléments l'alignement sera effectué. Nous détaillons successivement l'application de ces deux stratégies.

Application de la stratégie d'optimisation des mappings existants

Cette stratégie a pour objectif de limiter la recherche aux mappings plus précis que les mappings existants. Étant donné un mapping existant $M \equiv E_1 \Rightarrow E_2$, ce mapping est supposé non optimal, du fait de la stratégie. Si E_2 est un élément local, comme illustré dans la figure 4.6(a), on recherche alors un mapping $M' \equiv E_1 \Rightarrow E_3$ où E_3 est un

spécialisant local de E_2. Si E_1 est un élément local, comme illustré dans la figure 4.6(b), on recherche alors un mapping $M' \equiv E_3 \Rightarrow E_2$ où E_3 est un généralisant local de E_1. Dans les deux cas, la découverte de M' nécessite la suppression de M. Ces deux cas sont symétriques, on ne détaillera donc que le premier.

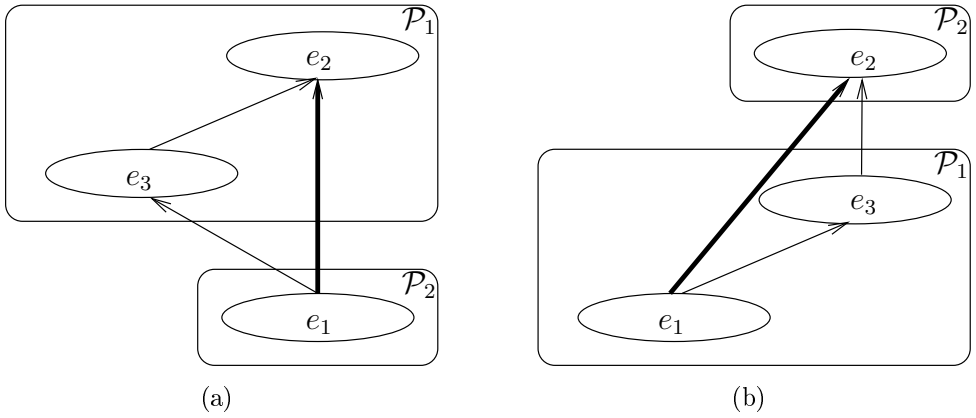

FIG. 4.6 – Scénarios d'optimisation de mappings existants

La méthode présentée dans ce chapitre permet de construire des couples d'éléments (E_3, E_1) ayant un contexte d'interprétation commun. Intéressons nous au cas du généralisant commun E_2. Si E_2 n'est pas un élément de la même ontologie que E_3 comme illustré dans la figure 4.7, $M \equiv E_1 \Rightarrow E_2$ n'est pas un mapping du pair local et ne peut donc pas être considéré comme non optimal par l'administrateur du pair local. Selon cette stratégie, nous ne rechercherons pas de mappings entre E_1 et E_3.

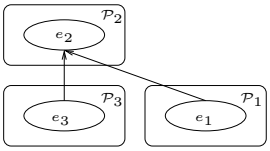

FIG. 4.7 – Généralisant commun distant

Si, au contraire, E_2 est un élément de la même ontologie que E_3 comme illustré dans la figure 4.8 et que, de plus, le lien sémantique entre E_1 et E_2 est représenté par le mapping $M \equiv E_1 \Rightarrow E_2$, alors, selon l'hypothèse de la stratégie d'optimisation des mappings existants, ce mapping est non optimal. L'existence d'un mapping $M' \equiv E_1 \Rightarrow E_3$ doit donc être étudié car E_3 étant un spécialisant local de E_2, M' est un mapping plus précis que M s'il est valide. Par contre, selon cette stratégie, on n'étudiera pas l'existence du mapping $E_3 \Rightarrow E_1$.

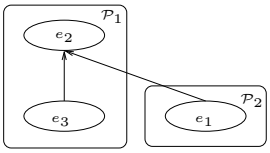

FIG. 4.8 – Généralisant commun local

Par exemple, dans la figure 4.9, $\mathcal{P}_2$:*Pianiste* $\Rightarrow$ $\mathcal{P}_1$:*Artiste* est un mapping de $\mathcal{P}_1$ qui conduit à l'étude du rapprochement de $\mathcal{P}_2$:*Pianiste* et de $\mathcal{P}_1$:*Musicien*.

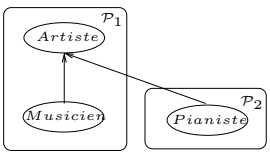

FIG. 4.9 – Exemple de mapping non optimal

L'application de la stratégie d'optimisation des mappings existants nous permet, dans cet exemple, de n'étudier l'existence que du seul mapping $\mathcal{P}_2$:*Pianiste* $\Rightarrow$ $\mathcal{P}_1$:*Musicien*. L'existence du mapping $\mathcal{P}_1$:*Musicien* $\Rightarrow$ $\mathcal{P}_2$:*Pianiste* ne sera pas testée lors de la phase d'alignement. De même, on ne cherchera pas de mappings entre $\mathcal{P}_1$:*Musicien* et les spécialisants de $\mathcal{P}_2$:*Pianiste* s'il n'existe pas de mapping les liant à $\mathcal{P}_1$:*Artiste*.

Application de la stratégie de préservation des mappings existants

Pour cette stratégie, tous les mappings du pair sont considérés comme optimaux. Il est donc inutile de valider un mapping $M \equiv E_1 \Rightarrow E_3$ s'il existe déjà, dans la connaissance du pair, un mapping $M \equiv E_1 \Rightarrow E_2$ et que E_2 est un généralisant de E_3 ou s'il existe déjà, dans la connaissance du pair un mapping $M \equiv E_2 \Rightarrow E_1$ et que E_2 est un spécialisant de E_3.

La méthode présentée dans ce chapitre permet de construire des couples d'éléments (E_3, E_1) ayant un contexte d'interprétation commun. On distingue alors les mêmes cas que pour la stratégie d'optimisation des mappings existants mais les conclusions sont différentes. Lorsque le généralisant commun n'appartient pas à l'ontologie locale (cf. figure 4.7), l'existence des mappings $E_3 \Rightarrow E_1$ et $E_1 \Rightarrow E_3$ doivent être étudiées car l'ajout de ces mappings ne remet en question l'optimalité d'aucun mapping existant. Lorsque le généralisant commun appartient à l'ontologie locale (cf. figure 4.8), l'existence du mapping $E_3 \Rightarrow E_1$ doit être étudiée mais pas celle du mapping $E_1 \Rightarrow E_3$. En effet, si $E_1 \Rightarrow E_3$ est un mapping valide, alors $E_1 \Rightarrow E_2$ est non optimal, ce qui contredit l'hypothèse de la stratégie.

Par exemple, dans la figure 4.10, $\mathcal{P}_2$:*Musicien* $\Rightarrow$ $\mathcal{P}_1$:*Artiste* est un mapping de $\mathcal{P}_1$ qui permet d'étudier le rapprochement de $\mathcal{P}_1$:*Pianiste* et de $\mathcal{P}_2$:*Musicien*.

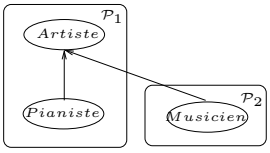

FIG. 4.10 – Exemple de mapping optimal

L'application de la stratégie de préservation des mappings existants nous permet, dans cet exemple, de ne pas chercher à étudier la validité du mapping $\mathcal{P}_2$:*Musicien* $\Rightarrow$ $\mathcal{P}_1$:*Pianiste*. En revanche, la phase d'alignement devra valider le mapping $\mathcal{P}_1$:*Pianiste* $\Rightarrow$ $\mathcal{P}_2$:*Musicien*, ainsi que tous les mappings entre $\mathcal{P}_1$:*Pianiste* et un spécialisant de $\mathcal{P}_2$:*Musicien* pour lesquels il n'existe pas de mapping avec $\mathcal{P}_1$:*Artiste*.

4.3.4 Vérification de l'extension conservative

Nous avons présenté, en 2.2.3, trois stratégies relatives au phénomène d'émergence de connaissances dans un système pair-à-pair connu sous le nom d'extension non conservative (cf. figure 4.11). Si la stratégie choisie par l'administrateur est la stratégie d'enrichissement de l'ontologie, toute extension non conservative est acceptée et intégrée à l'ontologie locale. Sur notre exemple, cela signifie que le lien ontologique $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_1$:*Animal* déduit des mappings $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_2$:*Adulte* et $\mathcal{P}_2$:*Adulte* $\Rightarrow$ $\mathcal{P}_1$:*Animal* doit être ajouté. Au contraire, si l'administrateur a choisi la stratégie de préservation de l'ontologie, toute extension non conservative est refusée. Ce refus a pour conséquence la suppression de certains des mappings ayant provoqué l'extension non conservative. La stratégie par défaut consiste à choisir la résolution individuellement pour chaque extension non conservative. Sur notre exemple, le lien $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_1$:*Animal* ne doit pas être ajouté, il est donc nécessaire de supprimer le mapping $\mathcal{P}_1$:*Homme* $\Rightarrow$ $\mathcal{P}_2$:*Adulte* ou le mapping $\mathcal{P}_2$:*Adulte* $\Rightarrow$ $\mathcal{P}_1$:*Animal*. Par défaut, l'administrateur choisit individuellement d'accepter l'extension conservative ou, s'il la refuse, les mappings à supprimer.

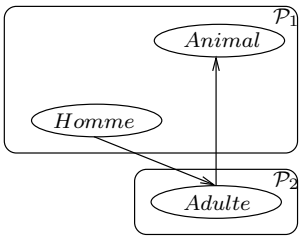

FIG. 4.11 – Exemple d'extension non conservative

Dans le cas qui nous intéresse, c'est l'ajout d'un mapping qui provoque le phénomène d'extension non conservative. La stratégie d'enrichissement de l'ontologie apporte une solution au problème de l'évolution de l'ontologie d'un pair. Nous n'exploiterons pas cette stratégie dans cette thèse car ce problème sort du cadre de notre étude.

Dans le cas de la stratégie de préservation de l'ontologie, il est nécessaire de choisir quel(s) mapping(s) supprimer. Le cas présenté en figure 4.11 est une simplification : chacun des mappings figurant peut représenter un ensemble de mappings. Lorsque la non conservativité est provoquée par l'ajout d'un seul mapping, cela signifie que l'un des ensembles ne contient qu'un seul mapping. On propose de supprimer le dernier mapping ajouté. En effet, l'information qu'il apporte est contredite par au moins un mapping qui a été validé par l'administrateur. Le mapping que l'on souhaite ajouter n'est, lui, pas validé par l'administrateur.

Remarque : on n'étudiera pas le choix dual qui consiste à accepter le nouveau mapping et à détruire les anciens. Bien que ce cas soit intéressant, il correspond plus à une logique de maintenance qu'à une logique de découverte de nouveaux mappings. En effet, lorsqu'un mapping est supprimé, l'information qu'il apporte est perdue. Notre but est d'enrichir les réponses aux requêtes, non de remettre en question la correction des réponses qui existent déjà.

Conclusion

Nous avons présenté une méthode de pré-sélection des éléments à aligner. Cette méthode repose sur la notion de contexte d'interprétation qui est exploitée pour sélectionner les couples d'éléments pour lesquels la recherche de mappings est pertinente. Lorsque cette sélection est insuffisante et que les couples retenus sont trop nombreux, nous proposons également un ensemble de critères de filtrage correspondant aux stratégies que l'administrateur d'un pair peut suivre.

Cette méthode de pré-sélection des éléments à aligner présente des caractéristiques intéressantes. Elle permet de limiter, dans une certaine mesure, les problèmes dus à la polysémie. En effet, l'exploitation du contexte d'interprétation garantit que les éléments à aligner appartiennent à un même domaine.

Toutefois, cette méthode ne permet pas de sélectionner tous les couples pertinents et intéressants d'éléments. En effet, nous nous appuyons sur les connaissances représentées, en particulier le contexte d'interprétation. Il est possible que des éléments aient un contexte d'interprétation commun mais que celui-ci ne soit pas représenté. Dans ce cas, ces éléments ne peuvent être rapprochés par notre méthode.

Chapitre 5

Génération de mappings au sein d'un système pair-à-pair sémantique

Sommaire

Introduction

La méthode présentée au chapitre 4 nous permet de construire un ensemble MC de couples d'éléments pour lesquels la recherche de mappings est *pertinente*. Pour chacun de ces couples, le premier élément appartient à l'ontologie du pair local et le second appartient à l'ontologie d'un pair distant. Notre objectif est maintenant de produire automatiquement un ensemble de mappings établissant une relation de subsomption entre les éléments des couples de MC et d'associer à ces mappings une mesure de confiance.

Plus précisément, MC étant l'union de quatre ensembles MC_{SUP}, MC_{SUB}, MC_{DOM} et MC_{RAN}, on cherchera à produire un mapping de la forme $(E_k \Rightarrow E_i, conf)$ pour les couples (E_k, E_i) appartenant à MC_{SUP}, les couples (E_i, E_k) appartenant à MC_{SUB}, les couples (E_k, E_i) et (E_i, E_k) appartenant à MC_{DOM} et à MC_{RAN} pour lesquels E_k est une propriété. Nous recherchons une relation de subsomption entre éléments et non de similarité. Ainsi, nous adoptons la notation $E_k \Rightarrow E_i$ signifiant que E_k est un élément plus spécifique que E_i. $conf$ est une mesure de confiance comprise entre 0 et 1. C'est une

indication pour la validation par l'administrateur. Une valeur proche de 0 signifie que le mapping a une grande probabilité d'être incorrect. Une valeur proche de 1 signifie que le mapping a une grande probabilité d'être correct.

L'ajout des mappings dans les connaissances du pair local dépendra de leur validation par l'administrateur du pair local. La figure 5.1 illustre cette décomposition du processus de découverte de mappings. Notons que, de par notre contexte consistant à établir des mappings avec des éléments d'ontologies multiples, ne garder que le meilleur mapping n'a pas de sens.

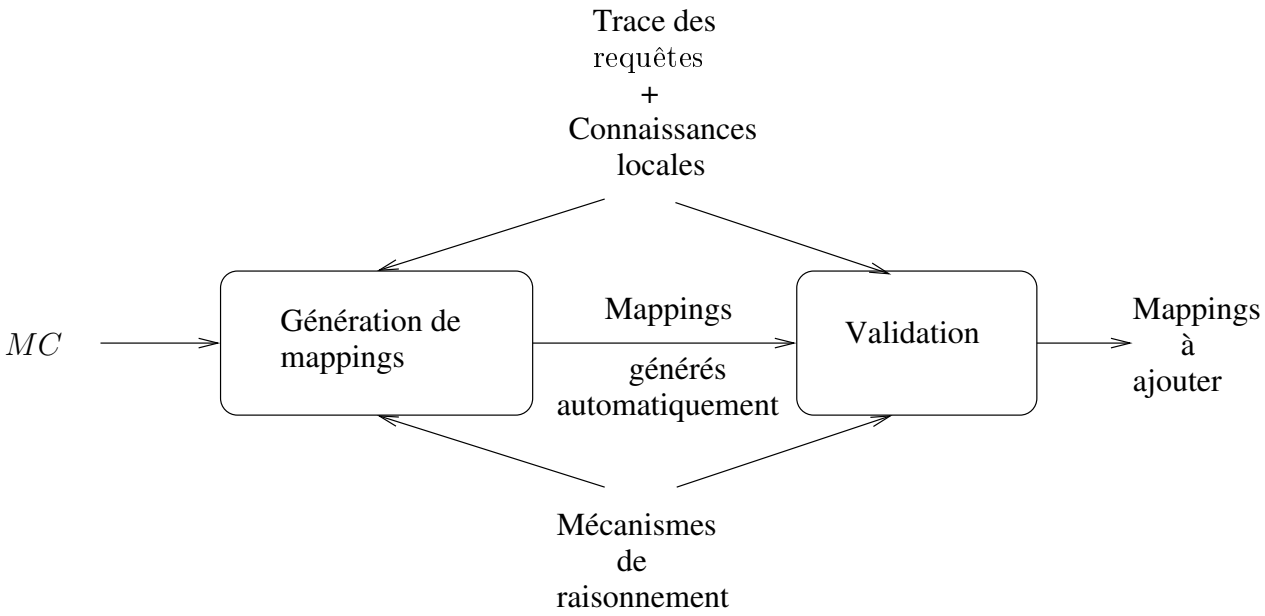

FIG. 5.1 – Décomposition du processus de découverte de mappings entre les éléments des couples de MC

La recherche de mappings pour les couples de MC ressemble au problème classique d'alignement d'ontologies. Cependant, certaines particularités l'en distinguent du fait du contexte d'étude.

- MC est l'union de quatre ensembles MC_{SUP}, MC_{SUB}, MC_{DOM} et MC_{RAN} (cf. 4.2.2) correspondant chacun à un type de mapping recherché (généralisation, spécialisation, typage de domaine ou typage de portée). Nous savons donc, pour chaque couple de MC, quels sont exactement les types de mappings à rechercher.
- Les éléments constituant les couples de MC appartiennent à des ontologies de nombreux pairs ayant chacun leur propre terminologie.
- Les mappings recherchés ont une cardinalité $n : m$ du fait de la recherche de mappings entre un élément d'une ontologie locale et des éléments de nombreuses ontologies distantes et du fait de l'approche proposée qui consiste à rechercher plusieurs spécialisants et/ou généralisants distants pour chaque élément local, et plusieurs spécialisants et/ou généralisants locaux pour un élément distant.
- Seule l'ontologie du pair local est connue. Les éléments appartenant à des ontologies de pairs distants, quand ils sont connus, sont des éléments isolés.

- Les éléments appartenant à l'ontologie du pair local sont décrits uniquement par leurs labels et les liens qu'ils entretiennent entre eux. Les éléments appartenant aux ontologies des pairs distants sont décrits uniquement par leurs labels.
- Les mécanismes de raisonnement mis en œuvre dans les systèmes pair-à-pair sémantiques sont exploitables par le processus de génération de mappings.

Néammoins, les techniques d'alignement de la littérature basées sur la structure des ontologies et consistant à repérer des structures analogues dans les deux ontologies alignées ne sont pas utilisables car la structure de l'ontologie d'un pair distant n'est pas connue. De même, les techniques d'alignement exploitant une description détaillée (attributs, cardinalité, commentaires, ...) des éléments ne sont pas exploitables car, du fait du processus de construction de MC par exploitation, entre autres, des mécanismes de réécriture, les éléments d'ontologies de pairs distants sont décrits uniquement par leurs labels. Enfin, les techniques d'alignement de la littérature sélectionnent le meilleur mapping parmi un ensemble de mappings candidats pour chaque élément d'une ontologie source. Ceci s'explique par le fait qu'il s'agit de rechercher quel est, dans une ontologie cible, l'élément le plus similaire à un élément d'une ontologie source. Ces techniques ne sont pas adaptées à notre contexte d'étude qui consiste à chercher des éléments liés par un lien de subsomption au sein des nombreuses ontologies d'un système pair-à-pair. Ainsi, pour un élément donné, nous recherchons plusieurs mappings même si ceux ci sont sélectionnés pour satisfaire le critère d'optimalité des mappings (cf. 2.1).

Notons enfin que les techniques d'alignement qui ne recherchent que des liens d'équivalence ne conviennent pas à notre contexte d'étude car, si on admet qu'un lien d'équivalence correspond à deux liens de subsomption réciproques, ces techniques permettent de trouver un mapping pour un couple de MC uniquement si ce couple appartient à la fois à MC_{SUB} et à MC_{SUP}.

En revanche, les mécanismes de raisonnement mis en œuvre dans SomeRDFS qui ont déjà été mis à contribution peuvent être une nouvelle fois exploités. Cette exploitation est intéressante. Elle permet, en particulier, d'inférer des relations de subsomption entre des éléments d'ontologies distantes indépendamment de considérations terminologiques portant sur les noms des labels. Les relations inférées sont des relations de subsomption dont la sémantique est bien définie. Elles sont, par ailleurs, sûres, contrairement à la majorité des relations établies par application de techniques d'alignement d'ontologies. L'exploitation des mécanismes de raisonnement compense ainsi certaines contraintes décrites ci-dessus.

Comme la pose de requêtes est coûteuse, nous souhaitons limiter le nombre de requêtes posées. Un certain nombre de requêtes ont déjà été posées par les utilisateurs et lors de la construction de MC (cf. chapitre 4). Nous proposons, dans un premier temps, d'exploiter la trace de ces requêtes. L'analyse de cette trace couplée à des techniques d'alignement adaptées au contexte d'étude telles que les techniques d'alignement syntaxiques permet de proposer des mappings pour certains couples de MC (cf. 5.1). Dans un second temps, lorsque l'exploitation de la trace des requêtes ne permet pas de produire un mapping, la pose de nouvelles requêtes est nécessaire. Nous proposons de profiter de ces requêtes pour exploiter l'ensemble du système pair-à-pair comme une ressource externe afin de proposer

des mappings que les mécanismes de raisonnement ne permettent pas de produire (cf. 5.2). La figure 5.2 illustre le processus de génération de mappings exploitant ces deux techniques. Les mappings générés doivent être validés par l'administrateur avant d'être ajoutés à la connaissances d'un pair. Nous décrivons en section 5.3 les outils d'aide à la validation proposés. La chaîne de traitement complète associée à la découverte de mappings est décrite en section 5.4.

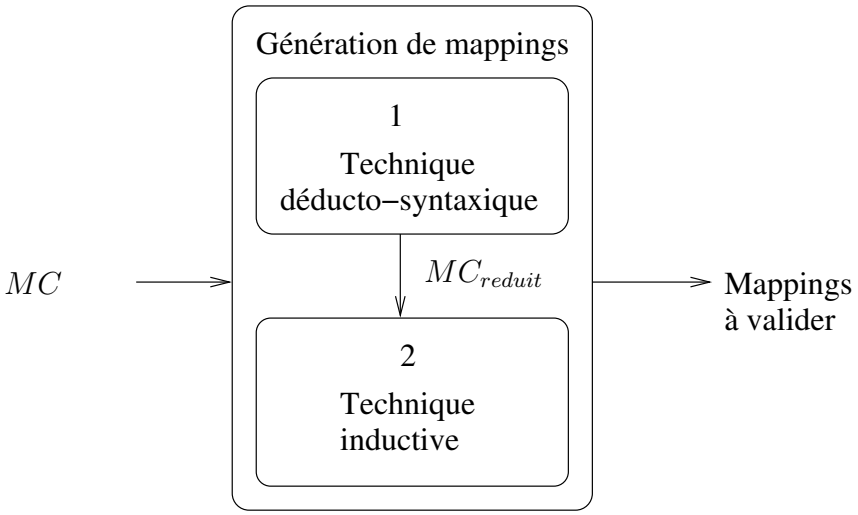

FIG. 5.2 – Décomposition du processus de génération de mappings

5.1 Technique déducto-syntaxique

L'objectif de cette technique est de vérifier si le lien $E_k \Rightarrow E_i$ associé à un couple de MC, composé des éléments E_i et E_k, peut être trouvé à partir des connaissances du pair local (ontologie et mappings) et de la trace des requêtes posées. Lorsque les requêtes nécessaires ont été posées, l'exploitation des mécanismes de raisonnement suffit pour conclure. Dans le cas contraire, l'application de techniques d'alignement en complément des mécanismes de raisonnement peut permettre de trouver ce lien sans que la pose de nouvelles requêtes soit nécessaire.

La technique que nous proposons consiste à trouver un élément E_m tel que les liens $E_k \Rightarrow E_m$ et $E_m \Rightarrow E_i$ sont connus ou peuvent être obtenus par exploitation des mécanismes de raisonnement seuls ou combinés à des techniques d'alignement. Cet élément E_m est recherché à partir des connaissances du pair local et de la trace des requêtes posées.

5.1.1 Exploitation des seuls mécanismes de raisonnement

La pose d'une requête $Q \equiv E_i$ permet d'identifier l'ensemble des spécialisants de l'élément E_i dans le système pair-à-pair. Dans notre approche, nous exploitons la trace des requêtes déjà posées pour vérifier si $E_k \Rightarrow E_i$ est vrai. Ceci revient à vérifier que E_k appartient à l'ensemble des spécialisants de E_i. Toutefois, la trace des requêtes posées ne contient pas nécessairement Q. Dans ce cas, la vérification peut être effectuée à partir

d'autres requêtes, complétée, soit par une analyse des réécritures obtenues, soit par une exploitation des connaissances du pair local.

Exploitation de la seule trace des requêtes posées

La trace des requêtes provient des résultats des réécritures de requêtes de la forme $Q_1 \equiv E_i$ ou $Q_2 \equiv E_i \wedge_{j \neq i} \{E_j\}$. Cette trace permet de savoir si $E_k \Rightarrow E_i$ est vrai en vérifiant qu'elle contient une requête du type Q_1 ou Q_2 et que E_k fait partie des spécialisants de E_i par analyse des réécritures.

Nous distinguons deux cas :

- L'élément E_i est un élément de l'ontologie locale. Les requêtes du type Q_1 et Q_2 ont déjà été exploitées par la méthode décrite au chapitre 3 et ont pu permettre l'ajout de $E_k \Rightarrow E_i$ s'il s'agit d'un *raccourci de mapping*.
- L'élément E_i est un élément de l'ontologie d'un pair distant. Si E_k est un spécialisant de E_i, le mapping $(E_k \Rightarrow E_i, 1)$ est généré. Contrairement au cas précédent, ce cas n'est pas traité par la méthode de sélection des mappings inférables décrite au chapitre 3 qui ne concerne que les requêtes formées à partir des éléments de l'ontologie du pair local.

Exploitation des connaissances du pair local associées à la trace des requêtes posées

Lorsqu'aucune requête de la forme Q_1 ou Q_2 (cf. paragraphe précédent) n'a jamais été posée, l'exploitation de la trace des requêtes posées ne permet pas de conclure sur la spécialisation d'un élément E_i par un élément E_k. La prise en considération des connaissances du pair local permet d'exploiter d'autres requêtes. Ces requêtes sont de la forme $Q_3 \equiv E_m$ ou $Q_4 \equiv E_m \wedge_{j \neq m} \{E_j\}$, E_m étant un spécialisant de E_i dans la connaissance du pair local. Si l'analyse des réécritures de ces requêtes montre que E_k est un spécialisant de E_m, il est, par inférence, un spécialisant de E_i.

Exemple : sur la figure 5.3(a), l'élément $\mathcal{P}_3$:*Jambon* est un spécialisant de l'élément $\mathcal{P}_2$:$Viande$ dans la connaissance de $\mathcal{P}_2$. S'il existe, dans la trace, une requête $Q \equiv \mathcal{P}_3$:*Jambon*, on trouve $\mathcal{P}_1$:*Serrano* et $\mathcal{P}_3$:*Jambon cru* parmi les réécritures de Q. On peut alors conclure que les couples $(\mathcal{P}_2$:$Viande, \mathcal{P}_3$:*Jambon cru*$)$ et $(\mathcal{P}_2$:$Viande, \mathcal{P}_1$:*Serrano*$)$ de MC_{SUB} permettent de générer les mappings $(\mathcal{P}_3$:*Jambon cru* $\Rightarrow \mathcal{P}_2$:$Viande, 1)$ et $(\mathcal{P}_1$:*Serrano* $\Rightarrow \mathcal{P}_2$:$Viande, 1)$.

5.1.2 Exploitation conjointe des mécanismes de raisonnement et de techniques d'alignement syntaxiques

Les connaissances du pair local et l'ensemble des requêtes posées sont limités. Certains liens $E_k \Rightarrow E_i$ déductibles à partir des mécanismes de raisonnement appliqués à l'ensemble du système pair-à-pair peuvent ne pas être déduits à partir des techniques présentées précédemment, qui exploitent les seules connaissances du pair local et la trace des requêtes posées. Toutefois, l'application de ces techniques peut avoir permis de trouver un élément E_m appartenant à l'ontologie d'un pair distant ou à l'ontologie du pair local partageant

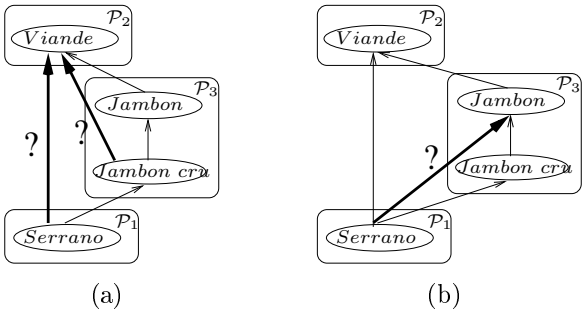

FIG. 5.3 – Exemple de mappings identifiables par la technique déducto-syntaxique

une relation de subsomption avec l'élément local mais pas avec l'élément distant. Ainsi, dans la figure 5.4(a), E_i est un élément de l'ontologie du pair local $\mathcal{P}$, le lien $E_m \Rightarrow E_i$ est obtenu à partir des connaissances de $\mathcal{P}$ mais le lien $E_k \Rightarrow E_m$ n'a pu être déduit. De même, dans la figure 5.4(b), E_k est un élément de l'ontologie du pair local $\mathcal{P}$, le lien $E_k \Rightarrow E_m$ est obtenu à partir des connaissances de $\mathcal{P}$ mais le lien $E_m \Rightarrow E_i$ n'a pu être déduit.

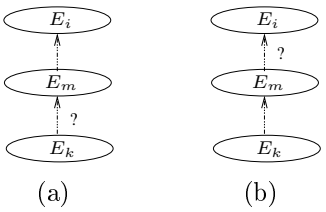

FIG. 5.4 – Schéma général du problème des mappings non identifiables par exploitation des seuls mécanismes de raisonnement

Exemple : sur la figure 5.3(a), si aucune requête composée de $\mathcal{P}_3$:*Jambon* ou de $\mathcal{P}_2$:*Viande* n'a été posée, aucun mapping n'est trouvé pour le couple ($\mathcal{P}_2$:*Viande*, $\mathcal{P}_3$:*Jambon cru*) de MC_{SUB} car pour déduire la spécialisation de $\mathcal{P}_2$:*Viande* par $\mathcal{P}_3$:*Jambon cru* il est nécessaire connaître le lien de spécialisation de $\mathcal{P}_3$:*Jambon* par $\mathcal{P}_3$:*Jambon cru*. Or, ce lien n'est pas connu de $\mathcal{P}_2$.

Exemple : sur la figure 5.3(b), lorsque le pair local est $\mathcal{P}_1$, si aucune requête composée de $\mathcal{P}_3$:*Jambon* n'a été posée, aucun mapping n'est trouvé pour le couple ($\mathcal{P}_1$:*Serrano*, $\mathcal{P}_3$:*Jambon*) de MC_{SUP} car pour déduire la généralisation de $\mathcal{P}_1$:*Serrano* par $\mathcal{P}_3$:*Jambon* il est nécessaire connaître le lien de généralisation de $\mathcal{P}_3$:*Jambon cru* par $\mathcal{P}_3$:*Jambon*. Or ce lien n'est pas connu de $\mathcal{P}_1$.

Les éléments E_m peuvent appartenir à l'ontologie du pair local ou à l'ontologie d'un pair distant. Les liens manquants peuvent donc correspondre soit à des liens ontologiques soit à des mappings. Les mécanismes de raisonnement n'ayant pas suffi à trouver ces liens, nous proposons de les rechercher grâce à des techniques d'alignement peu coûteuses et adaptées au contexte pair-à-pair. Dans un premier temps, nous détaillerons comment l'association de ces techniques et des mécanismes de raisonnement permet d'exploiter la trace

des requêtes posées et les connaissances du pair local afin de générer des mappings. Dans un deuxième temps, nous proposerons un exemple de techniques d'alignement adaptées.

Composition des mécanismes de raisonnement et des techniques d'alignement

Notre objectif est de calculer une mesure de confiance pour un lien $E_k \Rightarrow E_i$ donné. Lorsque ce lien ne peut être déduit par exploitation des connaissances du pair local et de la trace des requêtes posées, nous proposons de le trouver en appliquant des techniques d'alignement. Dans ce cas la mesure de confiance associée à ce lien correspondra à la mesure de confiance associée à l'alignement des éléments E_i et E_k. Pour cet alignement, nous proposons, tout d'abord, de rechercher l'ensemble $S(E_i)$ des éléments E_m plus spécifiques que E_i (cf. figure 5.5(a)) ou l'ensemble $S(E_k)$ des éléments E_m plus généraux que E_k (cf. figure 5.5(b)) par exploitation des mécanismes de raisonnement sur les connaissances du pair local et la trace des requêtes posées. Nous proposons, ensuite, d'aligner les éléments de $S(E_i)$ avec E_k ou les éléments de $S(E_k)$ avec E_i.

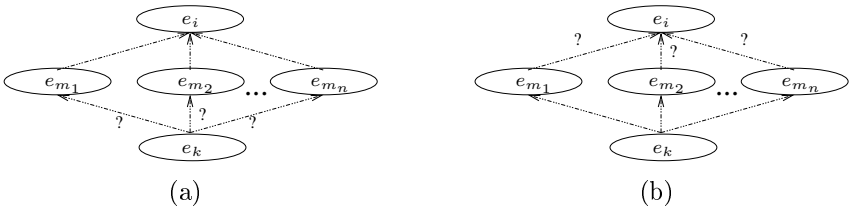

FIG. 5.5 – Schéma général dans le cas des mappings identifiables par composition des mécanismes de raisonnement et des techniques d'alignement

Pour limiter le nombre d'alignements réellement effectués, nous proposons de nous limiter aux cas où l'alignement de E_k (resp. E_i) avec E_m est un problème plus simple que l'alignement de E_k avec E_i. On distingue alors quatre cas :

- E_k (resp. E_i) et E_m sont des éléments de l'ontologie locale. Nous supposons que tous les liens entre deux éléments de l'ontologie locale sont représentés. Ainsi, si le lien entre E_k et E_m (resp. E_i et E_m) existait, il serait représenté et aurait permis de déduire $E_k \Rightarrow E_i$ par exploitation des mécanismes de raisonnement. Ce lien n'est donc pas recherché.
- E_k (resp. E_i) est un élément de l'ontologie d'un pair distant et E_m est un élément de l'ontologie du pair local. L'alignement de ces deux éléments pose le même problème que l'alignement entre E_k et E_i donc les mêmes difficultés. Ce lien n'est recherché que si (E_m, E_k) (resp. (E_m, E_i)) est également un couple de MC.
- E_k (resp. E_i) et E_m sont des éléments des ontologies de deux pairs distants. On possède alors moins d'informations pour aligner E_k et E_m (resp. E_i et E_m) que pour l'alignement de E_i et E_k car les éléments E_k et E_m (resp. E_i et E_m) ne sont décrits que par leur label. La définition de ces éléments n'est pas connue. Les ontologies auxquelles ils appartiennent sont inconnues du pair local. Nous choisissons donc de ne pas traiter ces cas par manque d'information.
- E_k (resp. E_i) et E_m sont des éléments de l'ontologie d'un même pair distant. Contrairement à l'ontologie du pair local, l'ontologie de chacun des pairs distants n'est pas connue. L'alignement de E_k et E_m (resp. E_i et E_m) est une façon de découvrir cette ontologie. L'avantage est que la terminologie utilisée est unique. Les labels

sont homogènes, ce qui permet de limiter les problèmes d'homonymie, de synonymie et d'obtenir des résultats issues de techniques d'alignement terminologiques relativement précis. Notons en effet que E_k (resp. E_i) et E_m n'appartiennent pas à l'ontologie du pair local, seul leur label est connu, ce qui restreint le champ des techniques d'alignement applicables aux seules techniques terminologiques ou syntaxiques.

Si, pour un des éléments E_m, l'alignement avec E_k (resp. E_i) permet de conclure que E_k est un spécialisant de E_m (resp.E_i est un généralisant de E_m), alors nous pouvons conclure, par inférence, que le lien $E_k \Rightarrow E_i$ est un mapping. La mesure de confiance de ce mapping est la même que celle accordée à l'alignement entre E_k (resp. E_i) et E_m.

Exemple : soit le couple ($\mathcal{P}_2$:$Viande$, $\mathcal{P}_3$:$Jambon\ cru$) dans MC_{SUB}, et soit $\mathcal{P}_3$:$Jambon \Rightarrow \mathcal{P}_2$:$Viande$ un mapping de $\mathcal{P}_2$, comme représenté dans la figure 5.3(a), $\mathcal{P}_3$:$Jambon\ cru \Rightarrow \mathcal{P}_3$:$Jambon$ est obtenu par alignement de $\mathcal{P}_3$:$Jambon\ cru$ et $\mathcal{P}_3$:$Jambon$ avec une mesure de confiance de 0.9. Ce résultat nous permet de conclure que ($\mathcal{P}_3$:$Jambon\ cru \Rightarrow \mathcal{P}_2$:$Viande, 0.9$) est un mapping.

Notons que, lorsque E_i est un élément de l'ontologie locale, E_k étant alors considéré comme un spécialisant de E_i, les éléments plus spécifiques que E_k sont également des spécialisants de E_i. Les requêtes de la forme $Q_3 \equiv E_k$ ou $Q_4 \equiv E_k \wedge_{j \neq m} \{E_j\}$ permettent alors d'identifier des spécialisants de E_i et ainsi générer des mappings $(E_{k'} \Rightarrow E_i, conf)$ ayant la même mesure de confiance que $(E_k \Rightarrow E_i, conf)$ à partir de couples $(E_i, E_{k'})$ de MC.

Exemple : sur la figure 5.3(a), $\mathcal{P}_3$:$Jambon\ cru$ est alors considéré comme un spécialisant de $\mathcal{P}_2$:$Viande$. Si la requête $Q_5 \equiv \mathcal{P}_3$:$Jambon\ cru$ a été posée, l'étude de ses réécritures permet d'identifier $\mathcal{P}_1$:$Serrano$ comme un spécialisant de $\mathcal{P}_3$:$Jambon\ cru$ puis d'en conclure que ($\mathcal{P}_1$:$Serrano \Rightarrow \mathcal{P}_2$:$Viande, 0.9$) est un mapping.

Description d'une technique d'alignement adaptée

Nous avons choisi de nous intéresser uniquement à l'alignement d'éléments d'une même ontologie à partir de techniques d'alignement syntaxiques. Cette exploitation particulière des techniques d'alignement syntaxiques au sein d'une même ontologie garantit l'homogénéité entre les termes utilisés pour désigner les éléments alignés et augmente ainsi la précision des résultats obtenus.

Nous proposons de nous appuyer sur une technique d'alignement syntaxique (cf. 1.1.1). La technique que nous exploitons considère qu'il existe un lien de subsomption $E_1 \Rightarrow E_2$ entre deux éléments E_1 et E_2, respectivement désignés par des labels l_1 et l_2 si l_2 est inclus dans l_1. Dans ce cas, le label l_1 contient alors au moins un terme t non inclus dans l_2. l_1 est plus précis que l_2 et dénote donc un élément plus spécifique.

Exemple : la comparaison des labels *Jambon cru* et *Jambon* permet de déduire que $\mathcal{P}_3$:*Jambon cru* est un spécialisant de $\mathcal{P}_3$:*Jambon*. Le terme *cru* correspond à la particularité de *Jambon cru* par rapport à *Jambon*.

Cependant, les labels peuvent être composés de plusieurs termes. Dans ce cas, le terme principal et les termes auxiliaires de chaque label doivent d'abord être identifiés afin d'appliquer correctement cette technique. Cette identification est permise par l'exploitation d'outils d'étiquetage morpho-syntaxique (par ex. TreeTagger (Schmid (1994)) qui distinguent les noms, verbes, adjectifs, prépositions et déterminants. Ces catégories permettent de considérer les noms comme des termes principaux, sauf s'ils sont derrière un déterminant, les termes appartenant à une autre catégorie lexicale étant considérés comme auxiliaires. Ceci est la règle implémentée dans l'outil d'alignement *TaxoMap* Reynaud & Safar (2007) conçu dans notre équipe de recherche, que nous avons choisie de retenir.

L'exemple précédent illustre l'application de cette technique lorsque tous les éléments comparés sont des classes. Lorsque certains éléments sont des propriétés, cette technique n'est applicable que si on retrouve, dans le label désignant les propriétés, les labels désignant les classes qui y sont liées. Dans le cas contraire, l'application de ces techniques peut générer des erreurs.

Contrairement aux classes qui représentent chacune un unique ensemble, les propriétés représentent chacune un lien sémantique entre deux ensembles (un domaine et une portée). Le label d'une propriété ne dénote pas nécessairement ces trois informations.

Exemple : soit une propriété P représentant le lien entre un artiste et les œuvres qu'il a créées.
- Lorsque P est désignée par le label *créé*, il est impossible d'identifier, au sein du label, le domaine et la portée de P.
- Lorsque le label désignant cette propriété est *Artiste*|*creer*|*Œuvre*, l'analyse du label peut permettre de trouver que *Artiste* caractérise le domaine de P et que *Œuvre* caractérise la portée de P.

Lorsque le label d'une propriété dénote à la fois son domaine, sa portée et le lien sémantique représenté, une étape d'analyse du label permet d'identifier chacun des sous-labels dénotant ces différents éléments. Dans notre exemple, le caractère | sépare les différents sous-labels. Le séparateur peut être différent d'une ontologie à l'autre. L'étape d'analyse du label consiste à identifier ce séparateur puis les différents sous-labels.

Le domaine et la portée peuvent être dénotés en faisant référence aux labels des classes de l'ontologie comme dans l'exemple précédent. Ils peuvent également être dénotés sans faire référence aux labels des classes de l'ontologie.

Exemple : soit une propriété P représentant le lien entre un artiste et les œuvres qu'il a créées.

- Lorsque le label désignant cette propriété est $Artiste|creer|Œuvre$, on retrouve le label de la classe *Artiste* qui type le domaine de P et le label de la classe *Œuvre* qui type la portée de P.
- Lorsque le label désignant cette propriété est $Createur|creer|Creation$, *Createur* et *Creation* ne sont pas des labels désignant des classes de l'ontologie, il est impossible de retrouver les classes typant respectivement le domaine et la portée de P.

Lorsqu'on recherche un mapping entre une classe et une propriété : nous savons s'il s'agit de la recherche d'un mapping correspondant à un typage de domaine ou de portée.

Lorsque le label d'une propriété ne dénote ni son domaine ni sa portée, il dénote uniquement le lien sémantique représenté. Sa comparaison avec le label d'une classe n'est alors pas justifiée.

Lorsque le label d'une propriété dénote à la fois son domaine, sa portée et le lien sémantique représenté, la comparaison du sous-label concerné (domaine ou portée) avec le label d'une classe est alors justifié. Lorsque le label d'une propriété fait référence aux labels des classes, les sous-labels dénotant son domaine et sa portée sont des labels de classes. On se ramène donc à l'application de la technique d'alignement basée sur les sous-chaînes appliquée aux labels de classes. Dans le cas où le label d'une propriété ne fait pas référence à des labels des classes, les termes comparés sont, par hypothèse, différents. La technique d'alignement basée sur les sous-chaînes ne permet donc pas d'obtenir des résultats satisfaisants.

Lorsqu'on recherche un mapping entre deux propriétés : l'alignement de deux propriétés P_1 et P_2 nécessite de comparer les domaines et les portées respectifs de P_1 et P_2 ainsi que les liens sémantiques représentés respectivement par ces deux propriétés.

Lorsque le label d'une propriété ne dénote ni son domaine ni sa portée, la technique d'alignement syntaxique, comme toute technique d'alignement n'exploitant que les labels, génère de nombreuses erreurs dues à l'absence de comparaison des domaines et portées respectifs. En particulier, il est possible de trouver un lien de subsomption entre des propriétés inverses.

Exemple : $\bar{P}$, la propriété inverse de P, peut être désignée par le terme *créé par*. L'application de la technique basée sur les sous-chaînes conlut que $\bar{P}$ est un spécialisant de P.

Remarque : La technique d'alignement syntaxique que nous exploitons permet d'identifier des liens de subsomption dans des ontologies où les éléments sont désignés par des termes d'une même langue. Dans un cadre pair-à-pair mondial, on ne peut garantir l'utilisation d'une unique langue. Il est donc nécessaire de détecter la langue utilisée pour chaque ontologie avant l'application des techniques d'alignement.

5.2 Technique inductive

La technique précédente permet d'identifier des couples de MC pour lesquels le lien recherché peut être obtenu à partir des connaissances du pair local et de la trace des requêtes posées. Cependant, pour certains couples de MC, le lien recherché ne peut pas être obtenu car il ne peut pas être déduit par application des mécanismes de raisonnement appliqués uniquement aux connaissances du pair local et à la trace des requêtes posées même lorsqu'elles sont enrichies par des techniques d'alignement syntaxique. En revanche, ce lien pourrait être déduit par exploitation des mécanismes de raisonnement sur l'intégralité du système pair-à-pair. Ainsi, étant donné un lien $E_k \Rightarrow E_i$, la pose de la requête $Q \equiv E_i$ permet de vérifier si E_k est effectivement un spécialisant de E_i et donc si $E_k \Rightarrow E_i$ est déduit.

Lorsqu'un lien $E_k \Rightarrow E_i$ ne peut pas être déduit dans le système pair-à-pair, cela signifie que E_k n'est pas représenté comme un spécialisant de E_i dans les connaissances du système pair-à-pair. La description de la connaissance dans le système pair-à-pair ne pouvant pas être jugée complète, il est alors impossible de distinguer si E_k est un spécialisant de E_i pour lequel le lien de spécialisation est manquant ou si E_k n'est pas un spécialisant de E_i. L'alignement de E_i et E_k doit permettre de faire cette distinction afin de proposer un mapping uniquement lorsqu'il s'agit d'un lien manquant.

Les deux éléments à aligner appartiennent à deux ontologies distinctes dont une seule est connue. Nous ne connaissons donc que le label désignant l'élément distant et le *contexte d'interprétation* qui a permis le rapprochement de ces deux éléments (cf. 4.2). Ces informations ne sont pas suffisantes pour permettre un alignement, il est donc nécessaire de les enrichir. Nous proposons d'exploiter une ressource externe (cf. 1.1.1) pour réaliser cet enrichissement.

Une ressource externe est une référence qui fait office d'oracle. L'exploitation des ressources externes nécessite une phase d'ancrage. Cette phase consiste à trouver, dans la ressource externe, une correspondance (appelée point d'ancrage ou ancre) pour chacun des éléments des ontologies à aligner. Lorsque la phase d'ancrage est terminée, l'alignement de deux éléments consiste à vérifier s'ils ont la même ancre dans la ressource externe ou s'il existe, dans la ressource externe, un lien entre leurs ancres respectives ou de calculer la similarité entre les ancres des éléments à aligner.

Les ressources externes peuvent être plus où moins structurées. Les moins structurées (thesaurus, dictionnaire, base de données lexicale...) contiennent des liens de synonymie et d'hyperonymie entre différents termes. Un lien de synonymie entre deux termes pourra être traduit par un lien d'équivalence. Un lien d'hyperonymie entre deux termes pourra être traduit par un lien de subsomption. L'avantage de ce type de ressources est qu'elles sont généralement validées. Leur inconvénient est que ces ressources ne sont pas nécessairement spécifiques d'un domaine. La phase d'ancrage nécessite alors une étape de désambiguisation car un même terme peut avoir plusieurs significations différentes selon le domaine d'application.

Les ressources externes plus structurées (ontologies, taxonomies...) manipulent des classes et des propriétés. Ce type de ressource est soit spécifique d'un domaine d'application, soit plus général. Si la ressource utilisée est spécifique, la désambiguisation lors de la phase d'ancrage est facilitée. S'il s'agit d'une ressource générale, la phase d'ancrage nécessite une désambiguisation.

Dans notre contexte, chaque pair a son ontologie relative à son domaine d'application. Il est donc impossible d'exploiter une ressource externe qui convienne quel que soit le domaine d'application et donc quel que soit le pair.

Notre approche consiste à considérer le système pair-à-pair comme une ressource externe. Contrairement aux ressources externes traditionnelles, les éléments à aligner appartiennent à la ressource externe. La phase d'ancrage n'est donc pas nécessaire.

Une autre particularité est que chaque pair est responsable de son ontologie et de son vocabulaire. Il ne s'agit donc pas d'une ressource unique, homogène, validée et faisant référence.

Cependant, le nombre d'ontologies composant le système pair-à-pair permet une approche statistique basée sur les labels des éléments. Ainsi, le fait qu'un label désigne un spécialisant de E_j dans une ontologie donnée ne signifie pas nécessairement que tout élément désigné par ce label dans une autre ontologie soit un spécialisant de E_j. Par contre, si ce label désigne un spécialisant de E_i dans de nombreuses ontologies distinctes, alors il est probable qu'un élément désigné par ce label dans une autre ontologie soit également un spécialisant de E_i.

Nous proposons de calculer la confiance d'un mapping $E_k \Rightarrow E_i$ à partir de la liste $\mathrm{L}(E_i)$ des labels utilisés pour désigner un spécialisant de E_i. Ce calcul prend en compte :

- la fréquence à laquelle le label désignant E_k est trouvé dans $\mathrm{L}(E_i)$. Une fréquence faible se traduit par un score proche de 0 et une fréquence forte par un score proche de 1.
- le nombre de pairs interrogés pour construire $\mathrm{L}(E_j)$. Lorsque peu de pairs participent à $\mathrm{L}(E_j)$, cette liste n'est pas représentative. Une taille réduite se traduit par un score proche de 0.5.
- la représentativité des pairs interrogés pour construire $\mathrm{L}(E_j)$. Le label ne doit pas être considéré de la même façon lorsqu'il désigne l'unique spécialisant E_i dans une ontologie, et lorsqu'il désigne un des nombreux spécialisants de E_i dans une autre ontologie. Ce paramètre n'influe pas directement sur le score mais il modifie la façon dont est calculée la fréquence d'un label et donc, à terme, la confiance.

Ainsi, l'algorithme 10 calcule la confiance d'un mapping potentiel $(\mathcal{P}_k{:}E_m \Rightarrow \mathcal{P}_i{:}E_j, conf)$. Une requête est posée en utilisant $\mathcal{P}_i{:}E_j$, puis on recherche si le label désignant $\mathcal{P}_k{:}E_m$ est souvent utilisé pour désigner des éléments plus spécifiques que $\mathcal{P}_i{:}E_j$.

Algorithme 10 : AnyCommonLabel($\mathcal{P}_k$:$E_m \Rightarrow \mathcal{P}_i$:$E_j$)

Entrées : un mapping P_k:$E_m \Rightarrow P_i$:E_j.
Sorties : la confiance du mapping P_k:$E_m \Rightarrow P_i$:E_j.

```
1 nbOk = 0
2 knownPeers ⇐ ∅
3 pour tous les P_n:E_s in rewrite(P_i:E_j) faire
4 |   si P_n ∈ knownPeers alors
5 |   |   add P_n to knownPeers
  |   fin
6 |   si equiv(E_s, E_m) alors
7 |   |   nbOk = nbOk + 1
  |   fin
  fin
8 retourner score(nbOk, |knownPeers|)
```

Cet algorithme est appliqué aux mappings déduits de l'ensemble MC pour lesquels l'application de la technique déducto-syntaxique n'a pas généré de mappings. La requête posée ligne 3 permet d'identifier tous les spécialisants de P_i:E_j. On compte, en ligne 4, le nombre de pairs distincts se partageant ces spécialisants et, en ligne 6, le nombre de ces spécialisants désignés par le même label que E_k.

Définition 21 (score)
$score(nbOk,\ nbPeers) = \frac{2 \times nbOk + 1}{2 \times nbPeers + 2}$

Définition 22 (nbOk)

$$nbOk(OkPeers, RwPerPeer) = \sum_i \begin{cases} 2 - \frac{|RwPerPeer(i)|}{\max_j(|RwPerPeer(j)|)} & si\ \mathcal{P}_i \in OkPeers \\ \left(1 - \frac{|RwPerPeer(i)|}{\max_j(|RwPerPeer(j)|)}\right) \times 0.5 & sinon \end{cases}$$

Cette fonction prend en paramètre deux ensembles : OkPeers, l'ensemble des pairs utilisant un label pour désigner un élément et RwPerPeer, l'ensemble des réécritures obtenues dans l'algorithme 10 indexées par pair. Cette fonction compte le nombre de fois qu'un label est utilisé pour désigner un spécialisant d'un élément donné en prenant en compte la qualité des pairs et le nombre d'éléments distincts. En effet, on souhaite distinguer l'absence d'emploi d'un terme comme label pour désigner un élément de l'absence de cet élément. Pour avoir une minoration du nombre d'éléments distincts, on prend le plus grand nombre d'éléments dans une même ontologie. Si une ontologie comporte peu d'éléments, le fait de trouver un élément désigné par le label recherché est plus important que si cette ontologie comporte beaucoup d'éléments. Le fait de ne pas trouver un tel élément est, en revanche, moins pertinent. Il est probable que l'élément soit totalement absent de l'ontologie. Lorsque l'ontologie comporte beaucoup d'éléments, il est probable que l'élément recherché soit désigné par un label utilisant un autre terme.

Cette technique s'applique indifféremment aux relations unaires (classes) et binaires (propriétés). Cependant, elle ne s'applique pas pour aligner des éléments de types différents (trouver les propriétés typées par une classe donnée).

5.3 Validation

Les mappings générés doivent être validés avant d'être ajoutés à la connaissance du pair local. Un mapping est validé si le lien qu'il représente est jugé correct et que ce mapping est un mapping optimal (cf. 2.1). La validation est alors un processus semi-automatique composé d'une phase automatique de vérification de l'optimalité et d'une phase semi-automatique de vérification de la correction.

L'administrateur ne peut juger de la correction d'un mapping uniquement à partir de sa mesure de confiance. Il est nécessaire que la sémantique associée à chacun des éléments mis en jeu soit bien connue. Contrairement aux éléments de l'ontologie du pair local, la sémantique d'un élément appartenant à l'ontologie d'un pair distant n'est pas connue. Nous proposons un ensemble d'outils de validation facilitant la découverte et la compréhension de la sémantique associée aux éléments manipulés.

Par ailleurs, il existe une interdépendance entre les mappings proposés. Cette interdépendance décrite en section 5.3.2 se traduit par le fait que la validation ou le rejet d'un mapping peut entraîner la validation ou le rejet automatique d'autres mappings.

5.3.1 Outils de validation

Bien que l'ontologie d'un pair distant soit inconnue, les informations réunies au cours des étapes précédentes peuvent permettre de comprendre la sémantique associée à certains de ses éléments. Cependant, ces informations sont trop nombreuses pour être exploitées directement. Nous proposons un système d'organisation de ces informations permettant à l'administrateur de se focaliser sur le sous-ensemble des informations obtenues relatives à la validation d'un mapping donné.

Lorsque ces informations ne suffisent pas, l'administrateur peut obtenir des informations supplémentaires par le biais du système pair-à-pair. Nous proposons d'intégrer un ensemble d'outils permettant l'obtention de ces informations supplémentaires dans le système de validation.

Les outils de présentation des informations obtenues et les autres outils d'obtention d'informations supplémentaires sont successivement présentés dans cette section.

Présentation des informations

Dans notre contexte, les mappings générés font intervenir des éléments isolés appartenant aux ontologies de nombreux pairs distincts. Pour chacun de ces mappings, un ensemble d'informations a été réuni au cours des différentes étapes de découverte de mappings. Cependant, présentées comme une masse, ces informations sont difficilement exploitables par un humain. Il est donc nécessaire de permettre à l'administrateur de gérer cette masse d'informations afin d'isoler celles qui justifient la validation ou le rejet d'un mapping généré. Nous présentons successivement 3 critères de présentation pouvant être choisis par l'administrateur.

Critère n°1 : Présentation des informations par pair

Les vocabulaires de plusieurs pairs sont mélangés dans cette masse d'informations. Ces différents vocabulaires appartiennent à des pairs distants et sont mal connus de l'administrateur. Cette hétérogénéité des informations est un problème majeur qui est une source d'erreurs.

Nous proposons de partitionner la masse d'informations en fonction des pairs distants concernés. Il s'agit donc, tout d'abord, de rassembler les mappings générés mettant en jeu le vocabulaire d'un même pair distant. Même si les liens entre les éléments de l'ontologie d'un même pair distant restent inconnus, cette focalisation sur les mappings générés entre le pair local et un unique pair distant à la fois permet de rompre l'isolement des éléments appartenant à l'ontologie d'un pair distant et donc une meilleure compréhension du vocabulaire du pair distant.

Critère n°2 : Présentation d'une vue sur l'ontologie d'un pair distant

Le fait de ne pas connaître les liens existant entre les éléments du vocabulaire d'un pair distant nous oblige à les présenter à l'administrateur sous la forme d'une liste. Cette présentation constitue un réel frein pour la compréhension des éléments et donc pour la validation.

Nous proposons de reconstituer certains liens de l'ontologie d'un pair distant à partir de la trace des requêtes posées et des connaissances du pair local, comme nous l'avons décrit en section 5.1. Les nombreux liens reconstitués lors du processus de génération de mappings forment une *vue* sur l'ontologie d'un pair distant. Nous parlons d'une *vue* sur l'ontologie car seule une partie de l'ontologie est reconstituée et, de plus, des techniques d'alignement peuvent avoir été utilisées pour reconstituer cette ontologie. Nous proposons également d'enrichir cette *vue* sur l'ontologie d'un pair distant en appliquant cette technique à l'ensemble des éléments de l'ontologie du pair distant.

Critère n°3 : Présentation de mappings existants par pair distant

Afin d'aider l'administrateur à valider les mappings générés, il est également nécessaire de lui présenter les liens existants entre les éléments appartenant aux connaissances du pair local et les éléments de l'ontologie d'un pair distant donné. Ces liens correspondent d'une part aux mappings du pair local et, d'autre part, aux liens qui peuvent être inférés dans le système pair-à-pair. Par construction de l'ensemble MC, tous les éléments distants mis en jeu dans un des mappings générés sont liés à un élément appartenant à la connaissance du pair local. La présentation de ces liens pour chaque pair distant permet à l'administrateur de mieux apprécier les conséquences de la validation d'un mapping généré.

Outils de l'Administrateur

La *vue* sur l'ontologie d'un pair distant qui est utilisée par l'administrateur pour valider les mappings générés est partielle. Il est possible que la sémantique de l'élément distant mis en jeu dans un mapping généré ne soit pas décrite de façon très précise dans cette *vue*. Dans ce cas, l'administrateur dispose de deux outils pour comprendre les éléments isolés :

- la pose de nouvelles requêtes afin d'identifier des liens non représentés dans la *vue* sur l'ontologie d'un pair distant.
- l'évaluation de requêtes afin de comprendre la sémantique d'un élément à partir des instances qu'il décrit.

Ces deux outils existent indépendamment du cadre de l'alignement (cf. 1.3.1). Toutefois, leur intégration au sein de l'outil d'alignement permet de faciliter leur utilisation par l'administrateur et d'améliorer l'efficacité de ce dernier pour la tâche de validation.

5.3.2 Interactions entre mappings générés

Lorsqu'un mapping généré est validé, des liens inférables apparaissent entre des éléments qui n'étaient, jusque là, pas liés. Si ces liens inférables correspondent à des mappings générés, la validation de ces mappings ne dépend pas de l'administrateur mais relève d'un traitement automatique.

Par ailleurs, le rejet d'un mapping généré permet d'obtenir des informations sur l'inexistence d'un lien de subsomption entre deux éléments. Ce type d'information n'est pas disponible dans le formalisme de RDF(S). Cette information supplémentaire permet de rejeter automatiquement tout mapping généré dont la validation rendrait inférable le mapping rejeté.

Exemple : Dans la figure 5.6, $M_1 \equiv \mathcal{P}_2{:}Painter \Rightarrow \mathcal{P}_1{:}Artiste$ et $M_2 \equiv \mathcal{P}_2{:}Painter \Rightarrow \mathcal{P}_1{:}Peintre$ sont deux mappings générés.

- Supposons que $\mathcal{P}_2{:}Painter$ désigne des peintres en bâtiment et que M_1 est étudié avant M_2 par l'administrateur. M_1 est alors rejeté par l'administrateur. Si M_2 est étudié et que l'administrateur le valide, M_1 devient inférable bien qu'il ait été rejeté auparavant.
- Supposons que $\mathcal{P}_2{:}Painter$ désigne des artistes peintre et que M_1 est étudié après M_2 par l'administrateur. M_2 est alors validé par l'aministrateur et M_1 devient inférable. La validation de M_1 par l'administrateur n'est donc pas nécessaire.

De façon plus générale, considérons $M \equiv E_k \Rightarrow E_i$ un mapping généré. Soient E_m un généralisant de E_k, E_n un spécialisant de E_k, E_r un spécialisant de E_i et E_s un généralisant de E_i (cf. figure 5.7). Lorsque M est rejeté par l'administrateur, les mappings générés de la forme $E_k \Rightarrow E_r$, de la forme $E_m \Rightarrow E_i$ ou de la forme $E_m \Rightarrow E_r$ sont automatiquement rejetés.

Lorsque M est validé par l'administrateur, les mappings générés de la forme $E_k \Rightarrow E_s$, de la forme $E_n \Rightarrow E_i$ ou de la forme $E_n \Rightarrow E_s$ sont automatiquement validés. Cependant,

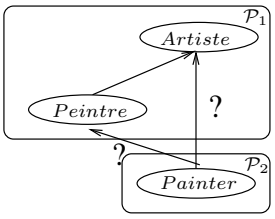

FIG. 5.6 – Exemple d'interaction entre mappings générés

leur ajout doit respecter le critère d'optimalité (cf. 2.1) et la stratégie sur les pairs (cf. 2.2.1) choisie par l'administrateur du pair local.

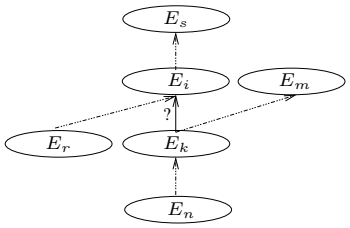

FIG. 5.7 – Ensemble des interactions possibles entre des mappings générés

5.4 Mise en œuvre

L'exploitation des mécanismes de raisonnement est un des fondements de notre approche. Cependant, cette exploitation est coûteuse. Afin de réduire ce coût, nous avons donc proposé d'exploiter, aux différentes étapes de notre approche, la trace des requêtes posées limitant ainsi le nombre de requêtes posées. Cependant, contrairement aux requêtes, l'analyse de la trace n'est pas conservée. Bien que ce traitement soit moins coûteux que la pose de requêtes, il est souhaitable de ne pas faire plusieurs fois l'analyse de la trace d'une même requête. De plus, dans la section 5.3, nous avons proposé la construction, par une analyse complète de la trace des requêtes, d'une *vue* sur l'ontologie d'un pair distant pour aider l'administrateur à valider les mappings générés. Ces *vues* sont obtenues par une analyse complète de la trace des requêtes posées.

Nous proposons de construire les *vues* sur les ontologies des pairs distants avant la phase de génération de mappings. Ainsi, la trace de chaque requête n'est analysée qu'une seule fois et peut être exploitée pour autant de couples de MC que nécessaire sans coût supplémentaire.

Par ailleurs, l'interdépendance des mappings générés permet d'automatiser la validation ou le rejet de certains mappings générés comme nous l'avons décrit en section 5.3. Cependant, ces mappings ont été générés par application d'une des techniques de génération de mappings que nous avons proposées. Étant rejetés, ils sont donc calculés inutilement.

Nous proposons une application par lots des traitements, comme illustré en figure 5.8, afin de limiter le nombre de mappings générés inutilement. Ainsi, aucune requête supplémentaire n'est posée au cours de la phase d'alignement. Le calcul de la mesure de confiance est effectué uniquement à partir de la trace des requêtes posées. Nous distinguons 3 types de calcul :

- global : les requêtes nécessaires ont été posées et la pose d'une nouvelle requête ne peut pas modifier la valeur de la mesure de confiance.
- partiel : certaines des requêtes nécessaires ont été posées. La pose d'une nouvelle requête peut modifier la valeur de la mesure de confiance. Cependant, elle reste une indication fiable.
- local : aucune requête nécessaire n'a été posée. Le calcul est effectué uniquement sur les connaissances du pair local. La valeur obtenue n'est pas une indication totalement fiable. Cependant, elle est la seule information disponible.

Le type de calcul utilisé pour produire la mesure de confiance est indiqué pour chaque mapping généré. L'administrateur y a donc accès en phase de validation. Lorsque la mesure de confiance d'un mapping généré est obtenue par un calcul partiel, l'administrateur peut valider ou rejeter ce mapping uniquement à partir de cette indication et de la *vue* sur l'ontologie du pair distant. Il peut également utiliser les outils présentés en section 5.3.

Nous proposons de lui permettre également de demander la pose automatique des requêtes nécessaires au calcul global de la mesure de confiance pour ce mapping. Les requêtes posées, automatiquement ou à la demande explicite de l'administrateur, sont ajoutées dans la trace et intégrées au calcul des mesure de confiance de l'ensemble des mappings générés. Ainsi, l'administrateur est responsable de toutes les requêtes supplémentaires posées.

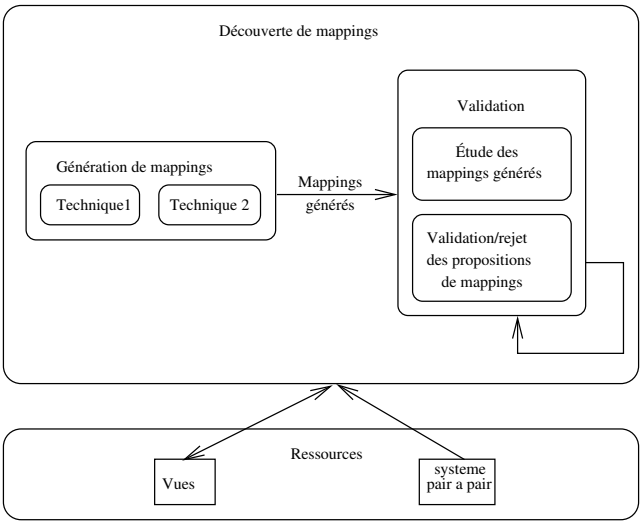

FIG. 5.8 – Chaîne complète de traitements de la découverte de mappings

Conclusion

Dans ce chapitre, nous avons montré que le contexte distribué rendait inapplicables les techniques d'alignement généralement utilisées dans la littérature. Toutefois, la spécificité du type de relations recherché et les mécanismes de raisonnement mis en œuvre dans le système pair-à-pair nous ont permis de faire des propositions de techniques menant à la découverte de mappings.

Les techniques d'alignement proposées peuvent sembler relativement simples mais elles doivent être rapprochées du fait qu'elles s'appliquent sur des couples d'éléments dont le rapprochement a été jugé *pertinent*. Cette notion de *pertinent* repose sur celle du contexte d'interpétation d'un élément se définissant comme l'ensemble des éléments avec lesquels il partage un lien. La notion de *pertinence* prend ainsi en compte la structure des ontologies. Les méthodes d'alignement proposées dans ce chapitre s'appliquent sur des couples d'éléments structurellement liés. Aucune méthode d'alignement structurelle n'est appliquée. Néanmoins, la structure des ontologies est quand même prise en compte dans le processus général d'alignement tel qu'il est présenté dans la figure 1.2 du chapitre 1 englobant l'étape de sélection des couples d'éléments à comparer.

Notre approche a été implémentée en partie dans un prototype décrit en 6.1.2 et a été partiellement validée par des expérimentations décrites en 6.2.

Chapitre 6

Réalisations et Expérimentations

Sommaire

Introduction

Dans ce chapitre, nous présentons, dans un premier temps, les différents développements logiciels réalisés dans le but de valider nos propositions. Nous nous attachons, en particulier, à la description de SpyWhere, un prototype implémentant les différentes méthodes proposées.

Dans un second temps, nous présentons les différentes expérimentations réalisées grâce à SpyWhere. Nous détaillons, pour chacune, le protocole expérimental suivi, les résultats obtenus et leur analyse.

6.1 Réalisations Logicielles

Afin de valider les méthodes proposées, plusieurs développements logiciels ont été nécessaires. Tout d'abord, nous avons implémenté la plateforme SomeRDFS décrite dans Adjiman *et al.* (2006). Nous avons ensuite implanté les algorithmes pour la découverte des raccourcis de mappings et pour la génération de nouveaux mappings dans le prototype SpyWhere.

6.1.1 SomeRDFS

Notre étude s'appuie sur la plateforme SomeRDFS. Il s'agit d'une plateforme modulaire reposant sur un noyau, SomeWhere. SomeWhere est un moteur de calcul de conséquences distribué en logique propositionnelle qui permet de charger des profils de pairs décrits en logique propositionnelle et de poser des requêtes sur des clauses écrites en logique propositionnelle. Le profil d'un pair comprend son ontologie, ses mappings et les informations nécessaires à son identification dans le réseau.

SomeRDFS est également constitué d'autres modules. Un premier module, dit de traduction, traduit des profils RDF(S) en profils exploitables par SomeWhere. Un second module, $DECA^{RDFS}$, implémente l'algorithme $DECA^{RDFS}$ qui calcule les réponses à des requêtes RDF(S) conjonctives. $DECA^{RDFS}$ traduit les requêtes RDF(S) en requêtes compréhensibles par SomeWhere en ayant recours aux services du module de traduction. Les réponses renvoyées par SomeWhere sont également traduites en RDF(S) par ce même module. Ces réponses correspondent à des réécritures. Enfin, un dernier module, dit module d'évaluation, évalue chacune des réécritures d'une requête. Les différents modules de SomeRDFS sont représentés en figure 6.1.

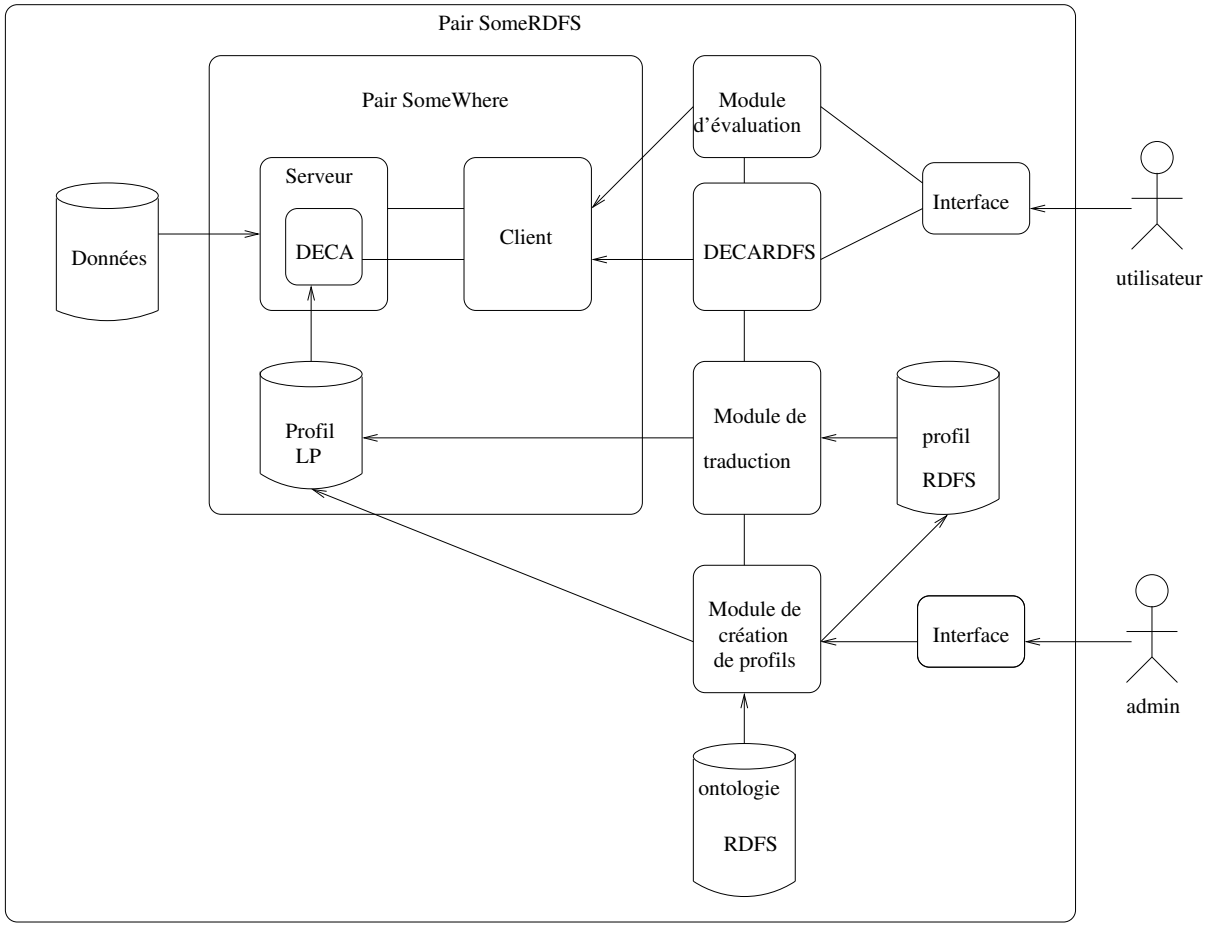

FIG. 6.1 – Architecture d'un pair du PDMS SomeRDFS

Certains de ces modules ont été implémentés par Philippe Adjiman dans le cadre de

sa thèse. C'est le cas de SomeWhere que nous avons repris. C'est le cas également du module d'évaluation mais, contrairement à SomeWhere, ce module a été modifié de façon à répondre à nos besoins. D'autres modules, ($DECA^{RDFS}$ et le module de traduction) avaient été spécifiés mais non implémentés. Nous les avons étendus et implémentés. Enfin, un nouveau module, dit de création de profils, permet la création de profils[1] à partir d'ontologies RDF(S). Le détail des modifications et extensions est présenté en annexe C. Nous avons également développé une interface graphique unifiée pour les différents modules de SomeRDFS ainsi que pour SpyWhere. Cette interface est décrite en annexe B.

6.1.2 SpyWhere

Les méthodes que nous proposons pour la découverte de mappings nécessitent un module spécifique appelé SpyWhere. Ce module se place en interface entre l'utilisateur et SomeRDFS de manière à collecter les informations utiles présentes dans les requêtes et dans leurs réponses. L'intégration de ce module est représentée en figure 6.2

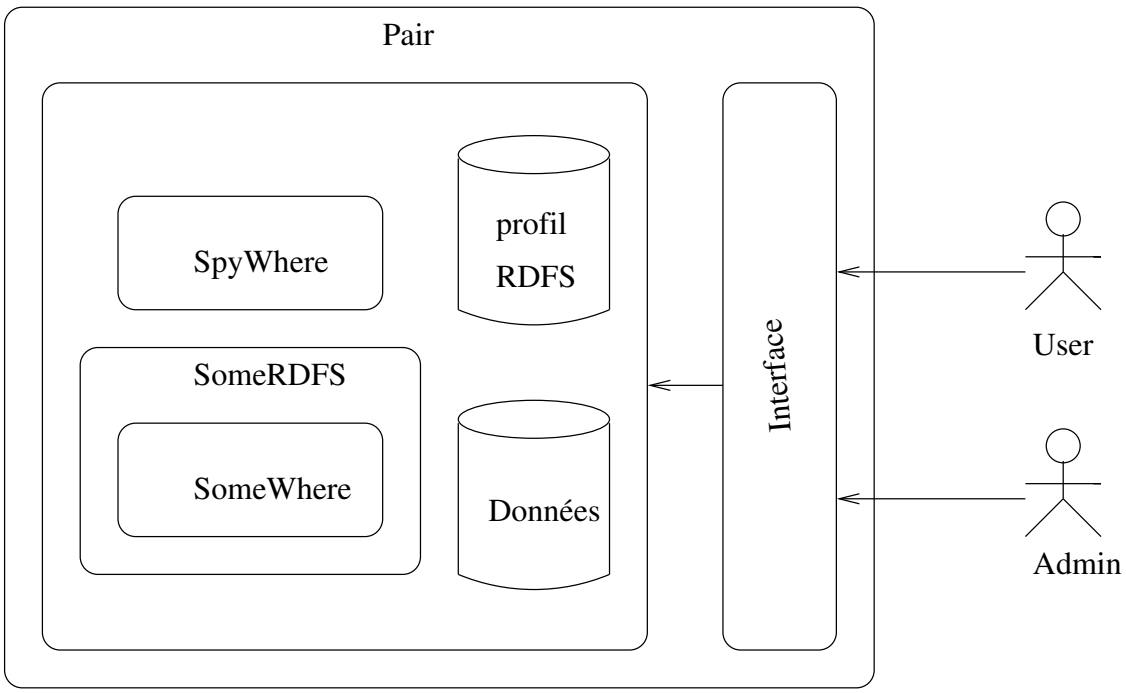

FIG. 6.2 – Architecture d'un pair doté de SpyWhere

Lorsqu'il souhaite lancer la recherche de nouveaux mappings, l'administrateur d'un pair doit activer le module SpyWhere. Toutes les requêtes des utilisateurs de ce pair sont alors redirigées vers ce module. SpyWhere transmet la requête à SomeRDFS pour obtenir les réponses. Ces réponses sont ensuite délivrées à l'utilisateur qui a posé la requête. Un composant de SpyWhere, l'analyseur de requêtes, recherche, parmi les requêtes d'utilisateur et leurs réponses, les informations exploitables pour les fonctionnalités de découverte de mappings inférables et de sélection des éléments à aligner. Nous détaillons ces deux

[1]Un profil est un fichier contenant l'identification d'un pair, son ontologie et ses mappings.

fonctionnalités (cf. figure 6.3) dans les sections suivantes.

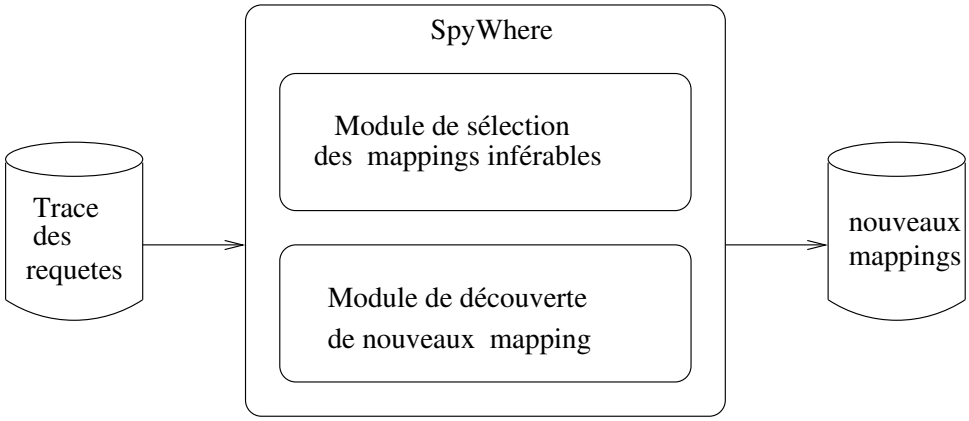

FIG. 6.3 – Fonctionnalités de SpyWhere

Nous avons modifié la modélisation de SomeRDFS en rendant accessibles les fonctionnalités de réécriture et d'évaluation unitaire (obtention des données associées à un élément) qui étaient jusque là combinées dans la fonctionnalité d'évaluation de requête. Cette modification permet de proposer à l'utilisateur deux types de requêtes : les demandes d'évaluation (requêtes originelles) et les demandes de réécritures.

Lors de l'activation de SpyWhere, l'ontologie et les mappings du pair local sont chargés dans un format interne. Cette opération permet de faciliter la manipulation des mappings existants ou inférables localement.

Module de sélection des mappings inférables

La sélection des mappings inférables décrite au chapitre 3 exploite la décomposition des mécanismes de réponse aux requêtes dans SomeRDFS en phases de réécriture et d'évaluation. Le module responsable de cette fonctionnalité reprend cette décomposition.

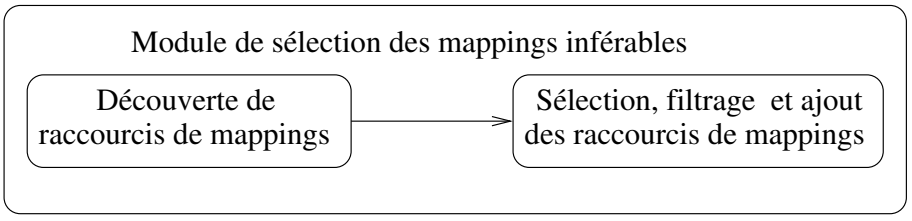

FIG. 6.4 – Module de sélection des mappings inférables

Lorsque SpyWhere est activé, le module de sélection des mappings inférables observe les requêtes posées par les utilisateurs. Lorsqu'une demande de réécritures est effectuée, l'utilisateur est sollicité pour connaître les réécritures dont il souhaite l'évaluation. Lorsque l'évaluation d'une réécriture est demandée, le *raccourci de mappings* correspondant est ajouté à une liste de mappings proposés à l'administrateur.

L'implémentation réalisée ne met pas en œuvre le filtrage des *raccourcis de mappings* dont l'automatisation présente peu d'intérêt lorsqu'il y a peu de *raccourcis de mappings* à étudier. La génération automatique de requêtes afin d'obtenir un plus grand nombre de *raccourcis de mappings* n'est pas pertinente car notre approche repose sur le fait que les utilisateurs choisissent les réécritures évaluées parce qu'ils supposent que ces réécritures correspondent à ce qu'ils recherchent. Il est difficile de simuler cette recherche automatiquement. Une autre possibilité pour obtenir un plus grand nombre de *raccourcis de mappings* est de disposer d'un réseau SomeRDFS réellement déployé (auprès de réels utilisateurs). Ne disposant pas de ce réseau, le filtrage des *raccourcis de mappings* n'est pas implémenté.

Nous avons choisi de solliciter l'utilisateur en lui demandant explicitement de quelles réécritures il souhaitait l'évaluation. Ce choix a été fait dans le but de simplifier le développement de l'interface. Cependant, ce procédé est très invasif et devra être remplacé. Une interface plus complexe pourra, par exemple, proposer à l'utilisateur la possibilité de sélectionner les réécritures à évaluer.

Module de découverte de nouveaux mappings

Le module de découverte des nouveaux mappings est composé de deux sous-modules : un sous-module de sélection des éléments *pertinents* à aligner et un sous-module de génération de mappings.

La sélection des éléments *pertinents* à aligner n'exploite que les mécanismes de réécriture des requêtes. L'identification des éléments cibles, décrite en 4.1.2, exploite les réécritures obtenues lors des réponses aux requêtes des utilisateurs. La sélection des couples d'éléments qu'il est pertinent d'aligner, décrite en 4.2, repose sur la génération automatique de requêtes dont les réécritures seront exploitées. Le module de sélection des éléments à aligner réalise ces deux étapes en s'intégrant au mieux dans SomeRDFS.

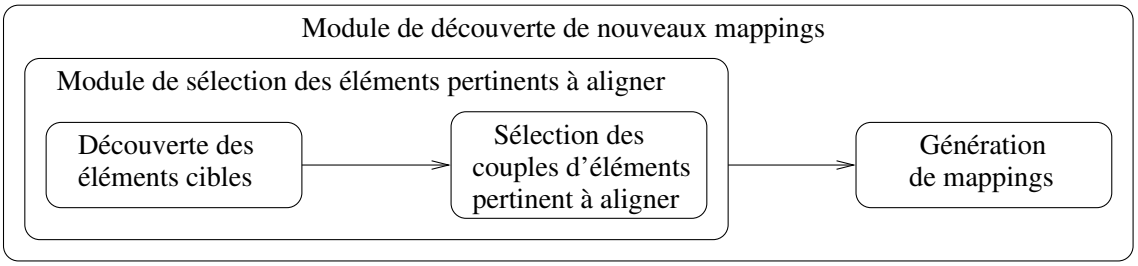

FIG. 6.5 – Module de découverte de nouveaux mappings

Pour exploiter les requêtes des utilisateurs, il est nécessaire d'obtenir les réécritures des requêtes. Dans le cas des demandes de réécritures, l'interface transmet les requêtes à SomeRDFS et à SpyWhere, puis transmet les réécritures obtenues en réponse à l'utilisateur et à SpyWhere.

Dans le cas des demandes d'évaluation, les réponses aux requêtes sont des données. Les réécritures des requêtes ne sont pas transmises à l'utilisateur. Il est donc impossible

de les obtenir via l'interface. Lorsque SpyWhere est activé, les demandes d'évaluation sont transformées en demandes de réécritures. Cependant, les utilisateurs obtiendraient alors les réécritures de leurs requêtes alors qu'ils en ont demandé les évaluations. Une solution simple serait de demander à la fois l'évaluation et la réécriture de chaque requête. Cependant, cette solution est coûteuse car l'évaluation d'une requête consiste en la réécriture de cette requête puis en l'évaluation de toutes les réécritures obtenues. Nous avons donc ajouté une fonctionnalité à SomeRDFS qui permet de spécifier un ensemble de réécritures à évaluer.

Comme pour le module de sélection des mappings inférables, nous avons considéré que chaque pair n'était interrogé que par un unique utilisateur. Le traitement des requêtes peut alors être fait en temps réel.

Nous proposons trois types de résultats :

- les éléments cibles identifiés, i.e. les éléments pour lesquels il est intéressant de trouver des mappings.
- les couples d'éléments *pertinents* à aligner
- les mappings générés

Identification des éléments cibles : L'implémentation réalisée concerne uniquement les éléments cibles élémentaires. La définition des éléments cibles recherchés est paramétrable : il est possible de choisir la fonction de comptage (méthode de comptage et stratégie) parmi celles décrites dans le tableau 4.2 du chapitre 4 ainsi que le seuil de blocage. Lors de la pose d'une requête, les éléments appartenant à l'ontologie du pair local qui apparaissent dans les réécritures sont listés. La fonction de comptage choisie est ensuite appliquée à chacun de ces éléments.

Lorsque la méthode de comptage choisie exploite les connaissances du pair local, la représentation interne des connaissances du pair local est exploitée. La structure de données a été spécialement choisie pour faciliter le comptage des mappings de spécialisation pour chaque élément quelle que soit la fonction choisie.

Lorsque la méthode de comptage choisie exploite les mécanismes de raisonnement mis en œuvre dans SomeRDFS, la fonction de comptage génère une demande de réécriture de l'élément dont elle doit évaluer le nombre de spécialisants appartenant à des ontologies de pairs distants. Les demandes de réécritures et leurs réponses respectives sont stockées localement. Ainsi, des demandes multiples de réécriture d'un même élément (par exemple, en cas de changement de stratégie ou de seuil de blocage) ne sont posées qu'une seule fois.

Sélection des couples d'éléments *pertinents* à aligner : Dans l'implémentation réalisée, les pairs d'intérêt peuvent être définis de manière globale ou locale. La construction des couples d'éléments à aligner est implémentée conformément à l'algorithme 6 décrit au chapitre 4. L'implémentation proposée exploite la représentation interne des connaissances du pair local. La structure de données a également été choisie pour permettre de connaître rapidement tous les spécialisants et tous les généralisants d'un élément. Comme pour l'identification des éléments cibles, les demandes de réécritures générées et leurs réponses respectives sont stockées.

La représentation interne est également exploitée afin de filtrer les couples d'éléments entre lesquels il existe déjà un mapping ou pour lesquels un mapping peut être inféré localement.

Pour chaque couple d'éléments, il est possible de connaître le type de mapping à rechercher (subsomption, typage de domaine ou typage de portée) et le scénario (généralisant commun ou spécialisant commun) à l'origine de la proposition de ce couple.

Les différents critères de filtrage présentés en section 4.3 sont implémentés et totalement fonctionnels. Il est donc possible de filtrer selon la stratégie sur les pairs, selon le type de mapping recherché (généralisation ou spécialisation) et selon la qualité supposée des mappings existants.

Génération de mappings : L'implémentation réalisée pour la génération de mappings est moins aboutie que les travaux que nous avons présentés au chapitre 5. Il s'agit davantage d'un cadre permettant d'intégrer facilement les différentes techniques proposées.

La technique déducto-syntactique est partiellement implémentée : nous avons implémenté les techniques exploitant uniquement les mécanismes de raisonnement (cf. 5.1.1). La combinaison mécanismes de raisonnement et techniques d'alignement syntaxiques n'a pas pu être complètement développée, faute de temps, bien que certaines mesures terminologiques (Levenshtein et Substring) aient été intégrées.

La technique inductive est totalement implémentée.

6.2 Démarche Expérimentale

6.2.1 Test de la méthode de sélection des couples d'éléments à aligner

Afin d'évaluer cette partie de notre approche, nous avons réalisé des expérimentations à partir d'ontologies du domaine des publications. Dans un premier temps, nous nous sommes particulièrement intéressé à l'applicabilité de notre approche lorsque les pairs sont connectés par un nombre réduit de mappings. Dans un deuxième temps, nous nous sommes intéressé à l'applicabilité de notre approche dans un réseau composé d'un nombre conséquent de pairs. Dans les deux cas, notre objectif était d'évaluer le processus d'identification d'éléments cibles élémentaires et des couples d'éléménts qu'il est *pertinent* d'aligner, ainsi que les différentes stratégies relatives aux pairs avec lesquels des mappings sont recherchés, exploitées dans ce processus. Cette évaluation tient compte de la quantité des couples d'éléments proposés mais aussi de la pertinence de ces couples.

Ontologies

Dans les premières expérimentations, nous avons construit un réseau de quatre pairs. Chaque pair possède sa propre ontologie du domaine des publications. Les caractéristiques des différentes ontologies sont données dans le tableau 6.1. Nous avons choisi des ontologies issues du Web, modélisant différents points de vues sur le domaine des publications et dont la taille et la granularité diffèrent. Les ontologies des pairs $\mathcal{P}_1$, $\mathcal{P}_3$ et $\mathcal{P}_4$ décrivent principalement des conférences (événements, publications et personnes). *Publication* est la racine de ces trois ontologies mais les publications sont décrites plus

précisément dans l'ontologie de $\mathcal{P}_4$: 47 % des éléments de l'ontologie sont des spécialisants de *Publication*. L'ontologie de $\mathcal{P}_2$ est sensiblement différente des autres ontologies. Elle décrit les documents d'une bibliothèque. L'élément à la racine est *Documentation* et l'élément *Publication* en est un spécialisant direct.

Pair	Ontologie	# éléments	# spécialisants de *Publication*
$\mathcal{P}_1$	http://lsdis.cs.uga.edu/proj/semdis/testbed/	172	14
$\mathcal{P}_2$	webode://droz.dia.fi.upm.es/Documentation+Ontology	95	26
$\mathcal{P}_3$	http://139.91.183.30:9090/RDF/VRP/Examples/rdf.rdf	144	36
$\mathcal{P}_4$	http://swrc.org/swrc.rdfs	231	110

TAB. 6.1 – Description des ontologies

1ère experimentation : Étude de la méthode de sélection de couples d'éléments pertinents à aligner selon la stratégie par défaut dans un réseau de 4 pairs

Description Nous avons, tout d'abord, ajouté six mappings, décrits dans le tableau 6.2 aux connaissances de $\mathcal{P}_1$. Tous les pairs sont ainsi connectés les uns aux autres soit directement, soit indirectement. Seul $\mathcal{P}_1$ possède des mappings avec tous les pairs. Pour connaître les éléments cibles et les couples d'éléments à aligner, nous avons posé plusieurs requêtes sur chaque pair de sorte que tous les éléments soient des *points d'intérêt*. Le seul critère de sélection des éléments cibles appliqué est alors le critère basé sur les points de blocage.

$\mathcal{P}_2$:*Publication*$(X) \Rightarrow \mathcal{P}_1$:*Publication*$(X)$	$\mathcal{P}_1$:*Publication*$(X) \Rightarrow \mathcal{P}_2$:*Publication*$(X)$
$\mathcal{P}_3$:*Publication*$(X) \Rightarrow \mathcal{P}_1$:*Publication*$(X)$	$\mathcal{P}_1$:*Publication*$(X) \Rightarrow \mathcal{P}_3$:*Publication*$(X)$
$\mathcal{P}_4$:*Publication*$(X) \Rightarrow \mathcal{P}_1$:*Publication*$(X)$	$\mathcal{P}_1$:*Publication*$(X) \Rightarrow \mathcal{P}_4$:*Publication*$(X)$

TAB. 6.2 – Mappings ajoutés

Résultats et analyse Le tableau 6.3 présente le nombre d'éléments cibles obtenus avec un seuil $t = 1$, selon chacune des fonctions de comptage. Du fait du faible nombre de mappings initiaux, la plupart des éléments sont des éléments cibles (cf. # éléments cibles dans le tableau 6.3). Les éléments cibles faisant partie des couples d'éléments qu'il est pertinent d'aligner représentent entre 8 et 92% des éléments d'une ontologie (cf. # Éléments cibles exploités dans le tableau 6.3). La variation de ce nombre dépend de la taille du sous-arbre auquel appartient l'élément *Publication*. D'après ces résultats, la sélection des éléments cibles selon le critère de *point d'intérêt* semble parfois inutile. Par exemple, pour $\mathcal{P}_1$, sans appliquer ce critère, il n'y a que 14 éléments cibles exploités sur les 172 éléments de l'ontologie. Cependant, ce cas est dû au faible nombre de mappings. Lorsque les mappings initiaux sont assez nombreux, la plupart des éléments cibles sont exploités comme nous le montrons dans la 3e expérimentation.

Dans ce cas simple, si on ne tient pas compte des contraintes du formalisme de façon à faire un calcul rapide et simple, chaque élément cible est comparé avec chacun des spécialisants de l'élément *Publication* appartenant à l'ontologie de chacun des pairs distants (cf. # Éléments distants pertinents dans le tableau 6.3). Il est alors facile de calculer le nombre de couples d'éléments sélectionnés. Par exemple, chacun des 36 éléments cibles

Pair	# Éléments	# Éléments distants	# Éléments cibles	# Éléments cibles exploités	# Éléments distants pertinents par élément cible exploité
$\mathcal{P}_1$	172	470	171	14	172
$\mathcal{P}_2$	95	547	93	87	160
$\mathcal{P}_3$	144	498	143	36	150
$\mathcal{P}_4$	231	411	230	110	76

TAB. 6.3 – Nombre d'éléments cibles obtenus

exploités de $\mathcal{P}_3$ sera comparé avec les 150 éléments distants pertinents appartenant aux ontologies des autres pairs. Sans ce processus de sélection des couples d'éléments à aligner, chacun des 144 éléments de $\mathcal{P}_3$ aurait été comparé aux 498 éléments (172 + 95 + 231) appartenant aux ontologies des autres pairs. Le nombre de couples d'éléments à aligner obtenus lorsqu'on applique notre sélection correspond alors à 7.5% du nombre de couples d'éléments à aligner obtenus sans sélection. De plus, lorsqu'on tient compte des contraintes du formalisme, le nombre de couples d'éléments sélectionnés est encore plus faible (cf. tableau 6.4 dans lequel on indique le nombre de couples d'éléments à aligner selon la sélection appliquée).

Pair	sans sélection [1]	sélection sans prise en compte du formalisme	sélection avec prise en compte du formalisme [2]	[2]/[1]
$\mathcal{P}_1$	172×(95+144+231) = 80840	14×(26+36+110) = 2408	1333	1,64%
$\mathcal{P}_2$	95×(172+144+231) = 51965	87×(14+36+110) = 13920	1674	3,22%
$\mathcal{P}_3$	144×(172 + 95 + 231) = 71712	36×(14+26+110) = 5400	2449	3,41%
$\mathcal{P}_4$	213×(172+95+144) = 94941	110×(14+26+36) = 8360	3522	3,71%

TAB. 6.4 – Comparatif du nombre de couples d'éléments à aligner selon la sélection appliquée

Par ailleurs, le nombre de couples d'éléments sélectionnés peut encore être réduit par application d'une stratégie. Par exemple, l'application de la stratégie de rapprochement pour $\mathcal{P}_3$ fera que seuls les éléments appartenant à l'ontologie de $\mathcal{P}_1$ seront considérés. Ainsi, chacun de ses 36 éléments cibles exploités seront comparés aux 14 éléments distants pertinents appartenant à l'ontologie de $\mathcal{P}_1$. Avec prise en compte du formalisme, on obtient alors seulement 325 couples d'éléments à aligner. Ceci montre l'intérêt et la nécessité du processus global de sélection menant à l'ensemble des couples d'éléments *pertinents* à aligner.

2nde expérimentation : Étude de la méthode de sélection de couples d'éléments pertinents à aligner dans un réseau de 4 pairs - Évaluation des stratégies

Description Nous avons ensuite réalisé un second test en ajoutant 7 nouveaux mappings (cf. tableau 6.5) aux connaissances de $\mathcal{P}_1$. Il s'agit de mappings de spécialisation pour quatre spécialisants directs de *Publication*. Ces nouveaux mappings mettent en jeu des éléments appartenant aux ontologies de $\mathcal{P}_3$ et $\mathcal{P}_4$. Nous avons également ajouté 2 nouveaux mappings aux connaissances de $\mathcal{P}_3$. Il s'agit de mappings de spécialisation pour des éléments apparaissant dans les mappings de $\mathcal{P}_1$. Ces deux mappings sont des liens avec l'ontologie de $\mathcal{P}_2$.

Nous nous sommes focalisé sur les 14 éléments cibles de $\mathcal{P}_1$ faisant partie des couples d'éléments qu'il est pertinent d'aligner. Nous avons donc posé une requête construite à

$\mathcal{P}_4$:$Book(X) \Rightarrow \mathcal{P}_1$:$Book(X)$	$\mathcal{P}_3$:$ConferencePaper(X) \Rightarrow \mathcal{P}_1$:$Conference(X)$
$\mathcal{P}_3$:$book(X,Y) \Rightarrow \mathcal{P}_1$:$Book(X)$	$\mathcal{P}_2$:$Article\_in\_Conference(X) \Rightarrow \mathcal{P}_3$:$ConferencePaper(X)$
$\mathcal{P}_2$:$Book(X) \Rightarrow \mathcal{P}_3$:$Book(X)$	$\mathcal{P}_3$:$Book(X) \Rightarrow \mathcal{P}_1$:$Book(X)$
$\mathcal{P}_3$:$journal(X,Y) \Rightarrow \mathcal{P}_1$:$published\_in(X,Y)$	$\mathcal{P}_3$:$book(X,Y) \Rightarrow \mathcal{P}_1$:$published\_in(X,Y)$
$\mathcal{P}_3$:$Journal(X) \Rightarrow \mathcal{P}_1$:$Journal(X)$	

TAB. 6.5 – Mappings ajoutés

partir de l'élément $\mathcal{P}_1$:*Publication* et nous avons demandé les éléments cibles puis les couples d'éléments pertinents à aligner.

Résultats et analyse Les figures 6.6, 6.7, 6.8 et 6.9 présentent l'évolution du nombre d'éléments cibles en fonction du seuil choisi. Chaque figure correspond à une stratégie donnée. Sur une même figure, chaque courbe correspond à une méthode de comptage. C1 correspond à la méthode de comptage exploitant les connaissances du pair local. C2 correspond à la méthode de comptage exploitant les mécanismes de raisonnement.

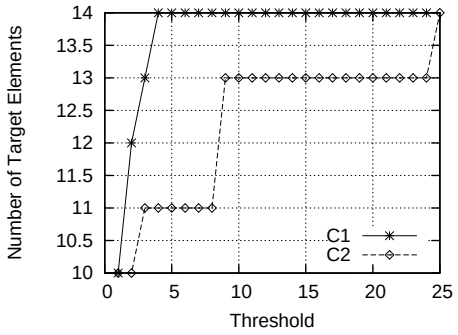

FIG. 6.6 – Nombre d'éléments cibles avec la stratégie par défaut

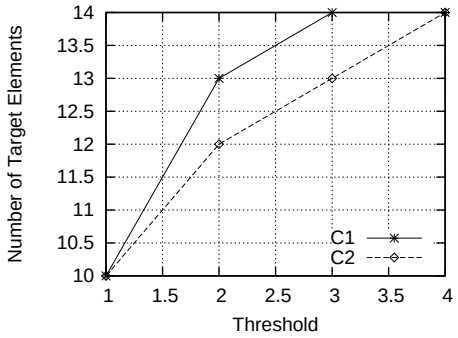

FIG. 6.7 – Nombre d'éléments cibles avec la stratégie de rupture de l'isolement

On observe, dans tous les cas, un nombre d'éléments cibles minimal de 10. Ce nombre correspond aux 10 spécialisants de l'élément *Publication* dans l'ontologie du pair $\mathcal{P}_1$ pour lesquels aucun mapping n'a été ajouté. Lorsque le seuil est suffisamment important (seuil de 25 dans tous les cas), les 14 éléments sont des éléments cibles. En faisant varier le seuil, pour une stratégie et une méthode de comptage données, il est possible d'obtenir différents ensembles d'éléments cibles. Le tableau 6.6 détaille les différents ensembles d'éléments cibles qu'il est possible d'obtenir pour chaque stratégie et chaque méthode de

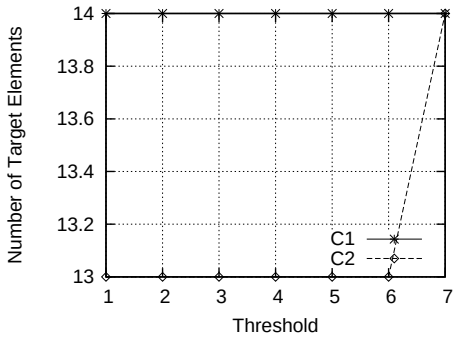

FIG. 6.8 – Nombre d'éléments cibles avec la stratégie de rapprochement 1

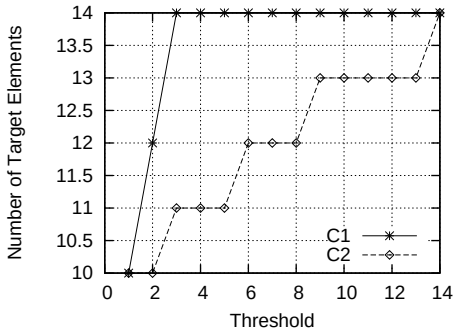

FIG. 6.9 – Nombre d'éléments cibles avec la stratégie de rapprochement 2

comptage.

À partir des quatre éléments qui ne sont pas des éléments cibles avec un seuil de 1, il est possible de construire 15 ($= C_4^4 + C_4^3 + C_4^2 + C_4^1$) ensembles. Dans l'expérimentation, on remarque que la stratégie par défaut permet d'obtenir 4 de ces ensembles grâce aux deux méthodes de comptage. Les stratégies de rupture de l'isolement et de rapprochement_2 permettent, chacune, d'obtenir un ensemble supplémentaire (cf. tableau 6.6).

De plus, les ensembles d'éléments cibles peuvent être différents selon le niveau d'application (local ou global) de la stratégie. Nous rappelons que le niveau d'application global consiste à définir, pour le pair local, un ensemble unique de pairs d'intérêt pour tous les éléments de son ontologie. Le niveau d'application local consiste à définir, pour le pair local, un ensemble de pairs d'intérêt spécifique pour chacun des éléments de son ontologie.

La définition des éléments cibles fait intervenir la notion de pair d'intérêt (cf. 4.3.1). Le niveau d'application de la stratégie modifie les pairs d'intérêt et donc les ensembles d'éléments cibles obtenus.

Pour les quatre premières lignes du tableau 6.6, les pairs d'intérêt ont été calculés de façon globale. Il existe un unique mapping entre les éléments *Publication* des pairs $\mathcal{P}_1$ et $\mathcal{P}_2$. $\mathcal{P}_2$ est donc un pair d'intérêt pour $\mathcal{P}_1$, selon les stratégies de rapprochement. En revanche, si ces stratégies sont appliquées au niveau local, $\mathcal{P}_2$ n'est pas nécessairement un pair d'intérêt pour tous les éléments de l'ontologie de $\mathcal{P}_1$:

– Avec la méthode de comptage exploitant les connaissances du pair local (C1), aucun spécialisant de l'élément *Publication* dans l'ontologie du pair $\mathcal{P}_1$ ne possède de mapping avec un élément de l'ontologie de $\mathcal{P}_2$. $\mathcal{P}_2$ n'est donc un pair d'intérêt pour aucun élément de l'ontologie de $\mathcal{P}_1$. Les ensembles d'éléments cibles obtenus pourront donc être différents par rapport à ceux obtenus par l'application des stratégies au niveau global (cf. stratégie de rapprochement_1 dans le tableau 6.6).
– Avec la méthode de comptage exploitant les mécanismes de raisonnement (C2), les éléments *Book* et $Conference$ possèdent des spécialisants appartenant à l'ontologie du pair $\mathcal{P}_2$. $\mathcal{P}_2$ peut donc être un pair d'intérêt pour ces deux éléments. Il existe également d'autres éléments qui ne possèdent pas de spécialisants appartenant à l'ontologie du pair $\mathcal{P}_2$. Pour ces éléments, $\mathcal{P}_2$ n'est pas un pair d'intérêt. Les ensembles d'éléments cibles obtenus pourront donc être différents par rapport à ceux obtenus par l'application des stratégies au niveau global (cf. stratégie de rapprochement_1 dans le tableau 6.6).

Les figures 6.10 et 6.11 présentent l'évolution du nombre d'éléments cibles en fonction du seuil choisi pour les stratégies de rapprochement appliquées au niveau local.

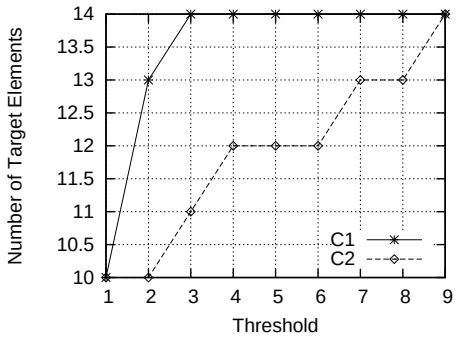

FIG. 6.10 – Nombre d'éléments cibles avec la stratégie de rapprochement 1 locale

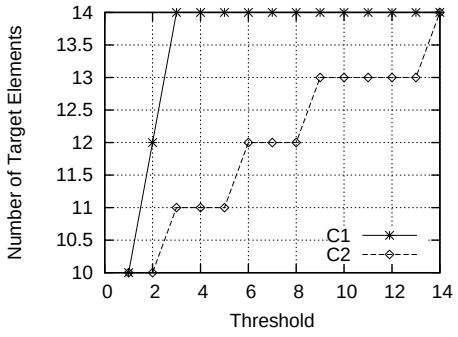

FIG. 6.11 – Nombre d'éléments cibles avec la stratégie de rapprochement 2 locale

L'application de la stratégie de rapprochement 1 locale permet d'obtenir deux nouveaux ensembles d'éléments cibles. Ceci est permis par la prise en compte du pair $\mathcal{P}_2$ pour un élément donné, uniquement s'il existe un lien de spécialisation entre cet élément et un élément appartenant à l'ontologie de $\mathcal{P}_2$.

On constate que la stratégie de rapprochement 2 permet d'obtenir exactement les mêmes ensembles d'éléments cibles par application d'une stratégie locale et d'une stratégie globale. Ceci s'explique par le fait que la valeur retenue est un maximum. Cette valeur n'est pas affectée lorsqu'on ajoute un pair avec lequel aucun lien sémantique n'est partagé.

	C_1	C_2
stratégie par défaut	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*}	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*} {*PublishedIn*}
stratégie de rupture de l'isolement	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*}	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*} {*Journal*, *PublishedIn*}
stratégie de rapprochement 1 globale	{*Book*, *Conference*, *Journal*, *PublishedIn*}	{*Book*, *Conference*, *Journal*, *PublishedIn*}
stratégie de rapprochement 2 globale	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*}	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*} {*Conference*, *PublishedIn*} {*PublishedIn*}
stratégie de rapprochement 1 locale	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Book*, *Conference*, *Journal*}	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Book*, *Conference*, *PublishedIn*} {*Conference*, *PublishedIn*} {*PublishedIn*}
stratégie de rapprochement 2 locale	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*}	{*Book*, *Conference*, *Journal*, *PublishedIn*} {*Conference*, *Journal*, *PublishedIn*} {*Conference*, *PublishedIn*} {*PublishedIn*}

TAB. 6.6 – Ensembles d'éléments cibles obtenus par chaque stratégie

On observe que la stratégie de rupture de l'isolement désigne tous les éléments comme éléments cibles pour un seuil faible (3 ou 4 selon la méthode de comptage). Ce phénomène est dû à la taille réduite du réseau. Un élément ne peut pas avoir des spécialisants dans plus d'ontologies de pairs distincts qu'il n'existe de pairs dans le réseau. Or, dans cette expérimentation, le réseau est composé de 4 pairs.

On observe le même phénomène lorsque les stratégies reposent sur la méthode de comptage exploitant les mappings. Dans ce cas c'est le faible nombre de mappings qui en est la cause.

Dans cette expérimentation, nous avons également étudié les couples d'éléments pertinents à aligner selon les différentes stratégies appliquées. À partir des éléments cibles, on génère des couples d'éléments qu'il est pertinent d'aligner. Les figures 6.12, 6.13, 6.14, 6.15, 6.16 et 6.17 présentent l'évolution du nombre de couples d'éléments pertinents à aligner en fonction du seuil choisi. Chaque figure correspond à une stratégie donnée. Sur une même figure, chaque courbe correspond à une méthode de comptage. C1 correspond à la méthode de comptage exploitant les connaissances du pair local. C2 correspond à la méthode de comptage exploitant les mécanismes de raisonnement.

On observe, tout d'abord, qu'on retrouve, dans tous les cas, sauf pour les deux stratégies locales, 1720 couples d'éléments à aligner pour un seuil de 1. Il s'agit d'un même ensemble de couples d'éléments constitué des couples faisant intervenir l'un des 10 éléments pour lesquels il n'existe aucun spécialisant dans l'ontologie d'un pair distant. Pour ces éléments, l'ajout d'un nouveau mapping de spécialisation, quel qu'il soit, a pour consé-

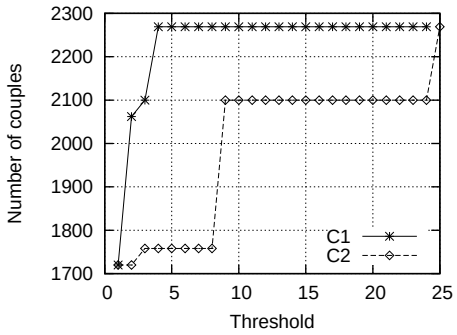

FIG. 6.12 – Nombre de couples d'éléments pertinents à aligner avec la stratégie par défaut

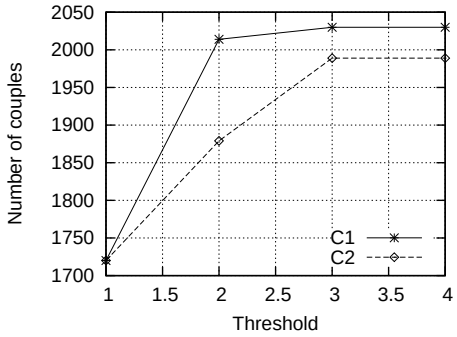

FIG. 6.13 – Nombre de couples d'éléments pertinents à aligner avec la stratégie de rupture de l'isolement

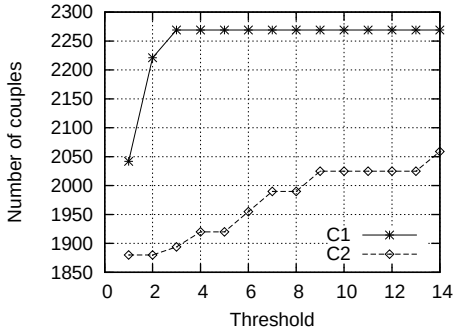

FIG. 6.14 – Nombre de couples d'éléments pertinents à aligner avec la stratégie de rapprochement 1

quence l'augmentation du nombre de spécialisants (pris en compte par la stratégie par défaut), l'augmentation du nombre de pairs dont l'ontologie contient un spécialisant (pris en compte par la stratégie de rupture de l'isolement) et l'augmentation du nombre de spécialisants par pairs (pris en compte par les stratégies de rapprochement). Il est donc intéressant quelle que soit la stratégie.

Pour les stratégies locales, lorsqu'il n'existe pas de spécialisants pour un élément, il n'existe a fortiori pas de pairs se partageant ces spécialisants. Il n'existe alors pas de pairs d'intérêt avec lesquels il est intéressant de trouver de nouveaux mappings. C'est pourquoi on ne trouve pas de couples d'éléments pertinents à aligner pour les éléments n'ayant pas

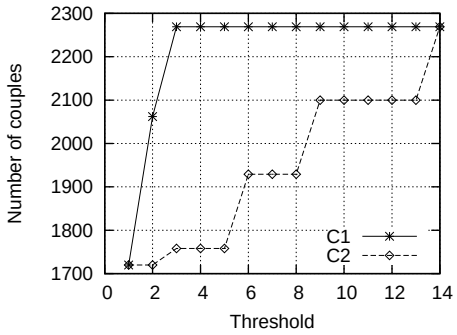

FIG. 6.15 – Nombre de couples d'éléments pertinents à aligner avec la stratégie de rapprochement 2

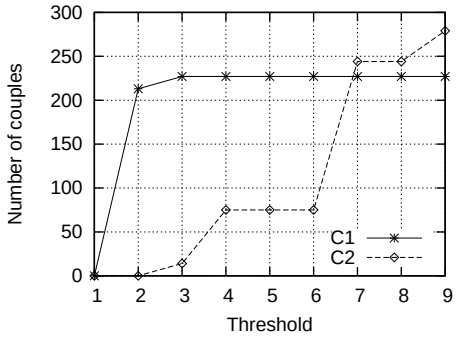

FIG. 6.16 – Nombre de couples d'éléments pertinents à aligner avec la stratégie de rapprochement 1 locale

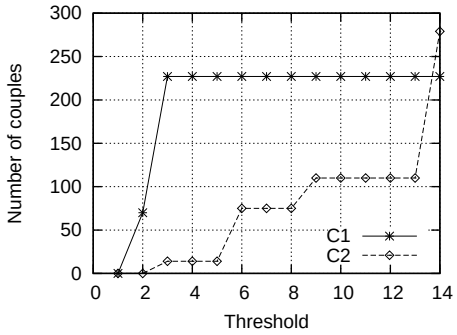

FIG. 6.17 – Nombre de couples d'éléments pertinents à aligner avec la stratégie de rapprochement 2 locale

de spécialisant avec les stratégies de rapprochement locales.

Nous avons vu précédemment que chaque stratégie permet d'obtenir des ensembles d'éléments cibles que les autres stratégies ne permettent pas d'obtenir. De ce fait, les ensembles de couples d'éléments pertinents à aligner correspondant à ces ensembles d'éléments cibles spécifiques sont également spécifiques à chaque stratégie.

Un même ensemble d'éléments cibles peut être obtenu par application de différentes stratégies. Dans ce cas, les ensembles de couples d'éléments pertinents à aligner correspon-

dant mettent en jeu des éléments appartenant aux ontologies des pairs d'intérêt. Or, les pairs d'intérêt peuvent différer selon la stratégie appliquée. Le tableau 6.7 donne, pour les quatre éléments cibles *Book*, *Conference*, *Journal*, *PublishedIn*, les pairs avec lesquels les mappings sont recherchés pour chaque stratégie. Le nombre de couples correspondant à chaque élément de l'ontologie locale est indiqué entre parenthèses.

		Book	*Conference*	*Journal*	*PublishedIn*
stratégie par défaut		$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (169)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (38)
stratégie de rupture de l'isolement	C_1	$\mathcal{P}_2$ (26)	$\mathcal{P}_2$, $\mathcal{P}_4$ (135)	$\mathcal{P}_2$, $\mathcal{P}_4$ (135)	$\mathcal{P}_2$, $\mathcal{P}_4$ (24)
	C_2	(0)	$\mathcal{P}_4$ (110)	$\mathcal{P}_2$, $\mathcal{P}_4$ (135)	$\mathcal{P}_2$, $\mathcal{P}_4$ (24)
stratégie de rapprochement 1 globale	C_1	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (135-169)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (135-171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (135-171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (26-38)
	C_2	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (0-169)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (0-171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (26-171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (24-38)
stratégie de rapprochement 2 globale	C_1	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (169)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (38)
	C_2	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (169)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (171)	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (38)
stratégie de rapprochement 1 locale	C_1	$\mathcal{P}_3$, $\mathcal{P}_4$* (0-143)	$\mathcal{P}_3$* (0-35)	$\mathcal{P}_3$* (0-35)	$\mathcal{P}_3$* (0-14)
	C_2	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$* (0-169)	$\mathcal{P}_2$, $\mathcal{P}_3$* (0-61)	$\mathcal{P}_3$* (0-35)	$\mathcal{P}_3$* (0-14)
stratégie de rapprochement 2 locale	C_1	$\mathcal{P}_3$, $\mathcal{P}_4$ (143)	$\mathcal{P}_3$ (35)	$\mathcal{P}_3$ (35)	$\mathcal{P}_3$ (14)
	C_2	$\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ (169)	$\mathcal{P}_2$, $\mathcal{P}_3$ (61)	$\mathcal{P}_3$ (35)	$\mathcal{P}_3$ (14)

TAB. 6.7 – Pairs participant aux couples d'éléments à aligner selon la sélection appliquée

Pour les stratégies de rapprochement_1, les pairs d'intérêt dépendent aussi du seuil choisi pour la sélection des éléments cibles. Dans le tableau 6.7, les valeurs indiquées entre parenthèses sont les nombres minimum et maximum de couples correspondant à chaque élément de l'ontologie locale. Les ensembles de pairs, pour ces stratégies, sont signalés par un "*" car ils correspondent à l'ensemble maximal des pairs d'intérêt obtenu pour un seuil suffisamment grand. Ce seuil dépend de la méthode de comptage et du niveau d'application de la stratégie. Dans cette expérimentation, un seuil de 15 est suffisamment grand pour toutes les stratégies de rapprochement_1. Pour un seuil inférieur il est possible d'obtenir d'autres ensembles de pairs d'intérêt que ceux indiqués. Par exemple, avec un seuil de 2, un niveau d'application local et la méthode de comptage exploitant les connaissances du pair local, $\mathcal{P}_4$ est le seul pair d'intérêt pour *Book*.

Pour les stratégies de rapprochement globales, tous les pairs du réseau sont des pairs d'intérêt pour tous les éléments appartenant à l'ontologie de $\mathcal{P}_1$. Il n'y a donc aucune sélection pour ces stratégies par rapport à la stratégie par défaut qui consiste à rechercher des mappings avec tous les pairs quels qu'ils soient. Ceci est dû à une particularité du réseau utilisé pour cette expérimentation : $\mathcal{P}_1$ partage au moins un mapping avec chaque pair du système. S'il existait un pair qui ne partage pas de mapping avec $\mathcal{P}_1$ alors, les stratégies par défaut et de rupture de l'isolement conduiraient à obtenir des couples faisant intervenir un élément appartenant à l'ontologie de ce pair tandis que l'ensemble des couples obtenus par application des stratégies de rapprochement ne seraient pas modifiés.

Cette expérimentation montre la pertinence des stratégies qui permettent de nombreuses variations sur les sélections d'ensembles de couples d'élements pertinents à aligner à partir d'un nombre restreint de paramètres (le seuil, la méthode de comptage et la stratégie). Aucune connaissance n'est nécessaire à l'administrateur pour choisir le paramétrage. Une fois la stratégie choisie, les autres paramètres peuvent être choisis par une succession de tentatives.

3ème expérimentation : Passage à l'échelle - Étude des éléments cibles

Contexte d'expérimentation Nous avons également souhaité évaluer notre approche dans un réseau de pairs plus important. Notre objectif était d'avoir des données statistiques sur tous les pairs. Ne disposant pas d'un tel réseau, nous l'avons créé.

Nous avons utilisé un réseau de 100 pairs créé sur un modèle de Watts et Strogatz (Watts & Strogatz (1998)) de degré 10 pour garantir les propriétés de *petit monde* attendues pour les réseaux SomeRDFS. Ce modèle permet de déterminer quels sont les pairs qui partagent des mappings sans fixer le nombre de mappings qu'ils partagent. Afin que tous les pairs n'aient pas tous le même nombre de mappings, nous avons décidé que deux pairs partagent 20 mappings (10 mappings dans chaque sens) si la différence, modulo 100 (le nombre de pairs), entre leurs indices respectifs est inférieure à 5. Dans les autres cas, ils partagent 10 mappings (5 dans chaque sens).

Après avoir défini la topologie précise du réseau, nous avons créé les pairs qui le constituent. Pour créer un pair il est nécessaire de disposer d'une ontologie. Pour notre réseau, il est donc nécessaire de disposer de 100 ontologies et de l'ensemble des mappings entre ces ontologies. Pour l'observation que nous souhaitons faire, il n'est pas nécessaire que chaque pair dispose de sa propre ontologie. Nous avons donc créé 100 pairs à partir de l'ontologie *http://lsdis.cs.uga.edu/proj/semdis/testbed/* dont les caractéristiques sont données dans le tableau 6.1. En revanche, chaque pair dispose de ses propres mappings générés aléatoirement. Tous les mappings générés correspondent à un lien existant ou inférable dans l'ontologie de départ. Chaque pair possède alors entre 140 et 220 mappings.

Description Nous avons posé plusieurs requêtes sur chaque pair de sorte que tous les éléments soient des *points d'intérêt*. Nous avons ensuite demandé, pour chaque pair, la liste des éléments cibles pour chaque fonction de comptage en faisant varier le seuil de blocage entre 1 et 50 pour chaque stratégie. Le seuil maximal devait être assez grand pour que tous les éléments soient élément cible avec ce seuil.

Résultats et analyse Les figures 6.18, 6.19, 6.20 et 6.21 présentent l'évolution du nombre d'éléments cibles en fonction du seuil choisi. Chaque figure correspond à une stratégie donnée. Sur une même figure, chaque courbe correspond à une méthode de comptage. C1 correspond à la méthode de comptage exploitant les connaissances du pair local. C2 correspond à la méthode de comptage exploitant les mécanismes de raisonnement. Les valeurs présentées sont des valeurs moyennes sur les 100 pairs composant les

réseau.

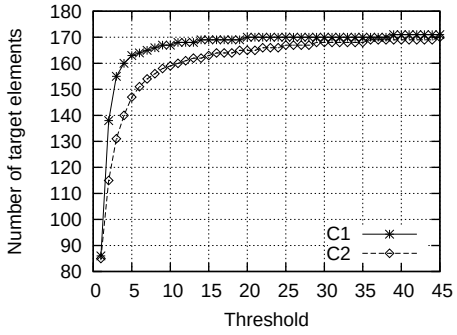

FIG. 6.18 – Nombre d'éléments cibles avec la stratégie par défaut

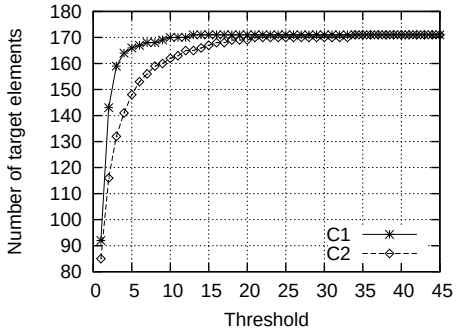

FIG. 6.19 – Nombre d'éléments cibles avec la stratégie de rupture de l'isolement

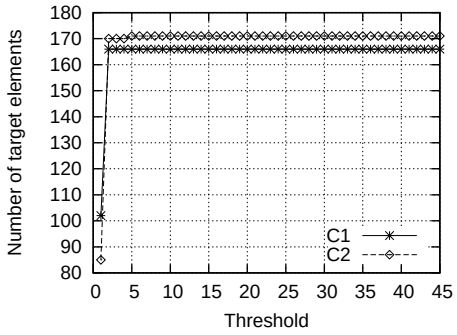

FIG. 6.20 – Nombre d'éléments cibles avec la stratégie de rapprochement 1

On observe, dans tous les cas, un nombre d'éléments cibles minimal de 80. Il s'agit d'éléments pour lesquels il n'existe, dans les connaissances du pair, aucun spécialisant appartenant à l'ontologie d'un autre pair. Le fait qu'il existe un si grand nombre minimal d'éléments cibles montre l'intérêt de la définition des éléments cibles par rapport à la notion de *point d'intérêt* même lorsqu'il existe un nombre relativement important de mappings initiaux.

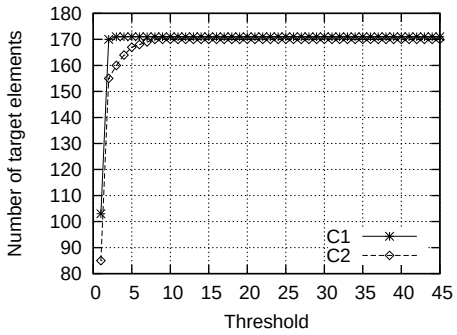

FIG. 6.21 – Nombre d'éléments cibles avec la stratégie de rapprochement 2

Les courbes 6.18 et 6.19 se ressemblent beaucoup. Cependant, l'application de la stratégie de rupture de l'isolement permet d'obtenir des ensembles d'éléments cibles qui ne peuvent pas être obtenus par application de la stratégie par défaut. Par exemple, il est possible d'obtenir un ensemble de 90 éléments cibles qui n'est pas obtenu avec la stratégie par défaut. Notons que, lorsqu'on observe chaque pair individuellement, les variations sur les ensembles d'éléments cibles sont plus prononcées. De plus, l'intersection et la différence de deux ensembles d'éléments cibles ayant des tailles proches sont non vides lorsque ces ensembles sont générés par application de stratégies différentes.

Pour les deux stratégies de rapprochement, les courbes montrent que tous les éléments sont des éléments cibles avec un seuil assez faible (entre 2 et 7) tandis que ce résultat est obtenu avec un seuil minimal de 20 pour la stratégie de rupture de l'isolement et un seuil minimal de 40 pour la stratégie par défaut.

Les stratégies de rapprochement ont pour objectif d'identifier les pairs distants avec lesquels un nombre minimum de liens sémantiques sont déjà partagés. Dans cette expérimentation, ces pairs sont définis localement pour chaque élément de l'ontologie du pair local. Nos mesures reflètent qu'il existe peu de spécialisants par pair pour chacun des éléments de l'ontologie du pair local (maximum 7 en moyenne).

Cette expérimentation montre que bien que l'administrateur puisse paramétrer la sélection des éléments cibles, il est nécessaire que ce paramétrage soit en accord avec les connaissances de l'ensemble du système. Si un seuil trop important est choisi, aucune sélection ne sera faite sur les éléments cibles.

4ème expérimentation : Passage à l'échelle - Étude des couples d'élements pertinents à aligner

Contexte d'expérimentation Le contexte d'expérimentation est le même que dans la 3ème expérimentation.

Description Du fait de la particularité de notre réseau où tous les pairs utilisent la même ontologie, nous avons pu mettre en place une validation automatique des couples d'éléments à aligner. Un couple est valide s'il est composé de deux éléments pour lesquels il existe un lien (explicitement représenté ou inférable) dans l'ontologie de départ.

Notre objectif était d'étudier, comme dans la 1ère expérimentation, le taux de sélection des couples d'éléments pertinents à aligner par notre méthode. Ces deux expérimentations se distinguent par la taille du réseau exploité. Nous souhaitions mettre en évidence le problème de passage à l'échelle. Comme dans la 1ère expérimentation, nous avons étudié l'ensemble des couples d'éléments qui peuvent être générés à partir de la configuration initiale. Nous avons choisi la stratégie par défaut et un seuil permettant de sélectionner tous les éléments comme des éléments cibles. Notre objectif est, en effet, d'étudier l'ensemble des couples d'éléments à aligner indépendamment de la sélection permise par les différentes stratégies.

Le fait de disposer de la liste des mappings devant être générés nous a permis d'évaluer la qualité de notre méthode de sélection.

Résultats et analyse Le tableau 6.8 présente les moyennes par pair des nombres d'éléments cibles exploités (éléments cibles faisant partie des couples d'éléments qu'il est pertinent d'aligner), de couples d'éléments pertinents à aligner obtenus et de couples d'éléments valides obtenus sans sélection, et avec notre méthode de sélection. Comme dans l'expérimentation précédente, les moyennes sont obtenues pour l'ensemble des pairs du système.

	sans sélection	avec sélection
# moyen d'éléments cibles exploités par pair	172	165.17
# moyen de couples d'éléments par pair [1]	2928816	4722.54
# moyen de couples valides par pair [2]	72864	447.08
[2]/[1]	2.48	9.4

TAB. 6.8 – Comparatif du nombre de couples d'éléments pertinents à aligner selon la méthode de sélection

On observe, dans le tableau 6.8, que le nombre de couples d'éléments à aligner est très important lorsqu'il n'y a aucune sélection. À l'échelle de 100 pairs, on obtient près de 3 millions de couples pour des ontologies de taille modeste (moins de 200 éléments). Ce nombre augmente de façon exponentielle par rapport à la taille du réseau. Notre méthode de sélection permet de limiter, de façon efficace, cette augmentation et empêche ainsi l'explosion combinatoire lors du passage à l'échelle. Lors de la 1ère expérience, le nombre de couples obtenus après sélection représentait 1.5 à 3.7 % du nombre de couples obtenus sans sélection. Ce ratio est ici de 0.16 % (= 4722.54/2928816). Ceci montre la pertinence de notre méthode de sélection par rapport au problème de passage à l'échelle.

Lorsqu'on s'intéresse à la qualité de la sélection, la première constatation est que de nombreux mappings valides n'ont pas été générés. En moyenne, seuls 447.08 mappings valides ont été générés alors qu'il est possible d'en trouver 72864 dans le réseau.

La principale cause de ce phénomène est le contexte totalement distribué. Dans ce contexte, un pair local possède un accès limité aux connaissances partagées dans le système pair-à-pair. Dans cette expérimentation, nous avons constaté qu'un pair pouvait

avoir accès, en moyenne, à 564.77 éléments appartenant à des ontologies de pairs distants, à l'aide des mécanismes de raisonnement mis en œuvre dans SomeRDFS. Au mieux, il n'est donc possible de générer que 1942,80 couples valides parmi les 72864 existants.

L'autre cause de ce phénomène est que notre approche repose sur la notion de contexte d'interprétation commun. Des éléments sont pertinents à aligner s'ils partagent un généralisant ou un spécialisant commun. Parmi les 1942,80 couples d'éléments valides qui pourraient être générés, certains ne le sont pas car le contexte d'interprétation commun n'est pas représenté dans le système pair-à-pair. Pour d'autres couples, le contexte d'interprétation commun est représenté dans le système pair-à-pair mais les mécanismes de raisonnements mis en œuvre dans SomeRDFS ne suffisent pas pour les exploiter.

Notre méthode permet de sélectionner 0.4% (= 4722.54 /(172×564.77)) des couples générés à partir des connaissances du pair local. Sur ce sous-ensemble, 9.4% (= 447.08/4722.54) sont des couples valides, ce qui correspond à 23% (= 447.08/1942,80) de l'ensemble des couples valides qui peuvent être obtenus sans sélection.

Ceci montre l'intérêt de notre méthode de sélection des éléments à aligner et évalue son importance en distinguant la sélection s'opérant naturellement du fait de l'architecture distribuée de celle mise en œuvre dans la méthode que nous proposons.

6.2.2 Test pour la méthode d'alignement

La technique d'alignement déducto-syntaxique que nous proposons repose sur la combinaison de mécanismes de raisonnement dont la preuve de correction et de complétude a été faite dans la thèse de Philippe Adjiman (Adjiman (2006)) et de techniques d'alignement qui ont été éprouvées. L'originalité de notre proposition tient en l'exploitation particulière des techniques d'alignement qui sont appliquées, non pas entre les éléments de différentes ontologies mais entre les éléments d'une même ontologie. Cette originalité ne modifie pas fondamentalement les techniques qui sont appliquées. La validation de la technique d'alignement syntaxique revient donc à valider la combinaison de deux techniques qui, individuellement ont été validées. Cette validation ne justifie donc pas, à nos yeux, la mise en place d'une évaluation expérimentale.

Les tests ont porté sur la technique inductive. Plus précisément, l'objectif a été d'évaluer la mesure de score proposée (cf. 5.2) pour calculer la confiance associée à un mapping.

Ontologies

Pour ces expérimentations nous avons utilisé 15 ontologies du domaine des publications proposées sur le site de la campagne d'évaluation pour d'alignement d'ontologies OAEI2008[2]. Il s'agit d'ontologies au format OWL dont nous avons extrait la taxonomie. Les caractéristiques de ces ontologies sont données dans le tableau 6.9. Nous avons ensuite créé un pair à partir de chacune de ces taxonomies.

[2]http://oaei.ontologymatching.org/

Nom	Nombre de Classes	Lien
Ekaw	77	http://ekaw.vse.cz
Sofsem	60	http://www.sofsem.cz
Sigkdd	49	http://www.acm.org/sigs/sigkdd/kdd2006
Iasted	140	http://iasted.com/conferences/2005/cancun/ms.htm
Micro	32	www.microarch.org
Confious	57	http://www.confious.com
Pcs	23	http://precisionconference.com
OpenConf	62	http://www.zakongroup.com/technology/openconf.shtml
ConfTool	38	http://www.conftool.net
Crs	14	http://www.conferencereview.com
Cmt	36	http://msrcmt.research.microsoft.com/cmt
Cocus	55	http://cocus.create-net.it/
Paperdyne	47	http://www.paperdyne.com/
Edas	104	http://edas.info/
MyReview	39	http://myreview.intellagence.eu/

TAB. 6.9 – Ontologies

Expérimentation

Description Les tests réalisés ne sont pas des tests de validation de la mesure de score proposée puisque celle-ci est de nature statistique et qu'elle prend donc tout son sens lorsque le réseau est constitué d'un nombre très important de pairs. Un tel cadre expérimental n'étant pas possible à établir, l'objectif de nos tests s'est réduit à une analyse/discussion de la mesure proposée.

Pour cela, nous nous sommes intéressé aux éléments appartenant à l'ontologie d'un pair $\mathcal{P}_1$ pour lesquels il existe des spécialisants appartenant aux ontologies d'autres pairs. En effet, la technique inductive consiste à rechercher les spécialisants d'éléments de l'ontologie du pair local dans l'ontologie de pairs distants. Nous avons recherché les spécialisants de ces éléments dans les ontologies des autres pairs du réseau (cf. tableau 6.10). Lorsqu'un même label désigne des spécialisants dans les ontologies de plusieurs pairs, nous indiquons, entre parenthèses, le nombre de ces ontologies.

Résultats et analyse Les mappings de $\mathcal{P}_1$ permettent d'obtenir des spécialisants appartenant à des ontologies de pairs distants pour 48 éléments parmi les 140 que compte l'ontologie de $\mathcal{P}_1$.

Élément	Spécialisants
Review	Review (5)
State	Country (2)
Activity	Tutorial (3) Submission_event Banquet Lecture activity Conference_Banquet Registration_of_participants_event Conference (2) submission_process Trip Session WelcomeTalk OrganizationalMeeting

Élément	Spécialisants
	Workshop
	NonAcademicEvent
	MealEvent
	BreakEvent
	Excursion
	CoffeeBreak
	FreeTimeBreak
	TalkEvent
	MealBreak
	AcademicEvent
	PaperPresentation
	ConferenceEvent
	Reception
	ConferenceDinner
	ClosingTalk
	SocialEvent
Main_office	Main_office
Item	Review (8)
	CD-ROM
	Conference_proceedings
	Camera_Ready_Copy
	Submitted_contribution
	Paper(5)
	Proceedings
	Slides
	Contribution
	CD
	Document
	ActivePaper
	ReviewForm
	PendingPaper
	RejectedPaper
	AcceptedPaper
	PublishedPaper
	RatedPapers
	WithdrawnPaper
Non_speaker	Passive_conference_participant
Transparency	Slides
Person	External_Reviewer
	Paper_Author
	Secondary_Reviewer
	Author (3)
	Reviewer (4)
	Passive_conference_participant
	Presenter
	SessionChair
Tutorial	Tutorial (3)
Deadline_for_notification_of_acceptance	date_of_notification
Delegate	Session_Chair (2)
	Listener
	Author (8)
	Secondary_Reviewer
	Passive_conference_participant
	Reviewer (5)

Élément	Spécialisants
	External_Reviewer (2) Paper_Author Presenter
Final_manuscript	Camera_Ready_Copy Camera_Ready_Paper
Registration_fee	Registration_fee
Submission	Paper (6) Rejected_paper (3) Camera_Ready_Copy PaperSubmitted_Paper Contribution PendingPaper ActivePaper PublishedPaper WithdrawnPaper RatedPapers AcceptedPaper
Document	Paper (7) Review (7) Submitted_Paper Contribution Document Slides Camera_Ready_Paper Rejected_paper (2) Camera_Ready_Copy Submitted_contribution ActivePaper ReviewForm WithdrawnPaper RatedPapers PublishedPaper PendingPaper AcceptedPaper
Book_proceeding	Proceedings
Publication	CD Electronic_proceeding CD-ROM Conference_proceedings Proceedings
Registration	Registration_of_participants_event
Place	Main_office Hotel Place (2) Conference_hall DiningPlace ConferenceVenuePlace AccommodationPlace MeetingRoomPlace
Receiving_manuscript	submission_process Submission_event
Fee	Conference_fees Fee (2) Registration_fee

Élément	Spécialisants
Full_day_tour	Trip Excursion
Plenary_lecture	Conference (2) Lecture
Activity_after_conference	Trip Excursion
Dinner_banquet	Conference_Banquet Banquet ConferenceDinner
Money	Fee (2) Conference_fees Registration_fee
Conference_activity	Conference (2) Lecture Tutorial (3) Session Conference_Banquet Banquet Registration_of_participants_event Reception CoffeeBreak WelcomeTalk ConferenceDinner
Conference_hall	Conference_hall
Session	Session
Lecture	Conference (2) Session Tutorial (3) Lecture
Listener	Listener
Activity_before_conference	Submission_event submission_process
Camera_ready_manuscript_deadline	end_date_of_camera-ready_submission
Speaker	Reviewer (4) External_Reviewer (2) Author (6) Regular_author Paper_Author Presenter
Cd_proceening	CD CD-ROM
Time	date_of_notification end_date_of_camera-ready_submission Important_dates
Reviewer	Reviewer (5) External_Reviewer (2) Secondary_Reviewer
Social_program	Conference_Banquet Banquet ConferenceDinner
Initial_manuscipt	Rejected_paper (2) Submitted_Paper
City	City

Élément	Spécialisants
Building	Hotel
Session_chair	Session_Chair (2)
Author	Author (4) Presenter
Deadline	end_date_of_camera-ready_submission date_of_notification
Form	ReviewForm
Coffee_Break	CoffeBreak
Welcome_adress	WelcomeTalk
Coctail_reception	Reception

TAB. 6.10: Éléments de l'ontologie du pair local et leurs spécialisants appartenant à des ontologies de pairs distants

Nous remarquons, tout d'abord, que le fait d'exploiter les mécanismes de raisonnements nous permet de trouver des liens entre des éléments désignés par des labels qui peuvent être très différents. Par exemple, le terme *Author* désigne souvent des spécialisants de *Speaker*. Ces deux termes ne sont pas syntaxiquement proches et l'application de techniques syntaxiques ne permet pas de savoir qu'il existe un lien de spécialisation entre les éléments désignés respectivement par ces deux termes.

Nous remarquons, par ailleurs que les labels trouvés dans plusieurs ontologies sont souvent non ambigus et désignent plutôt des concepts généraux. Ces concepts appartiennent à la plupart des ontologies d'un même domaine et ils font l'objet d'un consensus quand aux termes qui les désignent. On retrouve *Author*, *Review*, *Country*, *Tutorial*, *Paper*.

Beaucoup de labels n'apparaissent que dans une seule ontologie. Les raisons sont multiples.

Les labels désignant des spécialisants n'appartenant qu'à une ontologie peuvent être plus particuliers dans la mesure où les concepts qu'ils désignent peuvent être spécifiques d'un point de vue donné sur le domaine. Par exemple, dans l'ontologie MyReview, le concept *Paper* est considéré du point de vue du processus de soumission. On trouve *AcceptedPaper* et *RejectedPaper* comme spécialisants de *Paper*. Dans d'autres ontologies, comme cocus, *Paper* est considéré du point de vue du type de document. On trouve alors *InvitedPaper*, *ShortPaper*, *Abstract* et *FullPaper* comme spécialisants de *Paper*. Dans ce cas, pour que la mesure de score soit pertinente, il est nécessaire que les différents points de vue soient partagés par plusieurs ontologies. De même qu'il existe un consensus pour les termes désignant des concepts généraux du domaine, il doit exister un consensus pour les termes désignant des concepts plus spécifiques dans une communauté de pairs qui partagent un même point de vue sur une partie du domaine. À l'échelle d'un réseau composé de très nombreux pairs, la mesure proposée permet alors, non seulement de valider des mappings générés pour des concepts généraux mais également d'étendre cette validation aux concepts plus spécifiques en profitant de passerelles établies entre les différentes communautés de pairs.

Le fait qu'un label n'apparaisse que dans une seule ontologie peut aussi signifier que cette ontologie couvre un (sous-) domaine plus vaste que les autres ontologies. Par exemple, l'ontologie cocus décrit 4 spécialisants de *Paper* alors que dans d'autres ontologies, comme l'ontologie cmt, *Abstract* et *FullPaper* sont les seuls spécialisants de *Paper*. Il peut également s'agir de la représentation d'un niveau de détail particulier. Par exemple, dans l'ontologie Ekaw, *AcceptedPaper* et *RejectedPaper* sont séparés de *Paper* par 3 concepts intermédiaires. Au contraire, dans MyReview, ce sont des spécialisants directs de *Paper*. Dans ces deux cas, si le nombre de pairs dans le réseau augmente, les pairs décrivant le domaine avec un même niveau de détail et un même niveau de couverture seront également plus nombreux. Notre mesure prendra, là encore, tout son sens.

Conclusion

Dans ce chapitre, nous avons présenté les réalisations logicielles accomplies et les expérimentations que ce travail d'implémentation a rendu possible. En plus du développement logiciel, nous avons également construit plusieurs réseaux de pairs afin de pouvoir observer le comportement des différents algorithmes proposés. Ceci a nécessité une étude indépendante et un travail conséquent portant sur le développement de réseaux et le déploiement d'ontologies et de mappings au sein des différents pairs.

Au terme de cette évaluation, nous avons pu valider la méthode de sélection des couples d'éléments qu'il est pertinent d'aligner. Nous avons montré que cette sélection était nécessaire lors du passage à l'échelle d'un réseau comptant un nombre conséquent de pairs. Nous avons également montré l'intérêt de la méthode de sélection que nous proposons et des différentes stratégies mises en œuvre.

Enfin, nous avons analysé la mesure de score utilisée dans une des techniques d'alignement en nous appuyant sur un exemple de réseau pair-à-pair développé dans un cadre expérimental. Cette analyse a permis de montrer tout l'intérêt de cette mesure lorsqu'elle est appliquée sur un réseau de grande taille composé de pairs dont les ontologies couvrant un même domaine selon un même point de vue sont nécessairement multiples.

Conclusion et perspectives

Les travaux menés dans cette thèse s'inscrivent dans le contexte de la découverte de mappings entre ontologies dans un système pair à pair sémantique. L'objectif recherché est l'augmentation du nombre de mappings entre les ontologies des différents pairs du système afin d'améliorer les réponses données par ce système aux requêtes qui lui sont posées. Dans ces systèmes, l'absence de contrôle centralisé et le nombre important d'ontologies à aligner constituent des singularités qui nécessitent une approche adaptée.

Nous présentons, dans un premier temps, une synthèse de nos contributions, et, dans un deuxième temps, les perspectives envisagées pour la suite de nos travaux.

Contributions de la thèse

La proposition d'une méthode de sélection des mappings inférables

Nous avons d'abord défini la notion de *raccourcis de mappings* qui désigne des mappings qui ont la particularité d'être déductibles à partir du raisonnement mis en œuvre dans SomeRDFS. Une propriété inhérente à ces mappings particuliers est qu'ils introduisent de la redondance dans le système. Nous avons montré l'intérêt et l'inconvénient de la présence de redondance dans SomeRDFS.

L'intérêt de la redondance introduite par les *raccourcis de mappings* réside dans l'enrichissement du vocabulaire du pair local et la limitation des pertes occasionnées par la déconnexion d'autres pairs du réseau. Son inconvénient vient de la surcharge du système qu'elle provoque lors du processus de réponse aux requêtes. Nous en avons conclu qu'une sélection des *raccourcis de mappings* est nécessaire.

Nous avons proposé de faire reposer cette sélection sur une analyse des requêtes posées par les utilisateurs. Cette analyse est permise par la décomposition du processus de réponse aux requêtes en une étape de réécriture et une étape d'évaluation. Cette décomposition permet aux utilisateurs de demander une évaluation sélective des réécritures obtenues à une requête. Nous avons montré comment cette évaluation sélective des réécritures permet la sélection de *raccourcis de mappings*. Nous avons également proposé différents critères permettant à l'administrateur de sélectionner de façon plus précise les *raccourcis de mappings*.

La proposition d'une méthode de découverte de nouveaux mappings

Nous avons d'abord mis en évidence l'impossibilité de comparer tous les éléments composant les ontologies utilisées dans un système pair-à-pair et la nécessité de sélectionner les couples d'éléments qu'il est *pertinent* d'aligner. Nous avons proposé une sélection basée sur l'analyse des réponses aux requêtes des utilisateurs et sur la pose de requêtes au système. Nous avons ensuite proposé un ensemble de techniques spécifiques à notre contexte de travail permettant de générer des mappings à partir de cette sélection d'éléments à aligner.

L'analyse des réponses aux requêtes des utilisateurs permet d'identifier les éléments pour lesquels un manque de mappings provoque des réponses peu nombreuses. La recherche de nouveaux mappings pour ces éléments, appelés éléments cibles, est prioritaire.

Nous avons ensuite défini la notion de contexte d'interprétation commun afin de construire, à partir des éléments cibles, un ensemble de couples d'éléments qu'il est *pertinent* d'aligner. Rappelons que la notion de pertinence fait ici référence au fait qu'il existe une proximité structurelle entre les deux éléments d'un couple et non à la probabilité de trouver, à terme, un mapping entre ces deux éléments. Nous avons proposé d'exploiter les mécanismes de raisonnement mis en œuvre dans SomeRDFS à partir des éléments constituant le contexte d'interprétation d'un élément cible afin d'identifier les éléments distants qui partagent en partie ce contexte d'interprétation.

Nous avons également proposé différents critères permettant à l'administrateur de sélectionner de façon plus précise les couples d'éléments qu'il est pertinent d'aligner.

Enfin, nous avons proposé plusieurs techniques d'alignement exploitant, de façon conjointe, les mécanismes de raisonnement mis en œuvre dans SomeRDFS et des techniques d'alignement de la littérature (notamment les techniques syntaxiques et les techniques exploitant une ressource externe). Notre méthode repose sur une application séquentielle des deux techniques d'alignement que nous proposons.

La première technique consiste à exploiter les traces des requêtes posées et de les compléter, au besoin, par des techniques d'alignement syntaxiques en complément des mécanismes de raisonnement afin de générer des mappings. Les mappings générés par exploitation des seuls mécanismes de raisonnement sont sûrs. Il n'est pas nécessaire de les valider. Les mappings générés à partir de techniques d'alignement syntaxiques doivent être validés par l'administrateur.

La seconde technique proposée repose sur l'exploitation du système pair-à-pair sémantique comme une ressource externe. Les particularités de cette ressource nous ont permis de proposer une approche statistique basée sur le label des éléments. Cette approche consiste à identifier les spécialisants d'un élément grâce aux mécanismes de raisonnement puis à évaluer, à partir de cet ensemble de spécialisants, la probabilité qu'un label donné désigne un spécialisant de cet élément dans une nouvelle ontologie. Cette approche, appliquée aux couples d'éléments à aligner, permet de calculer la mesure confiance associée aux mappings générés.

La proposition d'un environnement de validation de mappings

Les méthodes de découverte de mappings que nous proposons permettent de générer un ensemble de mappings dont certains doivent être validés par l'administrateur. Cependant, cette validation est rendue difficile par le nombre important de mappings générés et la connaissance très limitée des ontologies des autres pairs. Nous avons donc proposé un environnement de validation des mappings générés permettant de mettre à la disposition de l'administrateur un ensemble d'outils pouvant l'aider dans cette tâche.

La première fonction de cet environnement est la visualisation des informations sur les éléments à aligner. Ces informations, recueillies au cours du processus de découverte de mappings, permettent à l'administrateur de valider les mappings générés. Notre originalité est la construction d'une *vue* sur l'ontologie d'un pair distant à partir de ces informations. Une *vue* permet de présenter sous la forme d'une ontologie incomplète des éléments qui, sans cela, sont vus comme isolés par l'administrateur du pair local. La mise à disposition des *vues* permet une meilleure compréhension de ces éléments par l'administrateur et facilite, de ce fait, la validation des mappings générés.

La seconde fonction de cet environnement est de permettre l'acquisition de nouvelles informations. Cette acquisition se fait au travers des requêtes posées par l'administrateur et au travers de la validation des mappings.

Les requêtes posées par l'administrateur lors du processus de validation afin de mieux comprendre les éléments apprtenant à des ontologies de pairs distants sont exploitées par la méthode de découverte de nouveaux mappings. L'administrateur a besoin de mieux comprendre ces éléments car il ne font pas partie du vocabulaire du pair local. Il s'agit donc généralement de requêtes qui n'ont encore jamais été posées.

Nous avons également proposé une exploitation de la validation des mappings générés afin de valider/rejeter automatiquement d'autres mappings générés. Pour cela, nous avons défini un ensemble de schémas d'interaction entre les mappings. Nous nous sommes ensuite appuyé sur ces schémas pour proposer des actions à appliquer automatiquement.

Une implémentation

Les méthodes proposées ont donné lieu à une implémentation au sein d'un prototype, SpyWhere. Il s'agit d'une implémentation partielle visant à permettre l'évaluation de notre approche. Certaines fonctionnalités, comme le filtrage des *raccourcis de mappings*, n'ont pas été implémentées car nous ne pouvions pas les évaluer. D'autres fonctionnalités n'ont pas pu être implémentées faute de temps. Ce prototype repose sur la plateforme SomeRDFS que nous avons dû implémenter en partie.

SpyWhere permet actuellement de :
- sélectionner les *raccourcis de mappings* à ajouter
- identifier les éléments cibles selon les différentes stratégies proposées
- identifier les couples d'éléments qu'il est pertinent d'aligner selon les différentes stratégies proposées
- générer des mappings selon une technique d'alignement choisie.

L'ensemble de ces fonctionnalités est accessible au travers d'une interface graphique intégrant également SomeRDFS.

Un ensemble d'expérimentations

Grâce à notre prototype, nous avons pu réaliser différentes expérimentations. Ces expérimentations nous ont permis de valider la méthode de sélection des couples d'éléments qu'il est pertinent d'aligner et de discuter de la mesure de score sur laquelle repose une des techniques d'alignement que nous avons proposée.

Nous n'avons pas pu réaliser une réelle validation complète de la méthode que nous avons proposée. Il serait intéressant de mettre en place un protocole expérimental reposant sur un réseau de taille conséquente, composé de pairs disposant chacun sa propre ontologie.

Perspectives

Les perspectives de cette thèse concernent, dans un premier temps, l'amélioration des méthodes proposées à partir de la définition d'un nouveau type de requêtes basées sur l'exploitation des mécanismes de raisonnement mis en œuvre dans SomeWhere mais non proposé dans SomeRDFS.

Nous envisageons, ensuite, l'application de nos méthodes dans un système utilisant un formalisme plus riche.

Enfin, nous avons considéré l'aspect dynamique d'un réseau pair-à-pair dans lequel les pairs sont libres de se connecter et de se déconnecter à tout moment. Nous avons proposé une solution basée sur la sélection des *raccourcis de mappings* pour limiter les conséquences de la déconnexion de pairs. Une autre perspective est l'amélioration de cette solution par une optimisation de l'algorithme DECA, sur lequel repose SomeWhere, au niveau du traitement des mappings.

Un autre aspect qui semble intéressant est le fait que chaque pair est responsable de ses connaissances et qu'il peut constamment les faire évoluer. Cette évolution implique une remise en question des mappings existants et la recherche de nouveaux mappings.

Exploitation de SomeWhere

SomeRDFS n'exploite qu'une partie des mécanismes de raisonnement mis en œuvre dans SomeWhere. Les modifications apportées aux spécifications de SomeRDFS consistent à fournir un accès aux différentes étapes du processus de réponse aux requêtes. Cependant, nous n'avons pas modifié l'algorithme $DECA^{RDFS}$ qui exploite SomeWhere. Une autre exploitation de SomeWhere autorise la définition d'un nouveau type de requête permettant de connaître tous les généralisants d'un élément. Ce type de requête n'est pas utile à SomeRDFS, c'est pourquoi il n'existe pas. Dans SpyWhere, la sélection des couples d'éléments à aligner exploite les contextes d'interprétation communs et les mécanismes de raisonnement. Les mécanismes de raisonnement sont exploités pour obtenir soit l'ensemble des spécialisants soit l'ensemble des généralisants d'un élément. Lorsqu'on

a besoin de connaître les généralisants d'un élément, on exploite actuellement les connaissances du pair local. Les éléments distants appartenant aux couples d'éléments pertinents à aligner obtenus par exploitation d'un spécialisant commun sont donc nécessairement des éléments appartenant aux connaissances du pair local. Disposer de requêtes de généralisation permettrait d'obtenir des éléments n'appartenant pas aux connaissances du pair local, par exploitation des spécialisants communs.

De plus, les contextes d'interprétation communs exploités sont actuellement des éléments appartenant aux connaissances du pair local. Ce nouveau type de requêtes permettrait d'exploiter tous les contextes d'interprétation communs.

Lors de la phase d'alignement, la détection des mappings inférables, serait simplifiée si on disposait de ce type de requêtes. Tous les mappings inférables seraient détectés grâce à 2 × n requêtes (où n est le nombre d'éléments dans l'ontologie du pair local). Actuellement, cette détection nécessite n + m requêtes (où n est le nombre d'éléments dans l'ontologie du pair local et m est le nombre d'éléments appartenant à des ontologies de pairs distants).

Extension à $DL-lite_R$

En parallèle à cette thèse, une nouvelle plateforme, SomeDL-lite, basée sur SomeWhere a été conçue. Cette plateforme est semblable à SomeRDFS à la différence qu'elle exploite le formalisme $DL-lite_R$ et permet une description plus précise des connaissances. Nos différentes méthodes peuvent être appliquées dans cette nouvelle plateforme. Une perspective serait alors de les adapter pour qu'elles exploitent toute la richesse de ce nouveau formalisme.

Par exemple, dans SomeDL-lite, il est possible de représenter le fait que deux propriétés sont des propriétés inverses. Deux propriétés P_1 et P_2 sont dites inverses si le domaine de P_1 correspond à la portée de P_2 et si la portée de P_1 correspond au domaine de P_1. La méthode de sélection des couples d'éléments à aligner, peut permettre de détecter ce type de liens entre deux propriétés si on adapte la prise en compte du formalisme que nous avons proposée par rapport à RDF(S).

La présence de la négation dans SomeDL-lite permet de représenter la disjonction entre éléments. Ce type de connaissances semble intéressant pour la sélection des couples d'éléments à aligner.

Extension des mappings dans SomeRDFS

Nous avons détaillé l'importance de la sélection des *raccourcis de mappings* pour limiter la surcharge du système par le processus de réponse aux requêtes. Cependant, cette sélection est nécessaire parce que tous les mappings sont exploités par les mécanismes de réponse aux requêtes. Une perspective intéressante serait l'ajout de la distinction entre mappings et *raccourcis de mappings*. Cette distinction permettrait, au niveau de l'algorithme DECA, d'exploiter les raccourcis de mappings uniquement lorsque l'exploitation des mappings seuls ne permet pas d'obtenir toutes les réponses attendues. Cette modification permettrait de proposer une sélection dynamique des *raccourcis de mappings* au moment de l'obtention des réponses aux requêtes et non une sélection statique au moment de l'ajout des *raccourcis de mappings*, comme nous le proposons dans le chapitre 3.

Le problème qui se pose alors est, d'abord, de détecter que toutes les réponses attendues n'ont pas été obtenues puis d'identifier les mappings défaillants et les *raccourcis de mappings* à exploiter en remplacement des mappings défaillants tout en limitant la surcharge provoquée dans le système.

Extension à l'évolution de mappings et d'ontologies

Dans cette thèse, nous avons montré la nécessité d'une sélection des éléments à aligner. Nous avons proposé d'exploiter les requêtes des utilisateurs comme filtre de sélection pour les éléments de l'ontologie du pair local. Les éléments cibles, pour lesquels la recherche de mappings est *intéressante*, sont alors des *points d'intérêt*. La définition des éléments cibles pourrait reposer sur un autre type d'événement : la modification de l'ontologie du pair local par l'administrateur. Lorsqu'un élément est ajouté, il n'existe aucun mapping mettant en jeu cet élément. Selon sa position d'ajout dans l'ontologie, il est possible que cet élément possède peu de spécialisants dans les ontologies de pairs distants. Il est alors intéressant d'étudier l'applicabilité de nos méthodes en redéfinissant la notion de *point d'intérêt* des éléments cibles dans ce cas.

Par ailleurs, nous avons exploité le système pair-à-pair comme une ressource externe afin de proposer une mesure de confiance pour les mappings. Une exploitation similaire pourrait être intéressante dans le cadre de la construction d'ontologie. Le système permet à l'administrateur d'obtenir une liste de termes désignant des éléments partageant un lien de subsomption avec un élément choisi. S'il souhaite décrire de façon plus précise un élément, l'administrateur peut s'inspirer des spécialisants de cet élément dans les ontologies des pairs distants.

Enfin, lorsque les connaissances d'un pair distant évoluent, il est possible que les mappings partagés entre le pair local et ce pair distant ne soient plus corrects. L'administrateur du pair local est alors amené à supprimer les mappings devenus incorrects. Cependant, les mappings ainsi supprimés peuvent faire partie de liens, restés corrects, avec d'autres pairs. Il serait intéressant d'étudier comment identifier les mappings à ajouter afin de maintenir ces liens après la suppression des mappings devenus incorrects.

Table des figures

Liste des tableaux

Annexes

Annexe A

Algorithmes de prise en compte du formalisme de RDF(S) pour la sélection de couples d'éléments à aligner

Dans le chapitre 4, nous avons décrit la façon dont nous proposons de sélectionner les couples d'éléments qu'il est pertinent à aligner. Plus particulièrement, en section 4.2.2, nous avons montré l'intérêt de la prise en compte du formalisme RDF(S) et notamment du type (classe ou propiéré) des éléments considérés. Nous décrivons, dans cette annexe, les algorithmes 11 et 12 appelés dans l'algorithme 7 du chapitre 4 afin de permettre la prise en compte de ces éléments.

Dans l'algorithme 11, les cas impossibles ne sont pas traités : ils ne peuvent pas se présenter du fait du formalisme de RDF(S). On suppose donc les ontologies et les mappings bien formés. Cette hypothèse n'est pas très contraignante et permet de supprimer de nombreux tests, ce qui, à l'échelle de milliers (voire plus) d'éléments à vérifier n'est pas négligeable. L'algorithme 11 permet de distinguer les cas où l'élément distant est ajouté à l'ensemble $MC(\mathcal{P}_{loc}{:}E_j)$ (si la conclusion de l'étude est OK sans condition ou OK avec condition et que la condition est vérifiée) des cas où l'élément distant n'est pas ajouté à l'ensemble $MC(\mathcal{P}_{loc}{:}E_j)$ (si la conclusion de l'étude est OK avec condition et que la condition n'est pas vérifiée).

Algorithme 11 : isReallyCommon($\mathcal{P}_{loc}$:E_j, $\mathcal{P}_{dist}$:E_k, $\mathcal{P}_i$:E_m)

Entrées : Un élément cible E_j appartenant à l'ontologie du pair local $\mathcal{P}_{loc}$, un élément E_k appartenant à l'ontologie d'un pair distant $\mathcal{P}_{dist}$ et $\mathcal{P}_i$:E_m le contexte d'interprétation commun à E_j et E_k
Sorties : renvoi TRUE si $\mathcal{P}_{dist}$:E_k doit être ajouté à l'ensemble MC($\mathcal{P}_{loc}$:E_j). renvoi FALSE si $\mathcal{P}_{dist}$:E_k ne doit pas être ajouté à l'ensemble MC($\mathcal{P}_{loc}$:E_j).

```
1  suivant les types respectifs de P_loc:E_j, P_i:E_m et P_dist:E_k faire
2      cas où P_loc:E_j est une classe, P_i:E_m est une classe, P_dist:E_k est une classe
3          retourner True
       fin
4      cas où P_loc:E_j est une propriété, P_i:E_m est une propriété, P_dist:E_k est une propriété
5          retourner True
       fin
6      cas où P_loc:E_j est une classe, P_i:E_m est une propriété, P_dist:E_k est une propriété
7          retourner True
       fin
8      cas où P_loc:E_j est une propriété, P_i:E_m est une classe, P_dist:E_k est une classe
9          retourner True
       fin
10     cas où P_loc:E_j est une propriété, P_i:E_m est une propriété, P_dist:E_k est une classe
11         retourner True
       fin
12     cas où P_loc:E_j est une classe, P_i:E_m est une classe, P_dist:E_k est une propriété
13         retourner True
       fin
14     cas où P_loc:E_j est une classe, P_i:E_m est une propriété, P_dist:E_k est une classe
15         si both P_loc:E_j and P_dist:E_k type le domaine ou la portée de P_i:E_m alors
16             retourner True
           fin
17         sinon
18             retourner False
           fin
       fin
19     cas où P_loc:E_j est une propriété, P_i:E_m est une classe, P_dist:E_k est une propriété
20         si P_i:E_m type à la fois le domaine de P_loc:E_j et de P_dist:E_k alors
21             si ∃P_n:E_r un élément plus general que les portées respectives de P_loc:E_j et de P_dist:E_k alors
22                 retourner True
23             sinon
24                 retourner False
               fin
           fin
25         sinon si P_i:E_m type à la fois la portée de P_loc:E_j et de P_dist:E_k alors
26             si ∃P_n:E_r un élément plus general que les domaines respectifs de P_loc:E_j et de P_dist:E_k alors
27                 retourner True
28             sinon
29                 retourner False
               fin
           fin
30         sinon
31             retourner False
           fin
       fin
   fin
```

On souhaite distinguer les éléments de MC($\mathcal{P}_{loc}$:E_j) appartenant aux ontologies de pairs distants selon qu'ils sont du même type (classe ou propriété) que $\mathcal{P}_{loc}$:E_j ou non. S'ils ne sont pas du même type, le seul mapping qui peut être représenté est un typage de la propriété par la classe. On souhaite alors, également, distinguer les typages de domaine et les typages de portée. L'ensemble MC($\mathcal{P}_{loc}$:E_j) est alors séparé en trois ensembles : les éléments du même type que $\mathcal{P}_{loc}$:E_j sont placés dans $MC'(\mathcal{P}_{loc}$:$E_j)$, les autres éléments sont placés dans l'ensemble $MC_{DOM}(\mathcal{P}_{loc}$:$E_j)$ si le typage concerne le domaine ou dans l'ensemble $MC_{RAN}(\mathcal{P}_{loc}$:$E_j)$ si le typage concerne la portée. L'algorithme 12 effectue cette répartition. Nous avons présenté les algorithmes 11 et 12 séparément afin de dissocier les traitements qu'ils effectuent respectivement. Cependant, dans l'implémentation, ces deux

algorithmes sont fusionnés.

Algorithme 12 : AddToMC ($\mathcal{P}_{dist}$:E_k, $\mathcal{P}_{loc}$:E_j, $\mathcal{P}_k$:E_m)

Entrées : Un élément cible E_j appartenant à l'ontologie du pair local $\mathcal{P}_{loc}$, un élément E_k appartenant à l'ontologie d'un pair distant $\mathcal{P}_{dist}$ et $\mathcal{P}_i$:E_m le contexte d'interprétation commun à E_j et E_k
Sorties : ajoute $\mathcal{P}_{dist}$:E_k au sous-ensemble de $MC(\mathcal{P}_{loc}$ souhaité :$MC'(\mathcal{P}_{loc}$, $MC_{DOM}(\mathcal{P}_{loc}$ ou $MC_{RAN}(\mathcal{P}_{loc}$
suivant *les types respectifs de* $\mathcal{P}_{loc}$:E_j, $\mathcal{P}_i$:E_m *et* $\mathcal{P}_{dist}$:E_k **faire**
  cas où $\mathcal{P}_{loc}$:E_j *est une classe,* $\mathcal{P}_i$:E_m *est une classe,* $\mathcal{P}_{dist}$:E_k *est une classe*
    AddTo ($\mathcal{P}_{dist}$:E_k, MC'($\mathcal{P}_{loc}$:E_j), $\mathcal{P}_{loc}$:E_j)
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une propriété,* $\mathcal{P}_i$:E_m *est une propriété,* $\mathcal{P}_{dist}$:E_k *est une propriété*
    AddTo ($\mathcal{P}_{dist}$:E_k, MC'($\mathcal{P}_{loc}$:E_j), $\mathcal{P}_{loc}$:E_j)
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une classe,* $\mathcal{P}_i$:E_m *est une propriété,* $\mathcal{P}_{dist}$:E_k *est une propriété*
    si $\mathcal{P}_{loc}$:E_j *type le domaine de* $\mathcal{P}_i$:E_m **alors**
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{DOM}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    sinon
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{RAN}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    fin
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une propriété,* $\mathcal{P}_i$:E_m *est une classe,* $\mathcal{P}_{dist}$:E_k *est une classe*
    si $\mathcal{P}_i$:E_m *type le domaine de* $\mathcal{P}_{loc}$:E_j **alors**
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{DOM}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    sinon
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{RAN}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    fin
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une classe,* $\mathcal{P}_i$:E_m *est une propriété,* $\mathcal{P}_{dist}$:E_k *est une classe*
    AddTo ($\mathcal{P}_{dist}$:E_k MC'($\mathcal{P}_{loc}$:E_j), $\mathcal{P}_{loc}$:E_j)
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une propriété,* $\mathcal{P}_i$:E_m *est une classe,* $\mathcal{P}_{dist}$:E_k *est une propriété*
    AddTo ($\mathcal{P}_{dist}$:E_k, MC'($\mathcal{P}_{loc}$:E_j), $\mathcal{P}_{loc}$:E_j)
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une propriété,* $\mathcal{P}_i$:E_m *est une propriété,* $\mathcal{P}_{dist}$:E_k *est une classe*
    si $\mathcal{P}_{dist}$:E_k *type le domaine de* $\mathcal{P}_i$:E_m **alors**
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{DOM}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    sinon
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{RAN}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    fin
  fin
  cas où $\mathcal{P}_{loc}$:E_j *est une classe,* $\mathcal{P}_i$:E_m *est une classe,* $\mathcal{P}_{dist}$:E_k *est une propriété*
    si $\mathcal{P}_i$:E_m *type le domaine de* $\mathcal{P}_{dist}$:E_k **alors**
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{DOM}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    sinon
      AddTo ($\mathcal{P}_{dist}$:E_k, $MC_{RAN}(\mathcal{P}_{loc}$:$E_j$), $\mathcal{P}_{loc}$:E_j)
    fin
  fin
fin

Pour certains des éléments de $MC(\mathcal{P}_{loc}$:$E_j)$, le lien sémantique avec E_j est déjà représenté par un mapping ou peut être inféré à partir des connaissances du pair local. Ces éléments doivent être identifiés et supprimés de $MC(\mathcal{P}_{loc}$:$E_j)$ car ils ne permettront pas de générer de nouveaux mappings. Dans l'algorithme 13, on vérifie donc, pour chaque élément de $MC(\mathcal{P}_{loc}$:$E_j)$ s'il n'est pas déjà connu comme spécialisant ou généralisant de E_j. L'existence d'un mapping explicite est vérifiée ligne 2 et celle d'un mapping inférable est vérifiée dans le bloc compris entre les lignes 4 et 7.

Algorithme 13 : AddTo ($\mathcal{P}_{dist}$:E_k, S, $\mathcal{P}_{loc}$:E_j)

Entrées : Un élément cible E_j appartenant à l'ontologie du pair local $\mathcal{P}_{loc}$, un élément E_k appartenant à l'ontologie d'un pair distant $\mathcal{P}_{dist}$ et un ensemble S

Sorties :

```
1 si P_loc:E_j → P_k:E_m or P_k:E_m → P_loc:E_j exists alors
2 |  ne rien faire
  fin
3 sinon
4 |  si isMoreSpecific(P_loc:E_j, P_dist:E_k) alors
5 |  |  ne rien faire
  |  fin
6 |  sinon si isMoreSpecific(P_k:E_m, P_loc:E_j) alors
7 |  |  ne rien faire
  |  fin
8 |  sinon
9 |  |  ajouter P_k:E_m à S
  |  fin
  fin
```

Si un mapping existe déjà (cf. ligne 2), ou s'il peut être inféré à partir des connaissances du pair local, alors l'élément $\mathcal{P}_{dist}$:E_k est ignoré et le couple ($\mathcal{P}_{loc}$:E_j, $\mathcal{P}_{dist}$:E_k) ne sera pas étudié. Sinon, l'élément $\mathcal{P}_{dist}$:E_k est ajouté à l'ensemble S. Dans la pratique, S est un des ensembles MC', MC_DOM ou MC_RAN associés à $\mathcal{P}_i$:E_j. L'algorithme14, est appelé aux lignes 4 et 6 vérifie si un mapping existe ou peut être inféré à partir des connaissances du pair local.

Algorithme 14 : isMoreSpecific($\mathcal{P}_i$:E_j, $\mathcal{P}_k$:E_m)

Entrées : Deux éléments : E_j appartenant à l'ontologie du pair $\mathcal{P}_i$ et E_m appartenant à l'ontologie du pair $\mathcal{P}_k$

Sorties : renvoie TRUE si $\mathcal{P}_i$:$E_j \Rightarrow \mathcal{P}_k$:$E_m$ peut être inféré à partir des connaissances du pair local. renvoie FALSE sinon.

```
1 si P_i:E_j ⇒ P_k:E_m est une connaissance du pair local alors
2 |  retourner TRUE
3 sinon si ∃P_loc:E_n tel que P_loc:E_n ⇒ P_k:E_m est une connaissance du pair local et
  isMoreSpecific(P_i:E_j, P_loc:E_n) alors
4 |  retourner TRUE
5 sinon
6 |  retourner FALSE
  fin
```

L'algorithme 14 comporte une boucle récursive (cf. ligne 3 et deux cas d'arrêt (cf. lignes 1 et 6). Cet algorithme permet de vérifier l'existence, au choix, d'un lien de spécialisation ou de généralisation. Le lien recherché est la généralisation du premier paramètre par le second sans que soit précisé quel est l'élément local et quel est l'élément distant. En appliquant cet algorithme, on s'assure de ne pas ajouter de mappings inutiles : en

effet, les mappings écartés par cet algorithme sont inférables localement. Ajouter un tel mapping ne permet pas d'améliorer les réponses aux requêtes. De plus, l'inférabilité étant locale, ajouter un tel mapping ne permet pas non plus d'augmenter la précision des requêtes qui peuvent être posées. Par contre, ajouter un tel mapping risque d'introduire de la confusion dans la compréhension de l'élément distant par les utilisateurs.

Annexe B

Interface graphique

L'architecture de SomeWhere est de type client/serveur. On distingue donc le logiciel serveur du logiciel client. SomeRDFS utilise, en plus de ces deux programmes, SpyWhere et un gestionnaire de profils. Chacun de ces programmes possède sa propre interface mais nous avons développé une interface permettant d'interagir de façon unifiée avec l'ensemble de ces programmes. Nous présentons cette interface graphique dans cette annexe en décrivant successivement les interactions autorisées avec chacun des programmes accessibles.

B.1 Interactions avec SomeRDFS

Un pair a besoin de pouvoir communiquer avec les autres pairs du PDMS. Pour cela il doit être capable, d'une part, d'être à l'écoute et de répondre à des requêtes, d'autre part, de poser des requêtes. Le premier rôle est assuré par la partie serveur, le second par la partie client.

B.1.1 La partie serveur de SomeRDFS

L'utilisation de la partie serveur d'un pair nécessite des compétences techniques : choix de la machine hôte, choix du port, choix des profils. C'est l'administrateur du pair qui interagit avec cette partie de someRDFS. L'administrateur doit pouvoir choisir le nom de la machine hôte (par défaut localhost) et le numéro de port (par défaut 50000) puis lancer le serveur. Il peut ensuite vouloir ajouter des profils précisant, chacun, l'identifiant d'un pair localisé sur cette machine hôte et contenant l'ontologie et les mappings de ce pair. Une fenêtre de sélection de fichier (cf. figure B.1) apparaît alors pour choisir le profil à ajouter. Le chemin du fichier correspondant à chaque profil ajouté est affiché. L'administrateur peut aussi vouloir supprimer un profil ou le mettre à jour. Par un double clic sur un profil listé, il lui sera possible d'accéder au fichier correspondant.

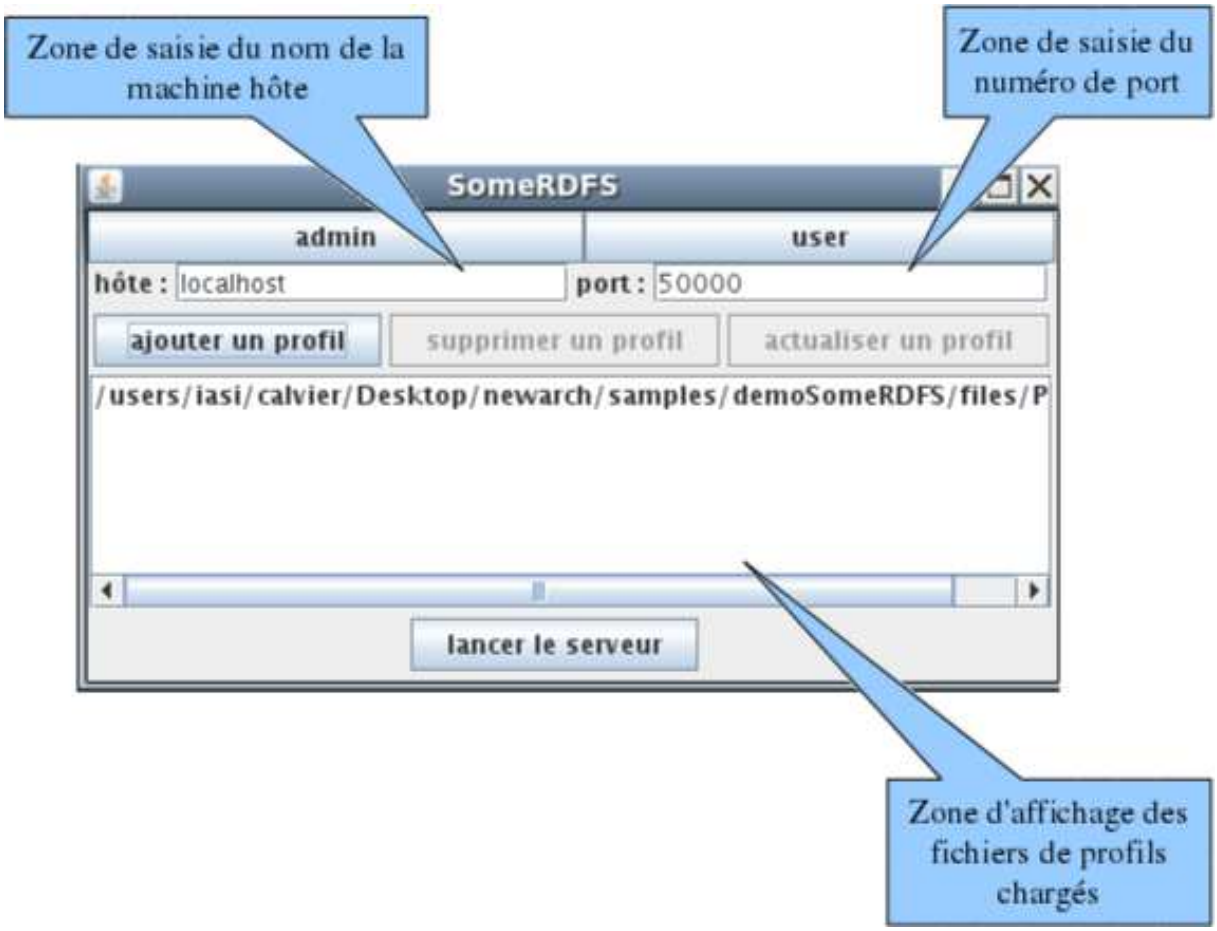

FIG. B.1 – Interface administrateur - sélection de profils

B.1.2 Le gestionnaire de profils

La création de profils

Pour créer un profil, on doit préciser l'ontologie RDF(S) sur laquelle s'appuie ce profil, des informations concernant l'adresse du pair (le nom de la machine et le port sur lesquel les requêtes seront attendues), et le nom du profil qui permet de distinguer les profils ayant la même adresse.
Nous proposons de donner la possibilité de créer des profils au sein du module permettant le chargement des profils. Lorsque l'administrateur souhaite ajouter un profil, il sélectionne le fichier RDF(S) à partir duquel le nouveau profil doit être créé. Une boîte de dialogue apparaît demandant de préciser le nom du nouveau profil. Ce profil est créé, sauvegardé dans un fichier de profil puis chargé par le serveur.

L'édition de profils

Les profils créés dans la section précédente ne contiennent aucun mapping. Pour modifier les profils il est nécessaire de travailler sur les fichiers les stockant. Nous proposons ici une interface permettant de travailler simultanément sur les deux versions d'un profil (Logique Propositionnelle et RDFS) et épargnant le coût d'une traduction du profil com-

plet lors d'une modification.

L'administrateur peut visualiser un profil chargé sur le serveur par un double clic sur le fichier correspondant dans la zone d'affichage des fichiers chargés. Le clic droit permet d'accéder à un menu permettant de visualiser ou d'éditer le profil. Un clic sur le bouton *modifier* fait apparaitre une fenêtre permettant de sélectionner le type de modification que l'on souhaite apporter. Une fois le choix effectué, une fenêtre spécifique (cf. figures B.2 et B.3) est affichée pour permettre de saisir la modification à apporter.

FIG. B.2 – Interface administrateur - ajout d'un lien ontologique dans un profil

FIG. B.3 – Interface administrateur - ajout d'un mapping dans un profil

B.1.3 La partie client de SomeRDFS

Interagir avec la partie client d'un pair nécessite de connaître son ontologie pour poser des requêtes. C'est la personne qui pose les requêtes, appelée utilisateur, qui est en relation avec cette partie de SomeRDFS.

Les différents types de requêtes que nous avons proposés peuvent être posés à partir de ce module client. Pour les requêtes locales comme pour les requêtes distantes, des informations sur le pair interrogé sont nécessaires. Cependant, dans le cas des requêtes distantes, ces informations sont explicitement données par la requête elle même, alors que, dans le cas des requêtes locales, ces informations doivent être collectées et traitées avant l'envoi de la requête.

La saisie des requêtes locales

Dans les requêtes locales, le pair auquel sont adressées les requêtes est appelé pair de référence. Son nom est précisé avant l'envoi de la requête.
Un client est supposé fortement lié à un serveur. De ce fait, l'adresse du pair de référence est fixée dans la partie serveur et n'a pas à être spécifiée lorsque des requêtes locales sont posées. Lorsque plusieurs profils ont été définis par l'administrateur, l'utilisateur doit sélectionner le profil qu'il souhaite utiliser pour le pair de référence.

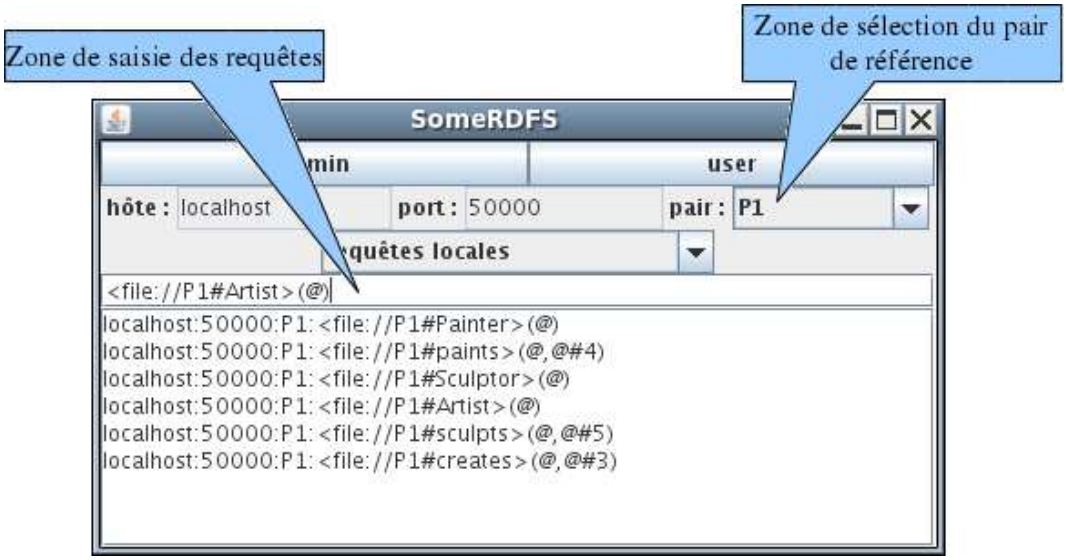

FIG. B.4 – Interface utilisateur

La pose de requêtes

L'utilisateur peut choisir le type de requête, locale ou distante, qu'il souhaite poser. Il peut ensuite saisir sa requête dans la zone de saisie prévue à cet effet. La syntaxe de la requête, donnée dans l'annexe C, permet de savoir s'il s'agit d'une demande d'évaluation ou d'une demande de réécritures. Une fois la saisie effectuée, le traitement sera alors spécifique au type de requête posée (évaluation / réécriture, locale / distante). Les réponses obtenues sont traitées et affichées dans la zone prévue à cet effet.

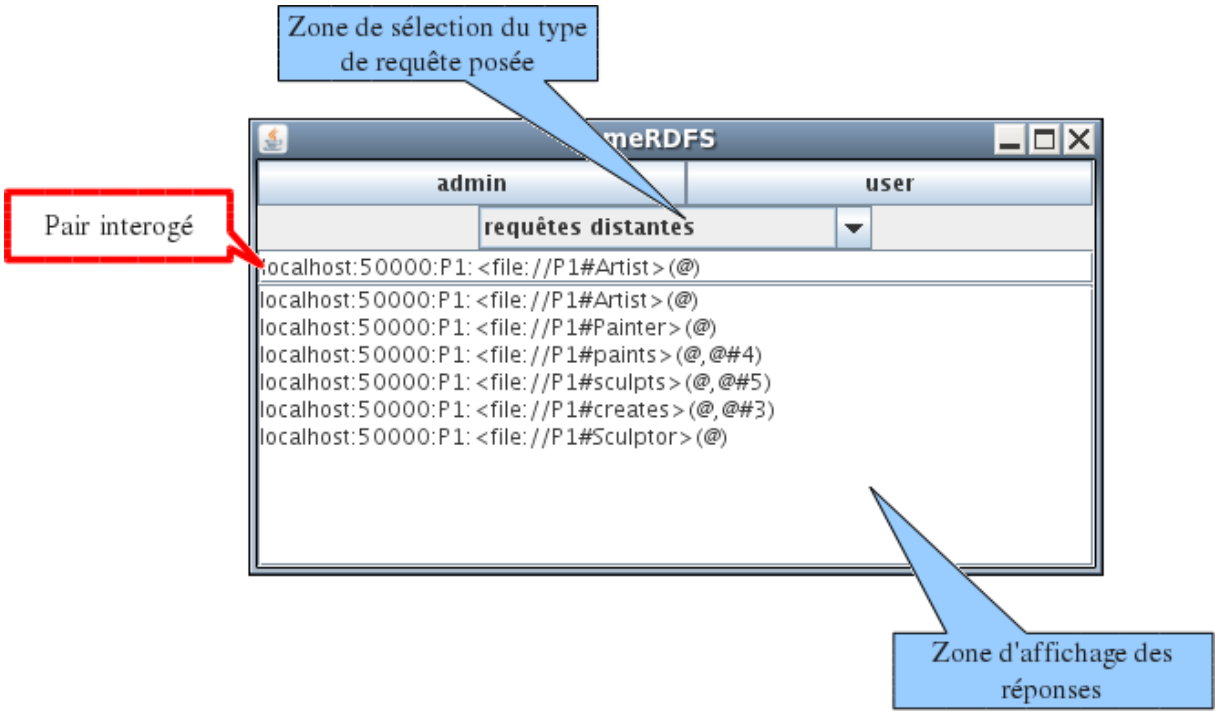

FIG. B.5 – Interface utilisateur - pose de requête

B.2 Interactions avec SpyWhere

L'utilisation de SpyWhere n'est pas systématique : elle relève d'une volonté de découvrir de nouveaux mappings. Nous proposons alors d'ajouter un bouton demandant, si besoin, l'activation de SpyWhere.

Une instance différente de SpyWhere existe pour chaque profil d'un serveur.

L'interface de SpyWhere se compose d'une fenêtre particulière (cf figure B.6) dans laquelle tous les mappings proposés sont affichés. Cette fenêtre est disponible dès lors que l'administrateur active SpyWhere. L'administrateur peut alors visualiser tous les mappings qui lui sont proposés. Les raccourcis de mappings générés après sélection (cf. 3.2) mais avant filtrage (cf. 3.3) et les couples d'éléments qu'il est pertinent d'aligner sont affichés (cf. B.6).

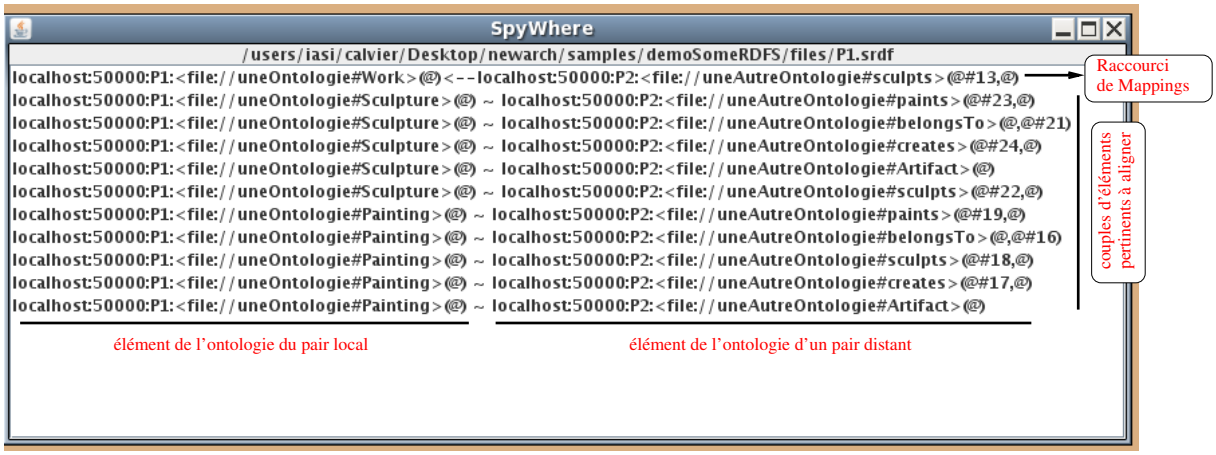

FIG. B.6 – Interface SpyWhere - visualisation des mappings proposés

Pour la détection de raccourcis de mappings, des boîtes de dialogue permettent de capturer les demandes d'évaluation et de savoir si l'utilisateur est satisfait des données obtenues en résultats des évaluations des réécritures. La recherche de candidats au mapping ne nécessite pas de fenêtres graphiques supplémentaires.

Annexe C

Développements réalisés relatifs à SomeRDFS

Notre étude s'appuie sur la plateforme SomeRDFS que nous avons décrite au chapitre 6.2. Cette plateforme, conçue par Philippe Adjiman dans sa thèse, n'était pas totalement implémentée : seul son noyau, SomeWhere, existait sous la forme d'un prototype. Nous avons donc terminé l'implémentation de SomeRDFS en nous inspirant de SomeOWL, un autre prototype développé par Philippe Adjiman, dont l'architecture est identique à celle de SomeRDFS.

Nous avons profité de cette tâche d'implémentation pour ajouter à SomeRDFS des fonctionnalités nécessaires pour son utilisation par SpyWhere.

Nous présentons ici l'implémentation des modules d'évaluation, de traduction et $DECA^{RDFS}$ que nous avons due réaliser.

C.1 Module d'évaluation

C.1.1 Description

Dans SomeWhere, il n'est pas possible d'associer des données à des clauses de la théorie puisque celles-ci sont exprimées en logique propositionnelle. Les requêtes ne sont donc pas évaluées. SomeWhere a toutefois été utilisé pour concevoir une autre application, SomeOWL, permettant de calculer des réponses à des requêtes OWL(PL). SomeOWL prend en compte les instances de classes. Ainsi, un fichier *nom_pair.xml* est spécifiquement utilisé pour stocker les données peuplant les éléments de l'ontologie du pair *nom_pair*. La forme de ce fichier est donnée figure C.1.

Le module d'évaluation exploitant ces fichiers a été intégré dans le client SomeWhere. Etant donnée une requête, une option ajoutée au client de SomeWhere permet ainsi d'obtenir à la fois les réécritures de cette requête et, pour chacune d'elles, leurs évaluations.

C.1.2 Adaptation effectuée

Nous avons adapté ce module d'évaluation de façon à ce qu'il corresponde mieux à nos besoins. Ces adaptations concernent d'une part les fichiers de données, d'autre part

```
<nom_pair>
<feuille nom="nom_relation1">
<url nom="URI1">ID1</url>
<url nom="URI2">ID2</url>
</feuille>
<feuille nom="nom_relation2">
<url nom="URI3">ID3</url>
</feuille>
</nom_pair>
```

FIG. C.1 – Schéma XML des fichiers de données

la conception du module.

Adaptation du fichier de données

SomeOWL n'exploitait qu'un seul type de relations, des relations unaires, permettant de représenter des classes. Dans SomeRDFS, nous devons également gérer des propriétés représentées par des relations binaires. L'adaptation du module d'évaluation à la gestion des relations binaires que nous proposons consiste à ne pas modifier la définition des arguments des relations mais à introduire un élément séparateur permettant de distinguer les deux arguments lorsque la chaîne de caractères introduite correspond précisément à la spécification de deux arguments. Ce caractère séparateur est le caractère " '". En effet, ce caractère est interdit dans le format des URL utilisé pour représenter les données.

Conception adaptée

Pour la découverte de mappings, nous avions besoin de pouvoir dissocier le calcul des réécritures de leur évaluation. La façon dont le module d'évaluation était conçu ne permettait pas l'évaluation sélective des réécritures. Nous avons alors repensé la conception du module. Tout d'abord, nous avons extrait le code correspondant à l'évaluation présent dans le client SomeWhere. Nous l'avons ensuite adapté pour qu'il ne s'agisse pas uniquement de fournir les données peuplant une relation mais d'évaluer des requêtes. Pour cela, il est nécessaire d'évaluer chacune de ses réécritures. Pour évaluer une réécriture complexe (faisant intervenir plus d'une relation), nous évaluons chacune des relations composant cette réécriture puis nous sélectionnons les données apparaissant dans toutes les évaluations. En rendant l'évaluation modulaire, nous autorisons deux types d'évaluation, en plus de l'évaluation de requêtes :

- l'évaluation d'un élément consistant à rechercher les données instanciant cet élément.
- l'évaluation d'une réécriture composée d'une conjonction d'éléments, consistant à rechercher les données instanciant l'ensemble de ces éléments.

C.2 Module de traduction

Le module de traduction se compose de quatre parties : d'un coté, RDF(S)2PL et RDF(S)2ASCII et, de l'autre, PL2RDF(S) et ASCII2RDF(S). Les deux premières parties correspondent à un encodage, les deux dernières à un décodage. Nous décrivons séparément chaque encodage, puis nous présentons les conditions qui garantissent les décodages.

RDF(S)2PL

Dans SomeRDFS, les connaissances des pairs sont représentées sous la forme de triplets (X Y Z) où X, Y et Z sont des noms de relations. X et Z peuvent être des propriétés ou des classes, les types de ces relations étant définis par Y qui est une relation prédéfinie de coreRDF(S). Y ne peut donc prendre que les valeurs *SubClassOf*, *SubPropertyOf*, *Range* ou *Domain*. Pour être utilisables par SomeWhere, les connaissances doivent être représentées par des clauses de logique propositionnelle de la forme $\neg A \vee B$ codées sous la forme *!A B*. Ainsi, les classes RDF(S) seront représentées par deux littéraux (CDOM et CRAN pour la classe C) et les propriétés par un seul littéral (PPROP pour la propriété P). L'encodage des triplets RDFS est effectué conformément aux règles suivantes (Adjiman *et al.* (2006)):

- R1 : (X SubClassOf Z) devient !XDOM ZDOM et !XRAN ZRAN.
- R2 : (X SubPropertyOf Z) devient !XPROP ZPROP
- R3 : (Z Range X) qui signifie que la portée de la propriété Z est typée par la classe X devient !ZPROP XRAN
- R4 : (Z Domain X) qui signifie que le domaine de la propriété Z est typé par la classe X devient !ZPROP XDOM

La représentation des classes par deux littéraux tient au fait que l'on a besoin de distinguer les typages de domaine et de co-domaine (cf. R3 et R4). Deux clauses sont ainsi nécessaires pour définir une sous-classe (cf. R1). Le module RDF(S)2PL est utilisé pour traduire un fichier contenant le profil d'un pair au format RDF(S) en un fichier contenant le profil d'un pair au format LP.

RDF(S)2ASCII

Nous avons fait le choix de manipuler des relations RDF(S) via leurs URI. Or certains caractères présents dans les URI, par exemple le caractère "*:*", sont des séparateurs pour SomeWhere. Un encodage des URI est donc nécessaire. Nous codons chaque caractère en ASCII en séparant chaque code par la lettre *a*. Ainsi le triplet
<http://mon_ontologie#ma_classe> Range <http://mon_ontologie#ma_propriete>
devient, dans le profil par défaut,
!localhost:50000:P1:a60a104a116a116a112a58a47a47a109a111a110a95a111a110a116a111a108a111a103a105a101a35a109a97a95a112a114a111a112a114a105a101a116a101a62PROP
localhost:50000:P1:a60a104a116a116a112a58a47a47a109a111a110a95a111a110a116a111a108a111a103a105a101a35a109a97a95a99a108a97a115a115a101a62RAN.

Décodage

Les réponses aux requêtes (données) n'ont pas besoin d'être décodées. En effet, bien que l'évaluation des réécritures se fasse sur des relations encodées, ceci n'a aucune conséquence sur les données associées. Ces dernières n'ayant pas été codées, peuvent être délivrées sans aucun traitement particulier. En revanche, lorsque l'évaluation n'est pas demandée, les réécritures obtenues par interrogation de SomeWhere doivent être décodées pour être présentées dans le même format que celui utilisé pour poser la requête initiale, i.e. RDF(S). Notons que ce processus de codage / décodage associé au fait qu'une réécriture peut contenir des éléments appartenant aux ontologies de différents pairs, nécessite que chacun des pairs utilise le même encodage.

C.3 $DECA^{RDFS}$

Dans les spécifications, $DECA^{RDFS}$ est un module permettant d'obtenir l'évaluation de requêtes composées de relations RDF(S). Le raisonnement utilisé par ce module est alors complètement transparent pour l'utilisateur. Or, les méthodes de découverte de mappings que nous proposons, exploitent ce raisonnement. Nous avons donc étendu ce module puis implémenté le module étendu. Le format des requêtes n'était pas spécifié de façon précise. Nous avons proposé un format en nous inspirant de la forme théorique des requêtes RDF(S), du format des requêtes SomeWhere et en intégrant les besoins d'expressivité du processus d'évaluation. Nous présenterons, successivement, le fonctionnement du module, les formats des requêtes et celui des réponses obtenues en sortie. En dernier point, nous reviendrons sur la nécessité d'une interface utilisateur.

C.3.1 Description de $DECA^{RDFS}$

La traduction d'une requête est plus complexe que la simple traduction de chacune des relations qui la composent. En effet, pour traduire une requête, il faut tenir compte des liaisons de variables. L'algorithme $DECA^{RDFS}$ (Adjiman *et al.* (2006)) permet de poser à SomeWhere l'ensemble des requêtes nécessaires pour obtenir toutes les réponses à une requête RDF(S) donnée. Ainsi, lorsqu'une requête posée à SomeRDFS porte sur une classe, deux requêtes sont posées à SomeWhere. En effet, chaque classe est traduite en deux clauses (cf. section C.2) qui font chacune l'objet d'une requête. Les requêtes étant conjonctives, le nombre de requêtes à poser à SomeWhere est exponentiel en fonction du nombre de classes qui composent la requête RDF(S). Par exemple, à la requête RDF(S) : $Q_1(X) \equiv C_1(X) \wedge C_2(X) \wedge C_3(X) \wedge P_1(X, a)$ où X est une variable et a une constante, correspondent 8 requêtes SomeWhere :

- $Q_1^1 \equiv C_1DOM \wedge C_2DOM \wedge C_3DOM \wedge P_1PROP$
- $Q_1^2 \equiv C_1RAN \wedge C_2DOM \wedge C_3DOM \wedge P_1PROP$
- $Q_1^3 \equiv C_1DOM \wedge C_2RAN \wedge C_3DOM \wedge P_1PROP$
- $Q_1^4 \equiv C_1RAN \wedge C_2RAN \wedge C_3DOM \wedge P_1PROP$
- $Q_1^5 \equiv C_1DOM \wedge C_2DOM \wedge C_3RAN \wedge P_1PROP$
- $Q_1^6 \equiv C_1RAN \wedge C_2DOM \wedge C_3RAN \wedge P_1PROP$
- $Q_1^7 \equiv C_1DOM \wedge C_2RAN \wedge C_3RAN \wedge P_1PROP$

– $Q_1^8 \equiv C_1RAN \wedge C_2RAN \wedge C_3RAN \wedge P_1PROP$

Par ailleurs, lorsqu'une propriété P est obtenue comme réécriture d'une classe C, la variable liée à la classe doit être associée à l'argument de P que type C et une autre variable non liée doit être associée à l'autre argument de P. Par exemple, si C_1 type le domaine de P_2, une réécriture de Q_1^1 peut être $R_1^{1,1} \equiv P_2PROP \wedge C_2DOM \wedge C_3DOM \wedge P_1PROP$ Dans ce cas, pour traduire cette réécriture en RDF(S), il est nécessaire d'introduire une variable libre Y. Ainsi la traduction de $R_1^{1,1}$ sera $P_2(X,Y) \wedge C_2(X) \wedge C_3(X) \wedge P_1(X,a)$.

C.3.2 Format des requêtes RDF(S)

Une requête RDFS $Q(X_1, X_2, \ldots, X_n) \equiv \bigwedge(\bigwedge_i(C_i(X_i)) \bigwedge_j(P_j(X_j^1, X_j^2)))$ sera posée sous la forme suivante : $(X_1\ X_2\ \ldots\ Xn)\ C_i(X_i)\ \ldots\ C_k(X_k)\ P_j(X_j^1, X_j^2)\ \ldots\ P_m(X_m^1, X_m^2)$. Le premier argument est utilisé pour l'évaluation. Il s'agit de la liste des variables pour lesquelles l'évaluation est demandée. Les arguments suivants sont les relations, qui composent la requête, séparées par des espaces. Les variables sont des chaînes de caractères débutant par @. Le caractère suivant ne doit pas être # car celui-ci est réservé pour les variables générées par $DECA^{RDFS}$. Toute chaîne de caractères ne débutant pas par @ est considérée comme une instance.

Dans les spécifications, SomeRDFS est pensé comme une interface permettant à un utilisateur connaissant l'ontologie d'un pair d'obtenir les données du PDMS qui sont pertinentes par rapport à une requête utilisant les relations de l'ontologie connue. Ainsi, dans SomeRDFS, un utilisateur ne peut poser une requête à un pair que dans le vocabulaire de ce pair. L'utilisateur précise donc, dans un premier temps, le pair auquel va s'adresser sa requête. Pour cela il indique l'adresse de ce pair. Par défaut cette adresse est *localhost:50000:P1*. La réponse attendue est l'ensemble des données pertinentes pour la requête posée. Un utilisateur ne pose ses requêtes qu'aux pairs dont il connait l'ontologie. Nous supposons qu'un utilisateur ne connait pas plus d'une ontologie. Nous nommons le pair associé à cette ontologie, *pair de référence*. Ce format de requête correspond alors à une demande d'évaluation posée au pair dit de référence.

Dans les spécifications existantes de SomeRDFS, le raisonnement effectué est totalement caché à l'utilisateur. Or les méthodes de découverte de mappings que nous proposons exploitent ce raisonnement. La méthode de découverte de raccourcis de mappings s'appuie sur la possiblité d'obtenir séparément les réécritures et les évaluations d'une requête donnée. Elle s'appuie aussi sur l'évaluation de relations n'appartenant pas au pair de référence. La méthode de découverte de candidats au mappings nécessite donc la possibilité de pouvoir poser des requêtes à des pairs différents du pair de référence, voire de pouvoir interroger différents pairs dans une même requête.

Pour répondre à ces différents besoins, nous avons étendu le format des requêtes dans le but de proposer trois façons supplémentaires d'interroger SomeRDFS. La forme globale d'une requête reste la même, $([(X_1\ X_2\ \ldots\ X_n)]\ C_i(X_i)\ \ldots\ C_k(X_k)\ P_j(X_j^1, X_j^2)\ \ldots\ P_m(X_m^1, X_m^2))$ mais elle est adaptée pour contenir toutes les informations nécessaires suivant le type de requête :

– demande d'évaluation d'une requête posée à des pairs quelconques du PDMS. Le nom d'une relation est complété par l'adresse du pair l'ayant définie. Ce type de requête est de la forme : $(X_1\ X_2\ \ldots\ X_n)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}C_i(X_i)\ \ldots\ nom\_hote_k{:}portk{:}nom\_pair_k{:}C_k(X_k)\ nom\_hote_j{:}portj{:}nom\_pair_j{:}P_j(X_j^1, X_j^2)\ \ldots\ nom\_hote_m{:}port_m{:}nom\_pair_m{:}P_m(X_m^1, X_m^2)$.
Par exemple : (@A @B) localhost:50000:P1:Peintre(@A) localhost:50000:P1:crée(@A, @B) localhost:50000:P2:Peinture(@B)

– demande de calcul des réécritures d'une requête posée à des pairs quelconques du PDMS. Le nom d'une relation est complèté par l'adresse du pair l'ayant définie. Ce type de requête est de la forme : $nom\_hote_i{:}port_i{:}nom\_pair_i{:}C_i(X_i)\ \ldots\ nom\_hote_k{:}port_k{:}nom\_pair_k{:}C_k(X_k)\ nom\_hote_j{:}port_j : nom\_pair_j : P_j(X_j^1, X_j^2)\ \ldots\ nom\_hote_m{:}port_m : nom\_pair_m{:}P_m(X_m^1, X_m^2)$.
Par exemple : localhost:50000:P1:Peintre(@A) localhost:50000:P1:crée(@A, @B) localhost:50000:P2:Peinture(@B)

– demande de calcul des réécritures d'une requête posée au pair de référence. Ce type de requête est de la forme : $C_i(X_i)\ \ldots\ C_k(X_k)\ P_j(X_j^1, X_j^2)\ \ldots\ P_m(X_m^1, X_m^2)$. Par exemple : Peintre(@A) crée(@A, @B)

Notons que, dans le cas des calculs de réécritures, le premier argument des requêtes contenant la liste des variables à évaluer est inutile. Dans le cas des requêtes adressées à des pairs quelconques, les adresses des pairs à interroger sont contenues dans les noms des relations composant la requête. Il n'est donc pas utile de les re-préciser. Dans ces requêtes, les noms de éléments sont de la forme suivante : $nom\_hote{:}port{:}nom\_pair{:}URI$.

Par ailleurs, nous avons autorisé (cf. C.1.2) des évaluations partielles permettant d'obtenir les données associées explicitement aux éléments composant la requête. Ce type de requête peut être intéressant pour l'évaluation des éléments d'un pair donné en isolant ses données propres des données obtenues du fait des mappings établis. Nous proposons donc de mettre à disposition deux types de requêtes supplémentaires.

– l'évaluation d'un élément. Ce type de requête peut être de la forme :
  – $(X)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}C_i(X)$
  – $(X)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}P_i(X\ Y)$
  – $(X)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}P_i(Y\ X)$
  – $(X\ Y)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}P_i(X\ Y)$
  Par exemple : (@A) localhost:50000:P1:Peintre(@A)

– l'évaluation d'une réécriture composée d'une conjonction d'élémentss. Le nom d'un élément est complété par l'adresse du pair l'ayant définie. Ce type de requête est de la forme : $(X_1\ X_2\ \ldots\ X_n)\ nom\_hote_i{:}port_i{:}nom\_pair_i{:}C_i(Xi)\ \ldots\ nom\_hote_k{:}port_k{:}nom\_pair_k{:}C_k(X_k)\ nom\_hote_j{:}port_j{:}nom\_pair_j{:}P_j(X_j^1, X_j^2))\ \ldots\ nom\_hote_m{:}port_m{:}nom\_pair_m{:}P_m(X_m^1, X_m^2))$.
Par exemple : (@A @B) localhost:50000:P1:Peintre(@A) localhost:50000:P1:crée(@A, @B) localhost:50000:P2:Peinture(@B)

C.3.3 Format de sortie

SomeRDFS a été développé dans le but d'être utilisé au travers d'une interface et d'autres modules logiciels. Aussi, le format des sorties n'est pas directement exploitable par un utilisateur. Deux formats de sortie sont possibles, selon que l'évaluation est demandée ou non.

Lorsque les réécritures sont demandées, la sortie est un ensemble de tableaux de relations où chaque tableau est une réécriture. Chaque cellule d'un tableau contient une chaîne de caractères représentant une relation sous la forme étendue hte:$port$:nom_pair:URI.

Lorsque l'évaluation est demandée, la sortie est un ensemble de tableaux où chaque tableau représente un nuplet correspondant à une réponse. Chaque cellule d'un tableau est associée à un élément du nuplet et contient une chaîne de caractères représentant une donnée. L'ordre des données pour une réponse correspond à l'ordre des variables utilisées pour poser la requête. Par exemple, si la requête est $Q_2(X, Y, Z) \equiv P_1(X, T)\ P_2(T, Y)\ P_3(Z, T)$, les réponses seront un ensemble de tableaux à trois colonnes et une ligne. La première colonne correspond aux données de la variable X, la seconde à celles de Y et la troisième à celle de Z.

C.3.4 Interface utilisateur

Les demandes d'évaluation et les demandes de réécritures ne sont pas traitées par les mêmes modules, une interface est donc nécessaire pour diriger chaque demande vers le module adéquat. De plus, les réponses à ces demandes nécessitent un traitement avant d'être délivrées à l'utilisateur. C'est ce même module interface qui récupère les réponses en sortie du module d'évaluation et de $DECA^{RDFS}$ pour les stocker dans un fichier. Nous décrivons plus en détail cette interface dans l'annexe B qui présente le travail réalisé pour l'ensemble des interfaces de SomeRDFS.

Bibliographie

Abdallah N. & Goasdoué F. (2008). Systèmes pair-à-pair sémantiques et extension non conservative d'une base de connaissances . In *BDA*.

Adjiman P. (2006). *Peer-to-Peer reasoning in propositional logic: algorithms, scalability study and applications*. PhD thesis, Université Paris-Sud.

Adjiman P., Chatalic P., Goasdoué F., Rousset M.-C. & Simon L. (2004). Distributed reasoning in a peer-to-peer setting. In *European Conference on Artificial Intelligence*, p. 945–946.

Adjiman P., Goasdoué F. & Rousset M.-C. (2006). SomeRDFS in the semantic web. *Journal on Data Semantics*, p. p. 158–181.

Aleksovski Z., Klein M. C. A., ten Kate W. & van Harmelen F. (2006a). Matching unstructured vocabularies using a background ontology. In *EKAW*, p. 182–197.

Aleksovski Z., ten Kate W. & van Harmelen F. (2006b). Exploiting the structure of background knowledge used in ontology matching. In *Ontology Matching*.

An Y., Borgida A. & Mylopoulos J. (2005). Inferring complex semantic mappings between relational tables and ontologies from simple correspondences. In *In ODBASE05*, p. 1152–1169.

Bouquet P., Magnini B., Serafini L. & Zanobini S. (2003). A sat-based algorithm for context matching. In *CONTEXT*, p. 66–79.

Bouquet P. & Serafini L. (2003). On the difference between bridge rules and lifting axioms. In *CONTEXT*, p. 80–93.

Castano S., Ferrara A. & Gianpaolo M. (2006). Results of the h-match ontology matchmaker in oaei 2006. In *Int. Workshop on Ontology Matching OM-2006*.

Castano S., Ferrara A. & Montanelli S. (2003a). H-match: an algorithm for dynamically matching ontologies in peer-based systems. In *SWDB 2003*, Berlin, Germany.

Castano S., Ferrara A., Montanelli S., Pagani E. & Rossi G. (2003b). Ontology-addressable contents in p2p networks. In *Proc. of the 1st WWW Int. Workshop on Semantics in Peer-to-Peer and Grid Computing (SemPGRID 2003)*, Budapest, Hungary. http://www.isi.edu/ stefan/SemPGRID/proceedings/proceedings.pdf.

Castano S. & Montanelli S. (2005). Semantic self-formation of communities of peers. In *Proc. of the ESWC Workshop on Ontologies in Peer-to-Peer Communities*, Heraklion, Greece.

Dieng R. & Hug S. (1998). Comparison of personal ontologies represented through conceptual graphs. In *ECAI*, p. 341–345.

Do H.-H. & Rahm E. (2007). Matching large schemas: Approaches and evaluation.

Doan A., Madhavan J., Domingos P. & Halevy A. Y. (2004). Ontology matching: A machine learning approach. In *Handbook on Ontologies*, p. 385–404.

Ehrig M. (2007). *Ontology Alignment: Bridging the Semantic Gap*, volume 4 of *Semantic Web And Beyond Computing for Human Experience*. Springer.

Ehrig M. & Staab S. (2004). Qom - quick ontology mapping. In *GI Jahrestagung (1)*, p. 356–361.

Elfeky M. G., Elmagarmid A. K. & Verykios V. S. (2002). Tailor: A record linkage tool box. In *ICDE*, p. 17–28.

Euzenat J. (2005). Alignment infrastructure for ontology mediation and other applications. In M. Hepp, A. Polleres, F. van Harmelen & M. Genesereth, Eds., *Proc. 1st ICSOC international workshop on Mediation in semantic web services, Amsterdam (NL)*, p. 81–95.

Euzenat J., Mocan A. & Scharffe F. (2008). Ontology alignments. In *Ontology Management*, p. 177–206.

Euzenat J. & Shvaiko P. (2007). *Ontology matching*. Heidelberg (DE): Springer-Verlag.

Euzenat J. & Valtchev P. (2004a). Similarity-based ontology alignment in owl-lite. In *ECAI*, p. 333–337.

Euzenat J. & Valtchev P. (2004b). Similarity-based ontology alignment in owl-lite. In *ECAI*, p. 333–337.

Fellegi I. P. & Sunter A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association*, **64**(328), 1183–1210.

Garey M. R. & Johnson D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Giunchiglia F. & Shvaiko P. (2003). *Semantic Matching*. Rapport interne, DISI, University of Trento.

Giunchiglia F., Shvaiko P. & Yatskevich M. (2004). S-match: an algorithm and an implementation of semantic matching. In *ESWS*, p. 61–75.

Giunchiglia F., Shvaiko P. & Yatskevich M. (2006). Discovering missing background knowledge in ontology matching. In *ECAI*, p. 382–386.

Gracia J., Lopez V., d'Aquin M., Sabou M., Motta E. & Mena E. (2007). Solving semantic ambiguity to improve semantic web based ontology matching. In *OM*.

Gruber T. R. (1993). Toward principles for the design of ontologies used for knowledge sharing. In *Formal Ontology in Conceptual Analysis and Knowledge Representation*: Kluwer Academic Publishers.

Halevy A. Y., Ives Z. G., Suciu D. & Tatarinov I. (2003). Schema mediation in peer data management systems. In *ICDE*, p. 505–.

Halevy Y., Ives G., Suciu D. & Tatarinov I. (2005). Schema mediation for large-scale semantic data sharing. *The VLDB Journal*, **14**(1), 68–83.

Hamdi F., Safar B., Niraula N. & Reynaud C. (2009). TaxoMap in the OAEI 2009 alignment contest. In *The Fourth International Workshop on Ontology Matching*, Chantilly, Washington DC. États-Unis.

Heese R., Herschel S., Naumann F. & Roth A. (2005). Self-extending peer data management. In *BTW*, p. 165–174.

Herschel S. & Heese R. (2005). Humboldt discoverer: A semantic p2p index for pdms. In *DISWeb'05*.

Jaccard P. (1901). Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. *Bulletin de la Société Vaudoise des Sciences Naturelles*, **37**, 241–272.

Jian N., Hu W., Cheng G. & Qu Y. (2005). Falconao: Aligning ontologies with falcon. In *K-CAP Workshop on Integrating Ontologies*, p. 87–93.

Kalfoglou Y. & Hu B. (2005). Crosi mapping system (cms) - result of the 2005 ontology alignment contest. In *Integrating Ontologies*.

Kalfoglou Y. & Schorlemmer M. (2003). Ontology mapping: the state of the art. *The Knowledge Engineering Review*, **18**, p. 1–31.

Kementsietsidis A., Arenas M. & Miller R. J. (2003). Mapping data in peer-to-peer systems: semantics and algorithmic issues. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, p. 325–336, New York, NY, USA: ACM.

Kotis K., Vouros G. A. & Stergiou K. (2006). Towards automatic merging of domain ontologies: The hcone-merge approach. *J. Web Sem.*, **4**(1), 60–79.

Langdon W. B. (2000). Natural language text classification and filtering with trigrams and evolutionary nearest neighbour classifiers.

Larson J. A., Navathe S. B. & Elmasri R. (1989). A theory of attribute equivalence in databases with application to schema integration. *IEEE Trans. Software Eng.*, **15**(4), 449–463.

Le B. T., Dieng-Kuntz R. & Gandon F. (2004). On ontology matching problems - for building a corporate semantic web in a multi-communities organization. In *ICEIS (4)*, p. 236–243.

Levenshtein V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, **10**(8), 707–710.

Lim E.-P., Srivastava J., Prabhakar S. & Richardson J. (1993). Entity identification in database integration. In *ICDE*, p. 294–301.

Limam L., Benbernou S., Hacid M.-S., Illarramendi A. & Ouksel A. M. (2008). Mapping discovery in p2p databases: a query-based approach. In *DBISP2P*, p. 26–41.

Lin D. (1998). An information-theoretic definition of similarity. In *ICML*, p. 296–304.

Madhavan J., Bernstein P. A. & Rahm E. (2001). Generic schema matching with cupid. In *Proceedings of the 27th International Conference on Very Large Data Bases*, p. 48–58: Morgan Kaufmann Publishers.

Melnik S., Garcia-molina H. & Rahm E. (2002). Similarity flooding: A versatile graph matching algorithm. In *ICDE*, p. 117–128.

Melnik S., Garcia-Molina H. & Rahm E. (2003). Rondo: A programming platform for generic model management. In *SIGMOD*, p. 193–204.

Mitra P., Noy N. F. & Jaiswal A. R. (2005). Ontology mapping discovery with uncertainty. In *in Fourth International Conference on the Semantic Web (ISWC-2005*, p. 537–547.

Mougin F., Burgun A. & Bodenreider O. (2006). Using wordnet to improve the mapping of data elements to umls for data sources integration. In *AMIA 2006*.

Médini L., Lunimeau N., da Silva C. F., Hoffmann P. & Ghodous P. (2007). Découverte de correspondances sémantiques par inférences dans un environnement P2P . In *workshop DECOR : Passage a l echelle des techinques de decouverte de correspondances of EGC 2007*.

Noy N. F. & Musen M. A. (2003). The prompt suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, **59**, 2003.

Pedersen T., Patwardhan S. & Michelizzi J. (2004). Wordnet: : Similarity - measuring the relatedness of concepts. In *AAAI*, p. 1024–1025.

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

Reynaud C. & Safar B. (2007). Techniques structurelles d'alignement pour portail web. In *Numéro spécial fouille du Web, Revue RNTI*.

Sabou M., d'Aquin M. & Motta E. (2006). Using the semantic web as background knowledge for ontology mapping. In *Ontology Matching*.

Saïs F., Pernelle N. & Rousset M.-C. (2009). Combining a logical and a numerical method for data reconciliation. *J. Data Semantics*, **12**, 66–94.

Sayyadian M., Lee Y., Doan A. & Rosenthal A. S. (2005). Tuning schema matching software using synthetic scenarios. In *In Proc. VLDB05*.

Schmid H. (1994). Probabilistic part-of-speech tagging using decision trees.

Shannon C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, **27**.

Sheth A. P., Larson J. A., Cornelio A. & Navathe S. B. (1988). A tool for integrating conceptual schemas and user views. In *ICDE*, p. 176–183.

Shvaiko P. (2006). *Iterative Schema-based Semantic MatchingIterative Schema-based Semantic Matching*. Phd thesis.

Shvaiko P. & Euzenat J. (2005). A survey of schema-based matching approaches. *Journal of Data Semantics IV*, p. 146–171.

Straccia U. & Troncy R. (2005). omap: Results of the ontology alignment contest. In *K-CAP Workshop on Integrating Ontologies*.

Stumme G. & Maedche A. (2001). Fca-merge: Bottom-up merging of ontologies. In *IJCAI*, p. 225–234.

Tatarinov I., Ives Z. G., Madhavan J., Halevy A. Y., Suciu D., Dalvi N. N., Dong X., Kadiyska Y., Miklau G. & Mork P. (2003). The piazza peer data management project. *SIGMOD Record*, **32**(3), 47–52.

Valtchev P. & Euzenat J. (1997). Dissimilarity measure for collections of objects and values. In *IDA*, p. 259–272.

Watts D. J. & Strogatz S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, **393**, 440–442.

Wu Z. & Palmer M. S. (1994). Verb semantics and lexical selection. In *ACL*, p. 133–138.

Zhang S. & Bodenreider O. (2006). Nlm anatomical ontology alignment system. results of the 2006 ontology alignment contest. In *Ontology Matching*.