RAPPORT DE RECHERCHE

L R I

GUIDING THE ONTOLOGY MATCHING PROCESS WITH REASONING IN A PDMS

CALVIER F E / REYNAUD C

Unité Mixte de Recherche 8623
CNRS-Université Paris Sud – LRI

06/2008

Rapport de Recherche N° 1495

CNRS – Université de Paris Sud
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bâtiment 490
91405 ORSAY Cedex (France)

Guiding the Ontology Matching Process with Reasoning in a PDMS

François-Elie Calvier, Chantal Reynaud

LRI Université Paris-Sud

Juin 2008

Abstract

This article focuses on ontology matching in a decentralized setting. The work takes place in the MediaD project. Ontologies are the description of peers data belonging to the peer data management system SomeRDFS.
First, we present some particularities of ontology matching in a peer-to-peer setting, the data model and query answering in SomeRDFS PDMSs. Then, we show how to take advantage of query answering in order to help discovering new mappings between ontologies, either mapping shortcuts corresponding to a composition of pre-existent mappings or mappings which can not be inferred from the network but yet relevant. This work has been partly implemented. We present the results of some experiments which confirm the interest of our approach.

Résumé

Dans cet article, nous nous intéressons à l'alignement d'ontologies dans un contexte distribué. Ce travail est réalisé dans le cadre du projet MediaD réalisé en partenariat avec France Telecom R&D. Les ontologies sont la description des données appartenant aux pairs du système pair à pair de gestion de données SomeRDFS. Dans une première partie, nous présentons les spécificités du processus de mise en correspondance en contexte distribué ainsi que la plateforme SomeRDFS. Nous montrons ensuite comment exploiter les mécanismes de raisonnement mis
en oeuvre dans SomeRDFS pour aider à découvrir de nouveaux mappings entre ontologies. Les mappings que nous cherchons à découvrir sont soit des raccourcis de mappings correspondant à une composition de mappings préexistants, soit des mappings qui ne peuvent pas être inférés à partir du PDMS mais qui sont pertinents. Ce travail a été partiellement implémenté. Nous présentons à la fin de ce papier quelques résultats d'expérimentations montrant l'intérêt de l'approche proposée.

Table of contents

1 Introduction

The use of peer-to-peer systems consists of querying for information to a network of peers. Queries are asked to one of the peers. The peers communicate to each other to answer queries in a collective way.
We focus on the ontology matching process in the peer data management system (PDMS) SomeRDFS (1). Ontologies are the description of peers data. Peers in SomeRDFS interconnect through mappings which are semantic correspondances between their own ontologies. Thanks to its mappings a peer may interact with the others in order to answer a query. A crucial aspect in SomeRDFS is that peers are equivalent in functionalities. No peer has a global view of the data management system. Each peer has its own ontology, its own mappings and its own data. It ignores the ontology, the mappings and the data of the other peers. In this setting, our work aims at increasing the mappings of the peers in SomeRDFS in order to increase the quantity and the quality of the answers of the whole data management system.

Our work takes place in the MediaD project[1], which aims at creating a peer-to-peer data management system allowing the deployment of very large applications that scale to thousands of peers. Earliest work produces the SomeWhere platform (2) based on a simple data model: propositional logic. Then, we deploied a PDMS using a data model based on RDF on top of the SomeWhere infrastructure. We called such a PDMS a SomeRDFS PDMS. The work described in this paper takes place in this setting. We are interested in identifying two kinds of mappings: mapping shortcuts corresponding to a composition of pre-existent mappings and mappings which can not be inferred from the network but yet relevant. In both cases, the idea is to make the generation of mappings automatically supported by query answering. We take also into account the strategy followed by a peer which is important to select the most relevant mappings to be represented. The method that we propose for discovering mapping shortcuts is based on query answering composed of two different and well-distinguished phases, rewriting and evaluation. The method that we propose for discovering the other kind of mappings is based on the identification of target relations. These relations are starting points in the mapping discovering process. They allow identifying relevant mapping candidates by limiting the matching process to a restricted set of elements. The main steps of the approach able to discover this second kind of mappings are the following: (1) looking for target relations, (2) looking for sets of mapping candidates for each target relation previously identified, (3) alignment of the elements of the sets of the mapping candidates. In this paper we focus only on steps 1 and 2. Future work will be devoted to step 3.

Thus, the paper is organized as follows. Section 2 defines the problem. We precise what makes the matching process specific in a distributed setting with reference to a centralized one. Then, we describe the fragment of RDFS that we consider as data model for SomeRDFS and query answering. Section 3 shows how the query answering process can be used to discover new relevant mappings according to a given strategy of a peer. First, we present the different strategies that can be followed by a peer. Then we present how mapping shortcuts and target relations can be identified using query answering. Finally we describe the techniques used to obtain a set of relevant mapping candidates from a set of target relations and according to a given strategy. In Section 4, we present experimentations made with SpyWhere, a prototype which partially implements our approach. Some related work is described in Section 5. We conclude and outline remaining research issues in Section 6.

2 Problem Definition

In this section, we present the specificities of a matching process performed on ontologies distributed over peers, data model and query answering in SomeRDFS PDMSs.

2.1 Matching Ontologies Distributed Over Peers

PDMSs combine features coming from peer-to-peer infrastructure and from traditional schema-based integration techniques.

Peer-to-peer computing consists of a network of distributed computational peers where the number of peers is unknown. Peers come and leave. The set of peers is highly dynamic. Moreover each peer is autonomous. There is no global control.

In information integration systems query answering is the main inference. When these systems are based on centralized mediators query answering process is usually achieved using a rewrite strategy.

[1]Research project funded by France Telecom R&D

In P2P data management systems each peer may have data to share with other peers and query processing is performed within the network. Users queries are propagated across the heterogeneous sources in a transparent way. The expected answers may be found all over the PDMS. In SomeRDFS PDMSs, data are semantically related with semantic mappings between the peers ontologies. Query answering takes into account the ontologies and is achieved using a rewrite and evaluate strategy.

We describe below how all these features influence the ontology matching process.

2.1.1 Huge Number of Elements to be Matched

In a centralized setting, matching ontologies often consists in a cartesian product of every element of both ontologies. One or more distance measures are computed and the results are analyzed. In PDMSs the unknown and a priori huge number of peers makes this kind of method unusable. Elements which are relevant to be matched must first be selected by filtering methods.

2.1.2 Distributed Ontology and Data

In a PDMS both the ontology and data are distributed through, respectively, the union of the peer ontologies and mappings and the union of the peer storage description. Each peer has a partial knowledge of the PDMS. In SomeRDFS, for example, it just knows its ontology, its mappings with the other peers and its own data. A peer can only access to a part of the data of a PDMS using its mappings. Mappings that can automatically be discovered, and usable discovering methods are so influenced by this restricted access.

2.1.3 Using Query Rewriting

Query rewriting provides information that can offer automated support for generating mappings. For example, the analysis of the answers may point that data are only provided by the original queried peer. In such a case it could be interesting to increase the mappings of this peer in order to obtain more answers.

2.2 Data Model of a SomeRDFS PDMS

In SomeRDFS ontologies and mappings are expressed in RDFS and data are represented in RDF. (Sub)classes, (sub)properties can be defined. Domain and range of properties can be typed. Classes inclusion, properties inclusion, domain and range typing of a property are the only authorized constructors. This language, denoted core-RDFS, has a clear and intuitive semantics. It is constructed on unary relations that represent classes and binary relations that represent properties. The logical semantics of core-RDFS, expressed in description logic (DL notation) and its translation in first-order logic (FOL), is given in Table 1.

Peers ontologies are made of core-RDFS statements involving the vocabulary of only one peer. A peer vocabulary is the union of a set of class names and a set of property names that are disjoint. The class and property names are unique to each peer. We use the notation $\mathcal{P}$:R for identifying the relation (class or property) R of the ontology of the peer $\mathcal{P}$.

Operator	DL Notation	FOL translation
Class inclusion	$C_1 \sqsubseteq C_2$	$\forall X, C_1(X) \Rightarrow C_2(X)$
Property inclusion	$P_1 \sqsubseteq P_2$	$\forall X \forall Y R_1(X,Y) \Rightarrow R_2(X,Y)$
Domain typing of a property	$\exists P \sqsubseteq C$	$\forall X \forall Y, P(X,Y) \Rightarrow C(X)$
Range typing of a property	$\exists P^- \sqsubseteq C$	$\forall X \forall Y, P(X,Y) \Rightarrow C(Y)$

Table 1: Core-RDF(S) operators

The specification of the data stored in a peer is done through the declaration of assertion statements relating data of a peer to relations of its vocabulary. The DL notation and the FOL translation of assertion statements are $C(a)$ and $P(a,b)$ where a and b are constants.

A mapping is an inclusion statement between classes or properties of two distinct peers (cf. Table 2 (a) and (b)) or a typing statement of a property of a given peer with a class of another peer (cf. Table 2 (c) and (d)). Mappings are defined as core-RDFS statements involving vocabularies of different peers.

Mappings	DL Notation	FOL translation
(a) Class inclusion	$\mathcal{P}_1{:}C_1 \sqsubseteq \mathcal{P}_2{:}C_2$	$\forall X, \mathcal{P}_1{:}C_1(X) \Rightarrow \mathcal{P}_2{:}C_2(X)$
(b) Property inclusion	$\mathcal{P}_1{:}P_1 \sqsubseteq \mathcal{P}_2{:}P_2$	$\forall X \forall Y, \mathcal{P}_1(X,Y) \Rightarrow \mathcal{P}_2(X,Y)$
(c) Domain typing of a property	$\exists \mathcal{P}_1{:}P \sqsubseteq \mathcal{P}_2{:}C$	$\forall X \forall Y, \mathcal{P}_1{:}(X,Y) \Rightarrow \mathcal{P}_2{:}C(X)$
(d) Range typing of a property	$\exists \mathcal{P}_1{:}P^- \sqsubseteq \mathcal{P}_2{:}C$	$\forall X \forall Y, \mathcal{P}_1{:}(X,Y) \Rightarrow \mathcal{P}_2{:}C(Y)$

Table 2: Mappings

2.3 Query Answering in a SomeRDFS PDMS

In SomeRDFS PDMSs ontologies, mappings and storage descriptions have all a FOL correspondence. Query rewriting is reduced to consequence finding over logical propositional theories solved by the DECA (DEcentralized Consequence finding Algorithm (1)) algorithm of SomeWhere, a propositional peer-to-peer reasoner. We illustrate the rewrite and evaluate strategy for query answering in a SomeRDFS PDMS on the following example.

Let us consider two peers $\mathcal{P}_1$ and $\mathcal{P}_2$, their ontologies (cf. Table 3), mappings (cf. Table 4) and data (cf. Table 5).

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

Table 3: Ontologies of $\mathcal{P}_1$ and $\mathcal{P}_2$

Let us consider the user query $Q_1(X) \equiv \mathcal{P}_1{:}Artifact(X)$ asked to the peer $\mathcal{P}_1$ to get all the artifacts known in the PDMS. Query answering is a two-step process: first, Q_1 is rewritten in a set of more specific queries. The set of all the rewritings of a query can be obtained from the conjunctions of the rewritings of each relation (property or class) of the query. Then, every rewriting is evaluated to get corresponding data.

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

Table 4: Mappings of $\mathcal{P}_1$ and $\mathcal{P}_2$

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

Table 5: Data of $\mathcal{P}_1$ and $\mathcal{P}_2$

One possible rewriting of $Q_1(X) \equiv \mathcal{P}_1{:}Artifact(X)$ is $\mathcal{P}_1{:}Painting(X)$. This means that one way to get data about $Artifact(X)$ is to get data about $Painting(X)$ in the same peer. Another rewriting can be $\mathcal{P}_2{:}Sculpture(X)$. That means that another way to get data about $Artifact(X)$ is to obtain data about $Sculpture(X)$ in $\mathcal{P}_2$. In this example rewritings are:

-$R_1(X) \equiv \mathcal{P}_1{:}Artifact(X)$ -$R_5(X) \equiv \exists Y, \mathcal{P}_1{:}Creates(Y,X)$
-$R_2(X) \equiv \mathcal{P}_1{:}Painting(X)$ -$R_6(X) \equiv \exists Y, \mathcal{P}_2{:}Sculpts(Y,X)$
-$R_3(X) \equiv \exists Y, \mathcal{P}_1{:}Belongsto(X,Y)$ -$R_7(X) \equiv \mathcal{P}_2{:}Sculpture(X)$
-$R_4(X) \equiv \exists Y, \mathcal{P}_1{:}Paints(Y,X)$

Let us note that $\mathcal{P}_1$ obtains R_2 by using a class inclusion statement and R_4 and R_5 by using the domain and range typing statements of the $\mathcal{P}_1{:}Paints$ and $\mathcal{P}_1{:}Creates$ properties. $\mathcal{P}_1$ uses the mappings $\forall X$ $\forall Y, \mathcal{P}_1{:}Belongsto(X,Y) \Rightarrow \mathcal{P}_2{:}Movement(Y)$ and $\forall X, \mathcal{P}_2{:}Sculpture(X) \Rightarrow \mathcal{P}_1{:}Artifacts(X)$ to get R_3 and R_7. Because of the relations of its vocabulary involved in the foreobtained rewritings, $\mathcal{P}_2$ is then queried. $\mathcal{P}_2$

will send obtained rewritings to $\mathcal{P}_1$ i.e. R_6. Finally, every rewriting is evaluated to get corresponding data. The answer set of Q_1 is:

$$Q_1(\mathcal{S}) = \underbrace{\emptyset}_{R_1(\mathcal{S})} \bigcup \underbrace{\{LaJoconde\}}_{R_2(\mathcal{S})} \bigcup \underbrace{\emptyset}_{R_3(\mathcal{S})} \bigcup \underbrace{\{lesdemoisellesd'Avignon\}}_{R_4(\mathcal{S})} \bigcup \underbrace{\emptyset}_{R_5(\mathcal{S})} \bigcup \underbrace{\{lePenseur\}}_{R_6(\mathcal{S})} \bigcup \underbrace{\{statuedeDavid\}}_{R_7(\mathcal{S})}$$

Users can pose unary or conjunctive queries. In case of conjunctive queries, rewritings are obtained from the conjunctions of the rewritings of each relation of the original query.

3 Exploiting SomeRDFS Reasoning

SomeRDFS reasoning, in particular, query answering can offer an automated support for discovering new mappings. We propose in this section a method to guide the ontology matching process based on query answering. Query answering is used to generate mapping shortcuts and to identify relations, denoted target relations, which are starting points in the mapping discovering process. These relations allow identifying relevant mapping candidates limiting that way the matching process to a restricted set of elements. Discovered mappings can be relevant or not according to the strategy involved in the PDMS. Thus, in a first sub-section we present the different strategies that can be followed by a peer. In the next sub-section, we present how mapping shortcuts and target relations can be identified using query answering. In the last sub-section, we describe the techniques used to obtain a set of relevant mapping candidates from a set of target relations and corresponding to a given strategy.

3.1 Strategies of a Peer

A PDMS can be seen as a very large data management system with a schema and data distributed through, respectively, the union of the peer ontologies and mappings, and the union of the peer storage description. It can also be viewed as a system where many peers each with its own ontology, mappings and data, choose to share data. In any case each peer has to access knowledge of the other peers. Having more mappings can be beneficial for three reasons. It is a way to access new data sources and so obtaining richer answers. It is a way to allow more precise queries assuming users are able to pose queries using the relations in mappings belonging to the vocabulary of distant peers. Finally, it is a way to make the PDMS steadier because less dependant of the comings and leavings of the peers in the network. Thus, any peer may decide to increase the number of its mappings. It can decide to look for new mappings whatever they are (strategy denoted S_1) or to look for particular mappings: either new mappings involving peers already logically connected to it (strategy denoted S_2) or mappings involving peers not yet logically connected to it (strategy denoted S_3). Two peers are logically connected if there exists a mapping between their two ontologies. The choice of one of these strategies depends on the number of already connected peers and on the number of mappings involving a given peer.

If a peer is logically connected to few other peers its access to the PDMS is unsteady and can easily be lost. To make its situation safer the peer will try and find mappings with not yet logically connected peers. The more peers a peer will be logically connected to the less it will be isolated. This is why this strategy makes the PDMS steadier. Moreover, new mappings can allow to establish direct links with relations that are more precise than the relations of the peer. They also may allow to obtain data which could not be obtained using query answering throughout the network.

A peer may be connected to another one through only a low number of connections. In that case, it can be relevant to try to establish more links with it before trying to learn from the rest of the PDMS. The more mappings there are between two peers the more complete description one peer has of the other one. New mappings make new data obtainable. It is also a way to favor answers to conjunctive queries because very often this kind of queries obtains answers only from rewritings containing relations belonging to the same peer.

3.2 Using Query Answering

3.2.1 Mappings Shortcuts Discovery

A mapping shortcut is a composition of mappings. Mapping shortcuts consolidate PDMSs by creating direct links between indirectly connected peers. In Figure 1, the $\mathcal{P}_1{:}Pianist(X) \Rightarrow \mathcal{P}_2{:}Artist(X)$ class in-

clusion is a mapping shortcut equivalent to the composition of $\mathcal{P}_1$:$Pianist(X) \Rightarrow \mathcal{P}_3$:$Musician(X)$ and $\mathcal{P}_3$:$Musician(X) \Rightarrow \mathcal{P}_2$:$Artist(X)$.

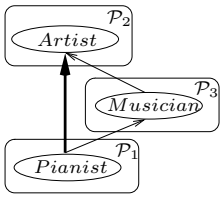


Figure 1: Mapping shortcut example

We could imagine automatically combining mappings in order to obtain shortcuts. Indeed, given a peer $\mathcal{P}$ systematic queries corresponding to each of its relation allows to identify mapping shortcuts involving each of them. However, this method generates a lot of traffic on the network and all the mappings obtained this way are not always useful. Mapping shortcuts are useful when some peers disappear from the PDMS. As an example, in Figure 1, each class belongs to a different peer. Without the $\mathcal{P}_1$:$Pianist(X) \Rightarrow \mathcal{P}_2$:$Artist(X)$ mapping shortcut if $\mathcal{P}_3$ leaves the network at any time $\mathcal{P}_1$ will not be able to access $\mathcal{P}_3$ or $\mathcal{P}_2$ anymore. However, they do not lead to more answers and they add caching in the rewriting process.

We propose a two-step automatic selection process. We first identify potentially useful mappings shortcuts exploiting query answering. In this step, the goal is to retain only mappings which would be useful in the rewriting process in regard to the queries really posed by users to the peer $\mathcal{P}$. However, all these mappings will not be systematically added to the set of mappings of $\mathcal{P}$ because the usefulness of some of them may be low. Thus, we propose then a second selection step based on filtering criteria which can be different from one peer to another.

To achieve the first step we need to distinguish the rewriting and evaluation phases of query answering. Query answering will not be a unique and global process anymore but two connected processes which can be separated if needed. Indeed, users do not always find the right needed relations in the ontology of the queried peer. In that case, they choose other relations among the relations of the queried peer. However, if a more specific relation is queried all the required data will not be obtained. On the other hand, if a more general relation is asked, all the required data will be obtained together with others, may be useless. For example, a user may need asking $\mathcal{P}_1$ for instances of $SteelSculptor$. Such a query can not be posed because of the lack of the $SteelSculptor$ relation in $\mathcal{P}_1$. The user could decide to ask for a more general relation, for example $\mathcal{P}_1$:$Artist(X)$. Rewritings obtained involve $\mathcal{P}_2$:$SteelSculptor(X)$ which is the relation he is interested in, but also $\mathcal{P}_2$:$WoodSculptor(X)$ and $\mathcal{P}_2$:$GlassSculptor(X)$. The evaluations of these two later relations are not needed with respect to the user's expectations. Considering rewriting as a process different from evaluation allows the user to examine the results of the rewriting phase in order to select which rewritings have to be evaluated.

The fact that the user selects rewritings that have to be evaluated is a good indicator of the relations he is really interested in. Thus, we propose to analyze the interactions between users and peers and to add mappings that are direct specialization links between the (more general) queried relation and the one the user has chosen to be evaluated. In this example, it would be $\mathcal{P}_2$:$SteelSculptor(X) \Rightarrow \mathcal{P}_1$:$Artist(X)$ added to $\mathcal{P}_1$. We consider that this mapping is a useful mapping shortcut. Note that if the user asks for the evaluation of several relations several mapping shortcuts will be proposed. For example, if a user of $\mathcal{P}_1$ is looking for instances of $MineralSculptor$, as $MineralSculptor$ is not a relation belonging to $\mathcal{P}_1$, his query may be $\mathcal{P}_1$:$Artist(X)$. Then, if the two relations $\mathcal{P}_2$:$SteelSculptor(X)$ and $\mathcal{P}_2$:$GlassSculptor(X)$, are required to be evaluated, $\mathcal{P}_2$:$SteelSculptor(X) \Rightarrow \mathcal{P}_1$:$Artist(X)$ and $\mathcal{P}_2$:$GlassSculptor(X) \Rightarrow \mathcal{P}_1$:$Artist(X)$ will be both identified as useful mapping shortcuts.

With regard to conjunctive queries, two cases can be distinguished. First, one of the relations in the query is not precisely the relation needed by the user. It is a relation in the vocabulary of the queried peer which is more general than the needed relation but the closest to it. In that case, we have first to distinguish between rewritings of the relations corresponding precisely to the needs of the user and those which do not satisfy his requirements and second to be able to make a correspondence between the relations in the query and their rewritings. If possible we have the same scenario than before dealing only with unitary queries. Thus, we would be able to propose adding mappings in the same way. Second, a conjunctive query can be a definition of what the user is looking for assuming that there is no relation within the queried peer vocabulary corresponding to his needs. Mapping shortcuts can be defined in that case when there are rewritings composed of only one relation. Those rewritings denote that the formula in the query refers to an entity in the domain. Thus, several mapping shortcuts can be added, one per relation in the query. In the same way mapping shortcuts can be defined when the number of relations in rewritings is lower than the number of relations in a query. We do not detail more discovering of mapping shortcuts based on conjunctive queries in this paper because of space limitations.

The second selection step is based on the strategy of the peer and potentially exploits filtering criteria defined by the administrator of this peer. Indeed, according to the strategy S_2 or S_3 chosen by a peer $\mathcal{P}$ only a subset of the mapping shortcuts will be considered: mappings only involving peers not yet connected to $\mathcal{P}$ will be retained for S_2, only mappings involving already connected peers will be retained for S_3. Then a peer may want to operate

a finer selection using additional filtering criteria. The usable criteria are specific to each peer but are limited. They concern either the kind of user (member of a particular group or of a given category: permanent users, temporary users, users making an intensive use of the peer, ... assuming that the group and the category are given when a user registers) who posed the query which originated the mapping (user-criterion) or the kind of relation belonging to $\mathcal{P}$ involved in the mapping (relation-criterion). The favored relations can be indicated one by one or according to their level in the hierarchy. We can, for instance, favor mappings establishing a connection with the n last levels in the class or property hierarchy of the ontology. A value is associated to each criterion, 1, 0.5 or 0, depending on whether the involved mapping has to be more or less favored. Let us note that the same mapping can be obtained several times from different queries potentially posed by different users. The weight of the user-criterion may be different from one mapping to another but the weight of the relation-criterion will always be the same. Thus, we propose a relevance measure for the mapping shortcuts which takes into account the weight of each additional filtering criterion but also the number of times that the mapping was obtained. Table 6 gives the value of the relevance measure of a mapping shortcut m_j when this mapping has been obtained n times given a sample of studied shortcut mappings composed of M elements. $W(U_{i,j})$ is the weight of the user-criterion for the occurrence i of the mapping m_j. $W(R_j)$ is the weight of the relation-criterion for the relation R_j.

We favor mapping shortcuts that are obtained a high number of times. They must be added to the set of mappings of the peer $\mathcal{P}$ whatever the user and the relation are. When the number n of times that a mapping is obtained is relatively high but not very high the relevance measure will be at least 0.5. That means that the number of times the mapping has been obtained is considered but with a medium value. When the number of times that a mapping is obtained is low (less than 50 of M), its relevance measure only depends on the average of the weights of the criteria.

n : # occurrences of m_j	relevance measure
$n \geq 80\% \times M$	1
$50\% \times M \leq n < 80\% \times M$	$\text{Max}\left(0.5, \frac{\sum_{i=1}^{n} W(U_{i,j}) + n \times W(R_j)}{2n}\right)$
$n < 50\% \times M$	$\frac{\sum_{i=1}^{n} W(U_{i,j}) + n \times W(R_j)}{2n}$

Table 6: Relevance measure of the mapping m_j with n occurrences

3.2.2 Identification of Target Relations Using Query Answering

In SomeRDFS mappings are the key notion for establishing semantic connections between the peers ontologies. They are used to rewrite queries posed in terms of a local ontology and rewritings are then run over the ontology of logically connected peers. That way, distant peers may contribute to answers. However, when a user interrogates the PDMS through a peer of his choice answers may come only from the original queried peer. This reveals a lack of specialization mappings. For instance, let consider the query $Q_3(X) \equiv \mathcal{P}_1{:}Painter(X)$ asked to $\mathcal{P}_1$ and the rewriting $\exists Y, P_1{:}Paints(X, Y)$. The relation in the answer only comes from $\mathcal{P}_1$. That means that there is no specialization mapping for $\mathcal{P}_1{:}Painter(X)$. Such relations are blocking points in query answering. They are what we call target relations. Our objective is to identify them and to consider them as starting points in the ontology matching process in order to discover mappings establishing connections with distant peers and so limiting answers provided only by one peer.

In the previous example we illustrated the notion of target relation by considering that these relations have no specialization mapping. However, this example is a simplification to present the intuition behind this notion. In fact, in our approach we consider that a relation is a target relation if it is an obstacle for its peer in achieving the strategy it has chosen to implement. As the different strategies that we consider (cf. Section 3.1) rely on the number of logically connected peers and on the number of specialization mappings the definition of a target relation will be based on a counting function. That function will differ according to the strategy of the peer and also according to the method used to count. Indeed, given a relation R of a peer $\mathcal{P}$, the number of distant relations involved together with R in RDFS statements, either specialization mappings of $\mathcal{P}$ or locally inferred statements, can be calculated either with regard to the knowledge of the peer $\mathcal{P}$ or with regard to rewritings obtained from queries. This is also true when given a relation R of a peer $\mathcal{P}$ we want to compute the number of distant peers corresponding to relations involved in specialization mappings of R_1 or locally inferred statements. The result of the counting function will be compared to a threshold that will be fixed by the administrator of the peer. When the value of the function is lower than the threshold the relation will be a target relation. We

first give a general definition of a target relation. We will then precise the general definition to handle all the different cases (one strategy has been chosen or not - counting is done in regard to peers' knowledge or rewritings).

Definition 1 (Target Relation) $\mathcal{P}_1{:}R_1$ is a target relation iff f($\mathcal{P}_1{:}R_1$) < t, f being a counting function and t a threshold.

In Table 7, we precise the definition of the function f for the relation R_1 of the peer $\mathcal{P}_1$ according to the strategy chosen by the peer and according to the method used to count. Three strategies are mentioned. The strategy S_1 is the default strategy. A peer which has this strategy will try to find new mappings, whatever they are. The two others are described in section 3.1. In Table 7, we precise the definition of the function f for the relation R_1

Strategy of a peer \ Method used to count	C_1 (in regard to the knowledge of $\mathcal{P}_1$)	C_2 (based on rewritings)
S_1	$\lvert\{\mathcal{P}_{i\neq 1}{:}R_j$ / $[\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_1]$ or $[\exists \mathcal{P}_1{:}R_k$ such that $\mathcal{P}_1{:}R_k \Rightarrow \mathcal{P}_1{:}R_1$ can be inferred and $\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_k]\}\rvert$	$\lvert\{\mathcal{P}_i{:}R_j \in RW\}\rvert$ where RW is the query rewriting set of $Q \equiv \mathcal{P}_1{:}R_1$
S_2	$\lvert\{\mathcal{P}_{i\neq 1}$ / $\exists \mathcal{P}_i{:}R_j$ such that $[\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_1]$ or $[\exists \mathcal{P}_1{:}R_k$ such that $\mathcal{P}_1{:}R_k \Rightarrow \mathcal{P}_1{:}R_1$ can be inferred and $\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_k]\}\rvert$	$\lvert\{\mathcal{P}_{i\neq 1}$ / $\mathcal{P}_i{:}R_j \in RW\}\rvert$ where RW is the query rewriting set of $Q \equiv \mathcal{P}_1{:}R_1$
S_3	$\min_i(\lvert\{\mathcal{P}_i{:}R_j$ / $\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_1$ or $[\exists \mathcal{P}_1{:}R_{k\neq 1})$ such that $\mathcal{P}_1{:}R_k \Rightarrow \mathcal{P}_1{:}R_1$ can be inferred and $\mathcal{P}_i{:}R_j \Rightarrow \mathcal{P}_1{:}R_k]\}\rvert)$	$\min_i \lvert\{\mathcal{P}_{i\neq 1}{:}R_j \in RW\}\rvert$ where RW is the query rewriting set of $Q \equiv \mathcal{P}_1{:}R_1$

Table 7: Definition of f($\mathcal{P}_1{:}R_1$)

of the peer $\mathcal{P}_1$ according to the strategy chosen by the peer and according to the method used to count. Three strategies are mentioned. The strategy S_1 is the default strategy. A peer which has this strategy will try to find new mappings, whatever they are. The two others are described in section 3.1

A peer which chooses the strategy S_2 looks for mappings involving peers not yet connected to it. A peer which implements the strategy S_3 looks for new mappings involving peers already connected to it. C_1 and C_2 are two counting methods. C_1 operates in regard to the knowledge of the peer, its ontology and its mappings. C_2 is based on rewritings obtained from queries. The target relations obtained using the counting function C_2 will be different from the target relations obtained using C_1; this is because C_2 takes into account distant relations which belong to rewritings produced by connected distant peers but not distant relations coming from disconnected peers and C_1 does the opposite.

If the strategy of $\mathcal{P}_1$ is S_1 and if C_1 is used the result of f($\mathcal{P}_1{:}R_1$) is the number of distant relations specializing R_1 according to the mappings of $\mathcal{P}_1$ or specializing another relation R_k of $\mathcal{P}_1$ with $R_k \Rightarrow R_1$ locally inferred. Using C_2 the result of f($\mathcal{P}_1{:}R_1$) will be the number of distant relations belonging to the rewritings of R_1. If this number of distant relations is lower than the threshold t then R_1 will be a target relation.

If the strategy of $\mathcal{P}_1$ is S_2 and if C_1 is used the result of f($\mathcal{P}_1{:}R_1$) is the number of distant peers involved in specialization mappings of R_1 or in statements specializing another relation R_k of $\mathcal{P}_1$ with $R_k \Rightarrow R_1$ locally inferred. If this number of distant peers is lower than the threshold t then R_1 will be a target relation. Using C_2 the result of f($\mathcal{P}_1{:}R_1$) will be the number of distant peers involved in the rewritings of R_1.

If the strategy of $\mathcal{P}_1$ is S_3, R_1 will be a target relation if there is at least one peer involved in not enough specialization statements of R_1. Thus, if C_1 is used, f($\mathcal{P}_1{:}R_1$) provides the minimum number of relations of a given distant peer specializing R_1 according to the mappings of $\mathcal{P}_1$ or specializing another relation R_k of $\mathcal{P}_1$ with $R_k \Rightarrow R_1$ locally inferred. If C_2 is used, f($\mathcal{P}_1{:}R_1$) will provide the minimum number of distant relations which belong to the rewritings of R_1 and which are involved in the mappings of $\mathcal{P}_1$.

Let us note that as the two counting methods are based either on the knowledge of a peer or on the rewritings of queries the computation may differ in time. Applying f using C_1 can be done off-line. Distant relations will be considered once a logical connection exists. On the contrary, using C_2, distant relations will only be considered if their peer is really connected. Moreover, C_2 will consider distant relations which belong to rewritings produced by connected distant peers and which cannot be obtained locally by using C_1. Furthermore, applying f using C_2 relies on rewritings which depend on queries. That means that the counting must be done several times. We assume that the administrator defines what several times means. Once we know how many measures are necessary, we can wait until enough measures are obtained. Another solution can be to store each query posed by a user, and to pose it again and again until enough measures are obtained according to a frequency fixed by the administrator.

3.3 Obtaining a Set of Relevant Mapping Candidates

Our objective is to use target relations in order to identify relevant mapping candidates, limiting that way the matching process to a restricted set of elements. In this section, we propose methods to discover new mappings from target relations. These methods are performed by a given peer given its target relations.

Target relations can allow discovering relevant mapping candidates according to two scenarios. In the first scenario, let us consider $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$ three peers with C_1, C_2 and C_3 three classes and the following mappings: $\mathcal{P}_1$:$C_1(X) \Rightarrow \mathcal{P}_2$:$C_2(X)$ and $\mathcal{P}_3$:$C_3(X) \Rightarrow \mathcal{P}_2$:$C_2(X)$, each known by the two involved peers. This scenario is represented Figure 2.

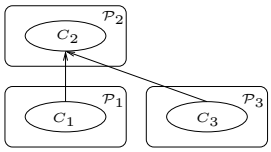


Figure 2: Scenario 1

From the point of view of $\mathcal{P}_1$ $C_1(X)$ is a target relation because there is no distant relation specializing $C_1(X)$. The query $Q_4(X) \equiv \mathcal{P}_1$:$C_1(X)$ has no rewriting. That target relation is interesting since $\mathcal{P}_1$:$C_1(X) \Rightarrow \mathcal{P}_2$:$C_2(X)$ is a mapping in $\mathcal{P}_1$, $Q_5(X) \equiv \mathcal{P}_2$:$C_2(X)$ could be a query posed to $\mathcal{P}_2$ by $\mathcal{P}_1$. The obtained rewritings would be $\mathcal{P}_1$:$C_1(X)$ and $\mathcal{P}_3$:$C_3(X)$ and looking for mappings between all the relations belonging to this set of rewritings is relevant. Indeed, it could allow to discover the mapping $\mathcal{P}_3$:$C_3(X) \Rightarrow \mathcal{P}_1$:$C_1(X)$ making that way a connection between $\mathcal{P}_3$ and $\mathcal{P}_1$. Note that, according to this scenario 1, the peers $\mathcal{P}_1$ and $\mathcal{P}_2$ can be the same, and $\mathcal{P}_2$ and $\mathcal{P}_3$ too.

In the second scenario let us consider $\mathcal{P}_1$ and $\mathcal{P}_2$ two peers, $\mathcal{P}_1$:C_1, $\mathcal{P}_2$:C_2 and $\mathcal{P}_2$:C_3 three classes. $\mathcal{P}_2$:$C_2(X) \Rightarrow \mathcal{P}_2$:$C_3(X)$ is a statement in $\mathcal{P}_2$. $\mathcal{P}_2$:$C_2(X) \Rightarrow \mathcal{P}_1$:$C_1(X)$ is a mapping in $\mathcal{P}_2$ and $\mathcal{P}_1$. This scenario is represented Figure 3.

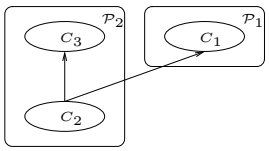


Figure 3: Scenario 2

From the point of view of $\mathcal{P}_2$ $C_2(X)$ and $C_3(X)$ are target relations because there is no distant relation specializing $C_2(X)$ nor $C_3(X)$. The query $Q_6(X) \equiv \mathcal{P}_2$:$C_3(X)$ has only one local rewriting which is $\mathcal{P}_2$:$C_2(X)$. No distant relations belong to the rewritings. This scenario is also interesting since $\mathcal{P}_2$:$C_2(X) \Rightarrow \mathcal{P}_1$:$C_1(X)$ is a mapping in $\mathcal{P}_2$, it could be relevant to look for mappings between $C_1(X)$ and $C_3(X)$, two relations which subsume $C_2(X)$. It could allow to discover the mapping $\mathcal{P}_1$:$C_1(X) \Rightarrow \mathcal{P}_2$:$C_3(X)$ establishing a connection between $\mathcal{P}_2$ and $\mathcal{P}_1$ usable to rewrite $\mathcal{P}_2$:$C_3(X)$.

Let us note that the $\mathcal{P}_1$:$C_1(X) \Rightarrow \mathcal{P}_2$:$C_2(X)$ mapping in scenario 1 and the $\mathcal{P}_2$:$C_2(X) \Rightarrow \mathcal{P}_2$:$C_3(X)$ mapping in scenario 2 can be locally inferred in $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. Furthermore, these two scenarios are elementary and could be combined in order to deal with more complex ones. These two scenarios use that target relations as starting points for the identification of relevant mapping candidates. However, all target relations will not allow finding relevant mapping candidates. Thus, we just consider target relations in regard to the two scenarios described above.

For each target relation we look for sets of mapping candidates, denoted MC. Our approach is based on the idea that it is relevant to look for connections between relations if they have common points. In our setting the common point that we are going to consider is a common relation, either more general or more specific. The construction of the set of mapping candidates can be achieved according to two processes, one for each scenario.

Specific Candidates Algorithm:

This algorithm is performed for target relations with one or more general relations, R_g, according to the knowledge of its peer (according to the ontology or to the mappings). This scenario is represented Figure 2 with C_2 in the place of R_g. In that case, we propose to pose the query $Q(X) \equiv R_g(X)$ in order to obtain its rewritings. The set of the rewritings is MC. It is composed of relations that are more specific than R_g.

Algorithm 1 SPEC_MC(P_i:R_j)

Require: A target relation, R_j, of $\mathcal{P}_i$
Ensure: Output contains the set of relations which are mapping candidates associated with R_j
  for all $\mathcal{P}_k$:R_m relation more general than $\mathcal{P}_i$:R_j **do**
    Add Rewrite($\mathcal{P}_k$:R_m) to $MC(\mathcal{P}_i$:$R_i)$
  end for

General Candidates Algorithm:

This algorithm is performed for target relations R_s with several (at least two) more general relations according to the knowledge of its peer (according to the ontology or to the mappings). This scenario is represented Figure 3

with C_2 in the place of R_s. In that case, all the more general relations are members of the set MC.

Algorithm 2 GEN_MC(P_i:R_j)

Require: A target relation, R_j, of $\mathcal{P}_i$
Ensure: Output contains the set of relations which are mapping candidates associated with R_j
  for all $\mathcal{P}_m$:R_n directly more general than $\mathcal{P}_i$:R_j **do**
    Add $\mathcal{P}_m$:R_n to $MC(\mathcal{P}_i$:$R_j)$
  end for

Let us note that a target relation may sometimes not match with any of these scenarios. A target relation which matches with one of them will be called a target relation with mapping candidates (TR_MC). Furthermore, one can decide to restrict MC to relations belonging to some given peers in order to represent mappings only with these particular peers (strategy S_2 or S_3).

4 Experiments

We conduct experiments to evaluate our approach with four SomeRDFS peers included local SpyWhere modules in the publication domain. We particularly focus on the applicability of our approach when very few mappings have already been specified. Our goal was to evaluate the identification process of target relations and of mapping candidates, and also to test counting methods and strategies used in this process.

Ontologies: Each peer had its own ontology in the publication domain whose characteristics are shown in Table 8. We tried to choose ontologies on the Web modelling publications from different points of view and being different according to their size (number of relations) and to encoded details (simplified or general views / more specialized ones). $\mathcal{P}_1$, $\mathcal{P}_3$ and $\mathcal{P}_4$ ontologies mainly describe conferences: events, publications, people. In all of them *Publication* is a root node of a tree but in $\mathcal{P}_4$ publications are described more precisely: 47 % of the relations in the ontology are specialized relations of *Publication*. $\mathcal{P}_2$ is much less similar to the other ones. It describes documentation in a library. The root node is *Documentation* and *Publication* is a direct descendant of *Documentation*.

Peer	Ontology	# relations	# *Publication* sub-relations
$\mathcal{P}_1$	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

Table 8: Descriptions of the ontologies

Experiments: We began by adding 3 mappings in $\mathcal{P}_1$ each of them being an inclusion statement between the class *Publication* in $\mathcal{P}_1$ and the class *Publication* in $\mathcal{P}_2$, $\mathcal{P}_3$ and $\mathcal{P}_4$, respectively. Mappings are oriented such that a distant relation is always a sub-relation of the local one. That way each peer can access to parts of the data of the PDMS. $\mathcal{P}_1$ is directly connected to all the other peers and $\mathcal{P}_2$, $\mathcal{P}_3$ and $\mathcal{P}_4$ are connected to all the other peers but sometimes not directly. Only the connection with $\mathcal{P}_1$ is direct. Then we look for target relations in each peer with regard to its knowledge. Table 9 shows the number of target relations being obtained with a threshold t = 1, with any strategy and counting method. Since there are only few mappings in each peer at this stage most of the relations are target relations (cf. *# target relations* in Table 9). Then we select the target relations that match scenario 1, the only scenario applicable in that case. The number of these relations (cf. *# TR_MC*) varies from 8 % to 92 % of the number of the relations in the ontology. It grows with the size of the sub-tree containing *Publication*. In this very simple case mapping candidates are easy to calculate. Whatever TR_MC may be $MC(TR\_MC)$ is the set of local or distant relations which specializes the root of the sub-tree including TR_MC. The alignment

Peer	# relations	# target relations	# TR_CM
$\mathcal{P}_1$	172	171	14
$\mathcal{P}_2$	95	93	87
$\mathcal{P}_3$	144	143	36
$\mathcal{P}_4$	231	230	110

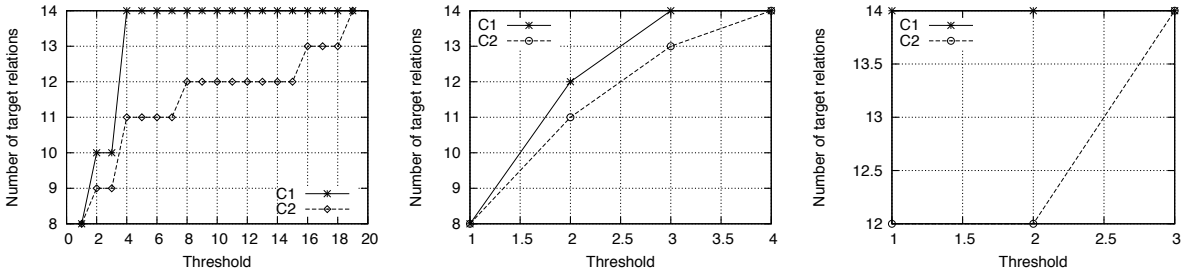
Table 9: Number of target relations obtained from the first experiment

process will then compare elements of the set two by two, a local relation to a distant one. That way each of the 36 TR_MC of $\mathcal{P}_3$ will be compared to 150 relations belonging to the other peers. To reduce the number of comparisons one can choose to focus only on relations belonging to some given peers. Thus, for example, if $\mathcal{P}_3$ considers only $\mathcal{P}_1$ relations in the alignment process each of its 36 TR_MC will be compared to only 14 relations. This proves the relevance of the applicability of S_2 and S_3 strategies.

$\mathcal{P}_3$:Book ⇒ $\mathcal{P}_1$:Book	$\mathcal{P}_4$:Volume_Book ⇒ $\mathcal{P}_1$:Volume	$\mathcal{P}_4$:Journal ⇒ $\mathcal{P}_1$:Journal
$\mathcal{P}_4$:Book ⇒ $\mathcal{P}_1$:Book	$\mathcal{P}_4$:Volume_Article ⇒ $\mathcal{P}_1$:Volume	$\mathcal{P}_3$:contains_article ⇒ $\mathcal{P}_1$:Journal
$\mathcal{P}_4$:author_Book ⇒ $\mathcal{P}_1$:Book	$\mathcal{P}_4$:Volume_Proceeding ⇒ $\mathcal{P}_1$:Volume	$\mathcal{P}_2$:Book ⇒ $\mathcal{P}_3$:Book
$\mathcal{P}_4$:author_Book ⇒ $\mathcal{P}_1$:listed_author_in	$\mathcal{P}_4$:keyword ⇒ $\mathcal{P}_1$:keyword	$\mathcal{P}_2$:Article ⇒ $\mathcal{P}_4$:Article
$\mathcal{P}_4$:author_Article ⇒ $\mathcal{P}_1$:listed_author_in	$\mathcal{P}_4$:Article ⇒ $\mathcal{P}_1$:Article	
$\mathcal{P}_4$:author_Unpublished ⇒ $\mathcal{P}_1$:listed_author_in	$\mathcal{P}_3$:Journal ⇒ $\mathcal{P}_1$:Journal	

Table 10: Added mappings

We made a second experiment adding 16 new mappings (cf. Table 10) in $\mathcal{P}_1$ involving 6 local relations which are direct sub-classes of *Publication*. We computed the number of target relations obtained by $\mathcal{P}_1$ as a function of the threshold t using the counting methods C_1 and C_2 according to the 3 strategies S_1, S_2 and S_3 (cf. Figure 4). Whatever the strategy or the counting method we note that 8 of $\mathcal{P}_1$ relations are always defined as target relations



(a) S_1 (b) S_2 (c) S_3

Figure 4: Number of target relations with the different strategies and counting methods

when the threshold is low (relations not involved in mappings in that experiment) and 14 (equal to ***# TR_MC*** in Table 9) when it is high. Let us note that the result obtained using C_2 assumes that all peers are connected. Given this same hypothesis, the number of target relations using C_1 is always higher than when using C_2. A given relation will be considered as a target one with a threshold higher when using C_2 than when using C_1.

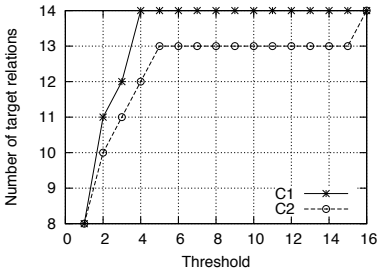


Figure 5: Number of target relations with the alternate S_3 strategy and both counting methods

Considering the strategies, we observed that when using S_3 and C_1, the number of target relations is constant and when using C_2 it takes only two values. Indeed, 14 $\mathcal{P}_1$ relations are not involved in mappings with $\mathcal{P}_2$ relations and 12 of them are not involved even when considering query rewriting. Figure 5 shows the evolution of the number of target relations when the max function replaces the min one in the computation of f($\mathcal{P}_i$:R_j). The number of target relations grows more progressively. All these experiments show that different sets of target relations can be obtained according to strategies, counting methods and the threshold t. These variable elements could be parameters allowing the administrator of a peer to select the set of target relations that are the most convenient for him.

5 Related Work

Currently a lot of works aim at automating generation of mappings. Usually we distinguish terminological, structural and semantic techniques. A survey of these techniques is presented in (3), (4) and (5). In the peer-to-peer setting the alignment problem can be solved in several ways.

Ontology matching techniques suited to data integration systems and developed at first in a centralized mediation context can be used. In Piazza (6), tools and techniques have been developed to simplify and assist mapping creation. A schema matching phase identifies elements in schemas that are to be mapped. Then, automatic techniques and human intervention combine the correspondences to provide a precise mapping. The focus in Piazza was more particularly on automated techniques for schema matching. Various machine learning techniques are composed each learner exploiting a different type of information either in the source schemas or in the data. Furthermore, knowledge from known validated mappings among schemas is reused. Such an approach assumes that the whole ontology of a peer can be known by any other peer. These techniques grounds on the techniques employed in the GLUE system (7) for the integration of heterogeneous data sources.

Other works consider that knowledge of interest in a domain is provided through the network by different ontologies that are autonomous. In (8) a peer stores its data according to same data model and provides a schema and a query language for accessing its data. To be indexed it has to export its schema by means of ontologies that is it has to define mappings between its schema and pre-existent ontologies. Once again all concepts and properties in the ontologies are assumed to be able to be known from all peers.

In (9) the ontology of a $HELIOS$ peer $\mathcal{P}$ is organized as a two-layer ontology where the upper layer describes the knowledge of $\mathcal{P}$ and the lower layer describes the knowledge that $\mathcal{P}$ has of other peers of the network it has interacted with. Each peer can acquire new knowledge or extend its knowledge by querying peers. Special queries, called probe queries, are sent by a peer interested in extending its knowledge of the network. Returned concepts generate new mappings in $\mathcal{P}$.

Our work differs in several ways. First, ontologies in SomeRDFS PDMSs are entirely decentralized. Each peer has its own ontology and ignores the ontology of the others. Second, we benefit from the query processing of the PDMS but queries are only routed to a limited number of peers not to all peers in the network as in (9). Furthermore, queries used in our approach are not specific. They are usual queries submitted to SomeRDFS PDMSs and specific processes are not needed to deal with. Thus, our approach is original because it reuses usual SomeRDFS reasoning mechanisms in order to select elements relevant to be mapped. Once this selection is done, we are going to focus on the matching process itself. In a previous work, we addressed the problem of taxonomy alignment when the structures of the taxonomies are heterogeneous and dissymmetric, one taxonomy being deep whereas the other is flat (10). We are going to explore the suitability of these techniques to our new context in order to propose extensions or adaptations really suited to the SomeRDFS PDMSs setting.

6 Conclusion and Perspectives

In this paper we have presented how SomeRDFS query answering can offer an automated support for discovering new mappings. In particular, we have shown that query answering in a decentralized setting can be used to select elements which are relevant to be matched when the number of elements to be matched is a priori huge and when no peer has a global view of the ontologies in the network. Our approach is based on query answering and filtering criteria.

Currently, we implemented the identification process of target relations and mapping candidates according to the default strategy S_1 and the counting method C_1. We have a running prototype, SpyWhere, providing mapping candidates in SomeRDFS peers. The first experiments show the relevance of our approach but also the need for filtering methods when query answering is used in a P2P setting to support discovering of new mappings. Thus, in a near future, we plan to extend our prototype SpyWhere for handling the other strategies and the second counting method and then we will integrate alignment techniques. That way we will be able to evaluate our approach more completely and to tune the thresholds in the definition function of target relations according to the size of the knowledge and of the mappings of a peer. Future research includes also considering coherence issues due to the integration of discovered mappings among older ones.

References

[1] Adjiman, P., Goasdoué, F., Rousset, M.C.: SomeRDFS in the semantic web. Journal on Data Semantics (2006) p. 158–181

[2] Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.C., Simon, L.: Distributed reasoning in a peer-to-peer setting: Application to the semantic web. JAIR **25** (2006) 269–314

[3] Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. **10**(4) (2001) 334–350

[4] Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Journal of Data Semantics IV (2005) 146–171

[5] Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review **18** (2003) p. 1–31

[6] Tatarinov, I., Ives, Z.G., Madhavan, J., Halevy, A.Y., Suciu, D., Dalvi, N.N., Dong, X., Kadiyska, Y., Miklau, G., Mork, P.: The piazza peer data management project. SIGMOD Record **32**(3) (2003) 47–52

[7] Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., Halevy, A.Y.: Learning to match ontologies on the semantic web. VLDB J. **12**(4) (2003) 303–319

[8] Herschel, S., Heese, R.: Humboldt discoverer: A semantic p2p index for pdms. In: DISWeb'05. (june 2005)

[9] Castano, S., Ferrara, A., Montanelli, S.: H-match: an algorithm for dynamically matching ontologies in peer-based systems. In: SWDB 2003, Berlin, Germany (September 2003)

[10] Reynaud, C., Safar, B.: When usual structural alignment techniques don't apply. In: The ISWC'06 workshop on Ontology matching (OM-06). (2006)