Automatic Generation of P2P Mappings between Sources Schemas

Karima Toumani[1], Hélene Jaudoin[2], and Michel Schneider[1]

[1] LIMOS, Blaise Pascal University
{ktoumani,michel.schneider}@isima.fr
[2] IRISA-ENSSAT
jaudoin@enssat.fr

Abstract. This paper deals with the problem of automatic generation of mappings between data sources schemas, by exploiting existing centralized mappings between sources schemas and a global ontology. We formalize this problem in the settings of description logics and we show that it can be reduced to a problem of rewriting queries using views. We identify two subproblems: the first one is equivalent to a well known problem of computing maximally contained rewritings while the second problem constitutes a new instance of the query rewriting problem in which the goal is to compute minimal rewritings that contain a given query. We distinguish two cases to solve this latter problem: (i) for languages closed under negation, the problem is reduced to the classic problem of rewriting queries using views, and (ii) for languages with the property of structural subsumption, a technique based on hypergraphs is proposed to solve it.

Keywords: Schema mappings, description logics, rewriting queries, hypergraphs.

1 Introduction and Motivation

The growing number of data sources distributed over networks and the relevant exploitation of such data, often heterogeneous, pose the problem of information integration. The use of this information has become a major concern in many areas in the industry, agriculture and commerce [1]. The underlying problems of sharing and integrating information has interested the research community in databases over the two past decades and continue today to be the subject of active investigations [11,10,1,12]. Work in this domain led to the development of techniques and tools to allow an efficient and a transparent access to multiple heterogeneous, distributed and autonomous sources. Two major classes of integration systems were defined : Mediation systems based on the paradigm mediator/wrapper [10] and peer-to-peer systems (P2P) [9]. In the first type of architecture, a central component, called *mediator*, acts as an interface between users and data sources. The mediator is composed of an *global schema* which provides a unified view of data sources. The queries are formulated over a mediation schema (global schema) and are rewritten using the terms of sources schemas. The complexity of mediation systems increases with respect to the number and the capacities of data sources. Although these systems are effective for applications with few data sources, they are poorly adapted to the new context of integration raised by the web, because they are based on a single global schema. The peer-to-peer systems are composed of a set of autonomous data sources (peers) such that each peer is associated with a schema that represents its domain of interest. Each peer's schema constitutes an entry point into the P2P system. In other words, queries are expressed on the local schema of the peer. The resolution of queries in mediation or P2P systems requires the existence of semantic mappings between global schema and data sources schemas in the mediation approach and between peers schemas in the peer to peer approach. From these mappings, the mediator, respectively the peer, analyzes queries and reformulates them in sub-queries executable by sources or peers.

In this article, we address the problem of automatic generation of semantic mappings between schemas of independent data sources. We call them *P2P* mappings compared to centralized mappings stored in the mediator. The main idea is to exploit existing centralized mappings for inferring P2P mappings. Indeed, various efforts have helped develop centralized mappings for example: *(i)* between data sources schemas and a global schema in the context of mediation systems or *(ii)* between web applications and a global ontology of the domain (i.e, ontology that defines the concepts of reference in a particular area). Regarding the latter case, the current development of the web, and particularly the Semantic Web, led to the development of more and more applications that are based on domain global ontologies. For example, the standard **ISCO88** gives a classification of professions while the consortium **RosettaNet** provides standards for trade. Our aim is at exploiting such centralized mappings in order to convert them into P2P mappings between data sources. For example, P2P mappings can be useful to process the distributed queries in P2P systems like SomeWhere [2] and PIAZZA [8,9].

In this paper, we investigate the problem of discovering P2P mappings in the setting of description logics [3]. These logics constitute an important knowledge representation and reasoning formalism. They have been used in different application areas and constitute an important language for the semantic web (e.g.; OWL)[1] of the W3C(World Wide Web Consortium)[2]. In the nutshell, the main contributions of our paper are the following. We formalise the problem of finding P2P mappings from a centralized set of mappings using description logics. Then we propose a technique for automatically generating such mappings. For this purpose, we show that the problem of discovering mappings is reduced to the problem of rewriting queries using views. Specifically, we identify two sub-problems: *(i)* the first one is equivalent to a well known problem of rewriting queries using views, where we search the maximally contained rewritings of queries [7,4,11], and *(ii)* the second one is to find minimal rewritings

[1] http://www.w3.org/2004/OWL/
[2] http://www.w3.org/

which contain a query. We show that the second problem constitutes a new way of rewriting queries problem and we distinguish two cases to solve it. For languages closed under negation, the problem is reduced to the classic problem of rewriting queries using views and for languages with the property of structural subsumption, a technique based on hypergraphs is proposed to solve it.

The paper is organized as follows. Section 2 introduces the basic concepts of description logics. Section 3 shows how the mapping generation problem can be reduced to the problem of rewriting queries using views. Two kind of rewriting problems, namely M_{max} and M_{min} are defined. Solving the problem M_{min} is detailed in the section 4. Finally, section 5 concludes the paper.

2 Preliminaries

In this section, first we give the basic definitions in the description logics, then we describe the notion of rewriting queries using views which is the core operation in our framework.

2.1 Description Logics: An Overview

Description logics (DLs) [3] are a family of knowledge representation formalisms designed for representing and reasoning about terminological knowledge. In DLs, an application domain is represented in terms of *concepts* (unary predicates) denoting sets of individuals, and *roles* (binary predicates) denoting binary relations between individuals. For example, using the conjunction constructor $\sqcap$ and the qualified existential quantification $\exists R.C$, where R is a role and C is a concept, we can describe the concept $Parent$ as follows : $Human \sqcap \exists hasChild.Human$. This concept denotes the set of individuals in the domain who are human and have at least one child.

There are different description logics defined by the set of the constructors they allow. For example, figure 1 give the constructors of two description logics used in our framework : $\mathcal{FL}_0$ and $\mathcal{ALN}$. $\mathcal{FL}_0$ is a simple logic that contains the universal concept (noted $\top$), the conjunction of concepts ($\sqcap$) and ($\forall R.C$). The language $\mathcal{ALN}$ is more expressive because it contains in addition to the constructors of $\mathcal{FL}_0$, the inconsistent concept (noted $\bot$), the negation of atomic concepts (i.e., descriptions formed only by a concept name) and cardinalities constraints on the roles ($\geq n\ R$ and $\leq n\ R$, where n is a positive integer and R is a role name).

The semantics of a concept description is defined by the notion of interpretation. An interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, .^{\mathcal{I}})$ consists of a non-empty set $\triangle^{\mathcal{I}}$, the domain of the interpretation, and an interpretation function $.^{\mathcal{I}}$ that maps each concept name A to a subset of $\triangle^{\mathcal{I}}$ and each role name R to a binary relation $R^{\mathcal{I}}$, subset of $\triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$. For example, any interpretation of a concept $\mathcal{FL}_0$ (resp. $\mathcal{ALN}$) must respect the semantics of this language constructors as given in figure 1.

The notions of satisfiability, subsumption and equivalence between two concepts are defined as follows:

Semantic	$\mathcal{FL}_0$	$\mathcal{ALN}$
$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$	X	X
$\bot^{\mathcal{I}} = \emptyset$		X
$(\neg A)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$		X
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \sqcap D^{\mathcal{I}}$	X	X
$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y : (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$	X	X
$(\leq nR))^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid (x,y) \in R^{\mathcal{I}}\} \leq n\}$		X
$(\geq nR))^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid (x,y) \in R^{\mathcal{I}}\} \geq n\}$		X

Fig. 1. Syntax and Semantic of the concepts constructors for description logics $\mathcal{FL}_0$ and $\mathcal{ALN}$, where n denotes a positive integer, A an atomic concept, C and D concepts, R a role and the symbol $\sharp$ denotes the cardinality of a set

- A concept C is satisfiable iff there is an interpretation I such as $C^I \neq \emptyset$. We say then that I is a valid interpretation or a model for C. The concept C is said inconsistent, noted $C \equiv \bot$ iff C does not admit a model.
- The concept C is subsumed by the concept D, noted $C \sqsubseteq D$, iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ $\forall I$.
- The concept C is equivalent to the concept D, noted $C \equiv D$, iff $C \sqsubseteq D$ and $D \sqsubseteq C$.

The description logics allow to describe an application domain at intentional level using a terminology or Tbox (i.e., a schema or an ontology). Let C be a name of concept and D a concept description. $C \equiv D$ (resp., $C \sqsubseteq D$) is a terminological axiom called *definition* (resp., *primitive specification*). A concept C occurring in the left-hand side of a definition (resp, of a primitive specification) is called *defined concept* (resp, *primitive concepts*). A terminology $\mathcal{T}$ is a finite set of terminological axioms such that no concept name appears more than once in the left-hand side of an terminological axiom.

In this paper, we consider that the terminologies are acyclic; it means that no concept name appears directly or indirectly in its own definition (resp., in its primitive specification). The semantics of a terminology is obtained by extending the notion of interpretation as follows. An interpretation $\mathcal{I}$ satisfied a terminological axiom $C \equiv D$ (resp., $C \sqsubseteq D$) iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ (resp., $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$). An interpretation $\mathcal{I}$ satisfied (or is a model for) a terminology $\mathcal{T}$ iff $\mathcal{I}$ satisfied every terminological axiom in $\mathcal{T}$.

Let C and D be two concepts of a terminology $\mathcal{T}$, the notions of subsumption and equivalence can be extended to the terminology as described below.

- C is subsumed by the concept D according to a terminology $\mathcal{T}$ noted ($C \sqsubseteq_{\mathcal{T}} D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$.
- C and D are equivalents according to a terminology $\mathcal{T}$ noted ($C \equiv_{\mathcal{T}} D$) iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$.

2.2 Rewriting Queries Using Views

The problem of rewriting queries using views is to find a query expression that uses only a set of views and is equivalent to (or maximally contained in) a given query Q. The problem of rewriting queries using views is intensively investigated in the database area [4]. This problem is important for query optimization and for applications such as information integration and data warehousing.

We address here the problem of rewriting queries using views as studied in the context of integration information systems [11]. This problem can be expressed in description logics, since, views definitions can be regarded as a terminology $\mathcal{T}$ and queries as concepts. More precisely, let Q be a query expression described using a subset of defined concepts in the terminology $\mathcal{T}$. A rewriting of Q using $\mathcal{T}$ is a query expression Q' which verifies the two following conditions: (i) Q' is described using only the defined concepts that appear in $\mathcal{T}$ and do not appear in Q, and (ii) $Q' \sqsubseteq_{\mathcal{T}} Q$.

Besides, Q' is called a *maximally-contained rewriting* of Q using $\mathcal{T}$ if there is no rewriting Q'' of Q such that $Q' \sqsubseteq_{\emptyset} Q''$ and $Q'' \sqsubseteq_{\mathcal{T}} Q$. Note that here the subsumption test between Q' and Q'' is achieved according to an empty terminology. This allows to consider, during the test, the concepts that appear in the descriptions of Q' and Q'' as atomic concepts. This allows, for example, to capture the Open World assumption (incomplete views), which is generally admitted in integration systems [11].

3 Generation of P2P Mappings

In this section, we present the problem of P2P mappings generation. This problem consists in the conversion of existing centralized mappings into P2P ones. More precisely, starting from mappings between sources schemas and a global ontology, our aim is at inferring logical relations (e.g., subsumption or equivalence) between concepts of source schemas. We first illustrate below the mapping generation problem on a running example and then we provide a formalization in the DL framework.

Figure 2 describes the inputs of the mapping generation problem: (i) an ontology $\mathcal{O}$, (ii) a set of sources together with their corresponding schemas $\mathcal{S} = \{S_1, S_2, S_3, S_4\}$, and (iii) a set $\mathcal{M}$ of centralized mappings between sources schemas and the ontology $\mathcal{O}$. A centralized mapping is a terminological axiom that establishes a semantic links between the concepts that appear in source schemas and the concepts of a global ontology. For example, the mapping $S_3.USAFlightWithStop \equiv USADeparture \sqcap StopFlight$ in Figure 2 specifies that the concept $USAFlightWithStop$ of the source S_3 has exactly the same meaning (i.e., is equivalent to) as the flights from USA with stops (i.e., the description $USADeparture \sqcap StopFlight$) in the global ontology. We aim at exploiting such centralized mappings to infer mappings between concepts of sources schemas. These latter mappings, called hereafter P2P mappings, are illustrated in Figure 3. The goal of P2P mappings is to establish semantic links between concepts of different source schemas. For example, the P2P mapping $S_1.FlightToEU \sqcap S_3.USAFlightWithStop \equiv S_2.FlightFromUSA$, specifies that the concept $FlightFromUSA$ of the source S_2 has the same meaning as the conjunction of concepts $FlightToEU$ of S_1 and $USAFlightWithStop$ of S_3.

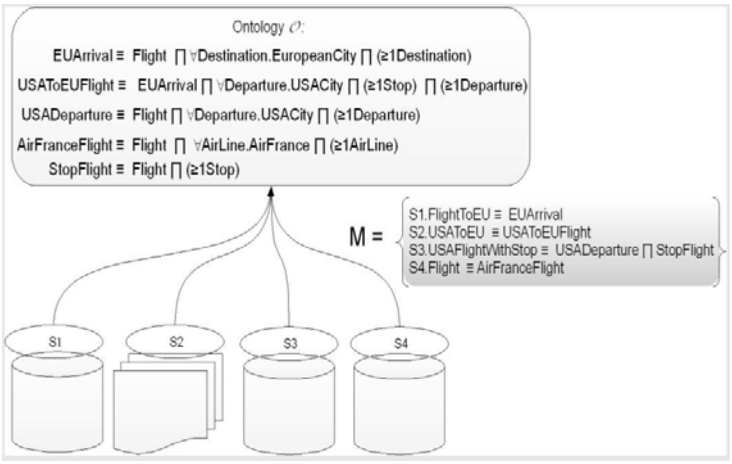
Ontology 𝒪:
EUArrival ≡ Flight ⊓ ∀Destination.EuropeanCity ⊓ (≥1Destination)
USAToEUFlight ≡ EUArrival ⊓ ∀Departure.USACity ⊓ (≥1Stop) ⊓ (≥1Departure)
USADeparture ≡ Flight ⊓ ∀Departure.USACity ⊓ (≥1Departure)
AirFranceFlight ≡ Flight ⊓ ∀AirLine.AirFrance ⊓ (≥1AirLine)
StopFlight ≡ Flight ⊓ (≥1Stop)
M =
S1.FlightToEU ≡ EUArrival
S2.USAToEU ≡ USAToEUFlight
S3.USAFlightWithStop ≡ USADeparture ⊓ StopFlight
S4.Flight ≡ AirFranceFlight
S1
S2
S3
S4

Fig. 2. Example of centralized mappings

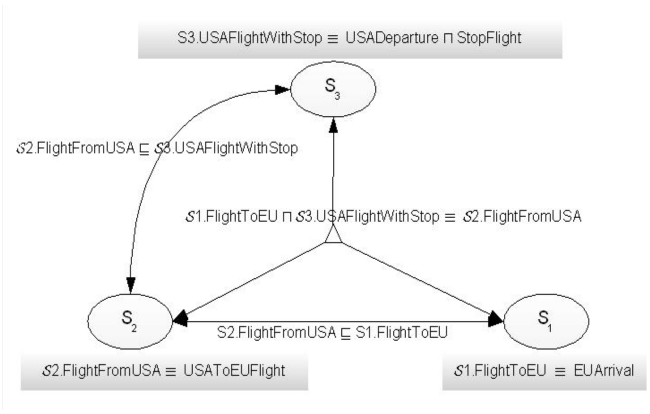
S3.USAFlightWithStop ≡ USADeparture ⊓ StopFlight
S3
𝒮2.FlightFromUSA ⊑ 𝒮3.USAFlightWithStop
𝒮1.FlightToEU ⊓ 𝒮3.USAFlightWithStop ≡ 𝒮2.FlightFromUSA
S2
S1
S2.FlightFromUSA ⊑ S1.FlightToEU
𝒮2.FlightFromUSA ≡ USAToEUFlight
𝒮1.FlightToEU ≡ EUArrival

Fig. 3. Example of P2P mappings

The main goal of this paper is to investigate the technical problems underlying the conversion of centralized mappings into P2P ones.

3.1 Formalization in the DL Setting

We consider the following inputs of our problem:

– a global ontology $\mathcal{O}$, described by mean of a terminology in a given description logic. We denote by D the set of defined concepts in $\mathcal{O}$,
– a set $\mathcal{S} = \{S_1, ..., S_n\}$, where S_i, $i \in \{1, .., n\}$, is a terminology that describes a source schema. We note C_{si} the set of defined concepts in S_i , and

- a set of centralized mappings $\mathcal{M} = \{M_1, ..., M_l\}$ between source schemas and the global ontology. A centralized mapping is given by a terminological axiom $S_i.C_j \equiv D_j$ where $C_j \in C_{si}$, $j \in \{1, n\}$, and D_j is a conjunction of defined concepts from D.

In oder to formally define the mapping generation problem, we first introduce the notions of P2P and non redundant (P2P) mappings.

Definition 1 (P2P mapping). *Let $\mathcal{O}$, $\mathcal{M}$ defined as previously. Let Q_1 and Q_2 be two descriptions defined using concepts from $\bigcup_{i \in \{1,..,n\}} C_{si}$ (i.e., using defined concepts that appear in source schemas). We assume that the set of defined concepts used in Q_1 is disjoint from the one used in Q_2.*

Then, any assertion M' of the form :

- $M' : Q_1 \equiv_{\mathcal{M} \cup \mathcal{O}} Q_2$, *or*
- $M' : Q_1 \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_2$

is a P2P mapping between source schemas.

As an example, the mapping $S_1.FlightToEU \sqcap S_3.USAFlightWithStop \equiv S_2.FlightFromUSA$ is a P2P mapping that establishes a semantic link between the schema S_2 and the schemas S_1 and S_3. Note that, from the previous definition, a P2P mapping establish a link between two disjoint sets of sources schemas (i.e., concepts of a given source schema cannot appear both at right-hand side and left-hand side of a P2P mapping).

It is worth noting that, not all the P2P mappings are interesting to discover. For example, in the Figure 3, the mapping $M_1' : S_2.FlightFromUSA \sqcap S_3.USAFlightWithStop \sqsubseteq S_1.FlightToEU$ is redundant because of the presence of the mapping M_2': $S_2.FlightFromUSA \sqsubseteq S_1.FlightToEU$. Indeed, the mapping M_1' do not convey any additional information w.r.t. to the one already provided by the mapping M_2'. We define below more precisely the notion of non redundant P2P mappings.

Definition 2 (Non redundant P2P mapping). *Let $\mathcal{O}$, $\mathcal{M}$ and S defined as previously. Let Q_1, Q_2 and Q', be descriptions defined using concepts from $\bigcup_{i \in \{1,..,n\}} C_{si}$ (i.e., using defined concepts that appear in source schemas).*

A P2P mapping M': $Q_1 \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_2$, is non redundant w.r.t. ($\mathcal{O}$, $\mathcal{M}$ $\mathcal{S}$) if and only if :

i) $\nexists Q' \mid Q_1 \sqsubseteq_{\emptyset} Q'$ *and* $Q' \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_2$ *and*
ii) $\nexists Q' \mid Q' \sqsubseteq_{\emptyset} Q_2$ *and* $Q_1 \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q'$.

For practical purposes (e.g., query processing in a P2P system), only non redundant mappings are relevant to consider. Continuing with the running example, the presence of the mapping M_2' makes the mapping M_1' useless for processing queries. For instance, consider a query $Q \equiv S_1.FlightToEU$, posed on the source S_1, and that looks for flights to Europe. Using the mapping M_1', the query Q can be rewritten into a query $Q' \equiv S_2.FlightFromUSA \sqcap S_3.USAFlightWithStop$

that computes answers from the sources S_2 and S_3. In the same way, using the mapping M'_2, the query Q can be rewritten into a query $Q'' \equiv S_2.FlightFromUSA$ that compute the answers from the sources S_2. However, we can observe that the set of answers returned by Q' is a subset of Q'' answers. Therefore, in this example, the mapping M'_2 is sufficient to answer the query Q and hence, in the presence of M'_2, the redundant mapping M'_1 is useless for computing Q answers.

In the remaining of this paper, we focus our attention on the problem of computing non redundant P2P mappings from a set of existing centralized mappings. We show in the next section how instances of such a problem can be reduced to different instances of the problem of query rewriting using views.

3.2 From Mappings Discovery to Query Rewriting

We consider in this section the problem of computing non redundant P2P mappings. This problem, noted *ConvertMapping* is defined precisely below.

Problem 1 (*ConvertMapping*$(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$). *Let $\mathcal{O}$ be a global ontology, $\mathcal{M}$ a set of centralized mappings and $\mathcal{S} = \{S_1, \ldots, S_n\}$ a set of sources schemas. The input description Q_{input} is a concept described using the defined concepts from $\bigcup_{i \in \{1,n\}} C_{si}$ (i.e., the description of Q_{input} uses only defined concepts that appear in source schemas). The problem is then to compute all the non redundant mappings w.r.t. $(\mathcal{O}, \mathcal{M}, \mathcal{S})$ of the following forms: $Q_{input} \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q$, $Q \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_{input}$ and $Q_{input} \equiv_{\mathcal{M} \cup \mathcal{O}} Q$.*

Hence, the problem is to compute all the non redundant mappings that involve the input description Q_{input} either in their left-hand side or in their right-hand side. It is worth noting that the definition of the *ConvertMapping* problem assumes that the input description Q_{input} is given *a priori* (e.g., provided by a user or selected from the defined concepts of the sources that appear in centralized mappings). Automatically computing relevant *input descriptions* w.r.t. some QoS criteria is an interesting question that is left open in this paper.

Note that, to solve *ConvertMapping*$(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ it is sufficient to focus only on subsumption based mappings (i.e., $Q_{input} \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q$ and $Q \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_{input}$) since by definition, their discovery enables to find equivalence mappings when they exist. In the sequel, we decompose a *ConvertMapping* problem into two subproblems: the first one, noted $M_{max}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$, concerns the discovery of P2P mappings of the form $Q \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_{input}$, and the second one, noted $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$, concerns the discovery of P2P mappings of the form $Q_{input} \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q$.

As stated by the following lemma, the problem $M_{max}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ can be straightforwardly reduced to the classical problem of query rewriting using views.

Lemma 1. *Let $\mathcal{O}$, $\mathcal{S}$, $\mathcal{M}$ and Q_{input} defined as previously. Then:*
$Q \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q_{input}$ is a non redundant mapping of Q_{input} w.r.t. $(\mathcal{O}, \mathcal{S}, \mathcal{M})$ iff Q is a maximally contained rewriting of Q_{input} using $\mathcal{M} \cup \mathcal{O}$.

The problem of computing maximally-contained rewriting of a query has been studied in the literature for different types of languages [4,11,7]. In the context of description logics, this problem has been shown to be decidable even for very expressive languages such as $\mathcal{ALCNR}$ [4].

The next section is devoted to the problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$.

4 The Problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$

In this section, we formalize the problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ as a new form of query rewriting where the aim is to compute *minimal* rewritings of Q_{input} using $\mathcal{M} \cup \mathcal{O}$. More precisely, we consider two families of description logics: (i) DLs closed under negation, and (ii) DLs with the property of structural subsumption.

Case of DL Closed under Negation. A description logic $\mathcal{L}$ is said to be closed under negation if for any description C in $\mathcal{L}$, $\neg C$ is also a description in $\mathcal{L}$. The following lemma says that in the particular case of such logics, the problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ can be reduced to a problem of computing maximally contained rewritings.

Lemma 2. *Let $\mathcal{O}$, $\mathcal{S}$, $\mathcal{M}$ and Q_{input} defined as previously. Then:*

$Q_{input} \sqsubseteq_{\mathcal{M}\cup\mathcal{O}} Q$ is a non redundant mapping of Q_{input} w.r.t. $(\mathcal{O}, \mathcal{S}, \mathcal{M})$ iff $\neg Q$ is a maximally contained rewriting of $\neg Q_{input}$ using $\mathcal{M} \cup \mathcal{O}$.

The demonstration of this lemma is based on the following axiom : $Q \sqsubseteq_{min} Q'$ is equivalent to $\neg Q' \sqsubseteq_{max} \neg Q$.

A good candidate logic for this case is the language $\mathcal{ALCNR}$ which displays two interesting properties: it is closed under negation, and the problem of query rewriting using views is decidable for this language.

Case of DL with Structural Subsumption. We consider now a case of another family of logics, namely DL with the structural subsumption. We introduce first some basic notions that enable to define precisely such logics. Then, we show how to solve the $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ problem in this setting using hypergraph techniques.

Definition 3 (Reduced clause form). *Let $\mathcal{L}$ be a description logic. A clause in $\mathcal{L}$ is a description A with the following property:*

$$(A \equiv B \sqcap A') \Rightarrow (B \equiv \top) \wedge (A' \equiv A) \vee (A' \equiv \top) \wedge (B \equiv A).$$

Every conjunction $A_1 \sqcap ... \sqcap A_n$ of clauses can be represented by the clause set $\hat{A} = \{A_1, ..., A_n\}$.

$\hat{A} = \{A_1, ..., A_n\}$ is called a reduced clause set if :

- *either $n = 1$.*
- *or no clause subsumes the conjunction of the other clauses, i.e, : $\forall i \mid i \in \{1, ..., n\}, A_i \not\sqsupseteq (\hat{A} \backslash A_i)$.*

The set $\hat{A}$ is then called a reduced clause form (RCF).

Consider for example a description $D \equiv A \sqcap B \sqcap \forall R.C$, where A, B, and C are atomic concepts. Then, an RFC of D is $\hat{D} = \{A, B, \forall R.C\}$. Let us now introduce the notion of structural subsumption as defined in [13].

Definition 4 (Structural subsumption). *The subsumption relation in a description logic $\mathcal{L}$ is said structural iff for any description $A \equiv A_1 \sqcap ... \sqcap A_n$ and any description $B \equiv B_1 \sqcap ... \sqcap B_m$ in $\mathcal{L}$ which is given by its RCF, the following holds:*

$B \sqsubseteq A \Leftrightarrow \forall A_j \in \hat{A}, \exists B_i \in \hat{B} \mid B_i \sqsubseteq A_j$

In other words , if $B \sqsubseteq A$ then $\hat{A} \subseteq \hat{B}$ ($\{A_j\} \subseteq \{B_i\}$).

We consider now the problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ in the framework of DLs with structural subsumption. The following lemma provides a characterization of the solutions in terms of clause sets.

Lemma 3. *(Characterization of the minimal rewritings subsuming a query)*

Let $\mathcal{O}$, $\mathcal{S}$, $\mathcal{M}$ and Q_{input} defined as previously but using a description logic with structural subsumption. Then:

$Q_{input} \sqsubseteq_{\mathcal{M}\cup\mathcal{O}} Q$ is a non redundant mapping of Q_{input} w.r.t. $(\mathcal{O}, \mathcal{S}, \mathcal{M})$ iff $\hat{Q}$ is a maximal set of clauses such that $\hat{Q} \subseteq \widehat{Q_{input}}$.

The demonstration of this lemma is based on the structural subsumption propriety. Indeed, with the following descriptions given by their RCFs, if $A \sqcap B \sqcap C \sqsubseteq$ $\mathbf{A} \sqcap \mathbf{B}$ and $A \sqcap B \sqcap C \sqsubseteq \mathbf{A}$, we note here that $\mathbf{A} \sqcap \mathbf{B} \sqsubseteq \mathbf{A}$. So we deduce that the description with the maximal set of clauses is the minimal.

Therefore, to solve a $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ in the context of DLs with structural subsumption, one has to look for descriptions Q such that their RFCs are made of maximal, with respect to set inclusion, sets of clauses that are included in the RFC of the input description. As shown below, this problem can be reduced to a problem of computing minimal transversals of a given hypergraph.

We recall first the definitions of hypergraphs and minimal transversals.

Definition 5. *(Hypergraph and transversals [6]) A hypergraph $\mathcal{H}$ is a pair (Σ, Γ) of a finite set $\Sigma = \{V_1, ..., V_n\}$ and a set $\Gamma = \{\epsilon_1, ..., \epsilon_n\}$ of subsets of Σ. The elements of Σ are called vertices, and the elements of Γ are called edges.*

A set $\mathcal{R} \subseteq \Sigma$ is a transversal of $\mathcal{H}$ if for each $\epsilon \in \Gamma$, $\mathcal{R} \cap \epsilon \neq \emptyset$. A transversal $\mathcal{R}$ is minimal if no proper subset $\mathcal{R}'$ of $\mathcal{R}$ is a transversal. The set of the minimal transversals of an hypergraph $\mathcal{H}$ is noted Tr($\mathcal{H}$).

We show now how to map a $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ problem into a hypergraph based framework. Consider an instance of the problem $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ where all the descriptions are expressed using a description logic with structural subsumption. We assume that concept descriptions are represented by their RCFs. We note by $N_{\mathcal{M}}$ the set of defined concepts of $\mathcal{M}$ that do not belong to one of the source schemas used in Q_{input}. Then, we build the associated hypergraph $\mathcal{H}_{\mathcal{M}_{min}} = (\Sigma, \Gamma)$ as follows.

- Each concept $C_i \in N_{\mathcal{M}}$ such that $\hat{C_i} \subseteq \hat{Q_{input}}$, is associated with a vertex V_{C_i} in the hypergraph $\mathcal{H}_{\mathcal{M}_{min}}$. So $\Sigma = \{V_{C_i}, i \in \{1,..,l\}\}$.
- Each clause $A_j \in \hat{Q_{input}}, j \in \{1,..,k\}$ is associated with an edge ϵ_{A_j} in $\mathcal{H}_{\mathcal{M}_{min}}$ such that $\epsilon_{A_j} = \{V_{C_i} \mid A_j \in \{\hat{C_i} \cap \hat{Q_{input}}\}$.

Consider for example a $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ problem where $\mathcal{O}$, $\mathcal{S}$, $\mathcal{M}$ are depicted by the Figure 2 and $Q_{input} \equiv S_2.FlightFromUSA$. The associated hypergraph $\mathcal{H}_{\mathcal{M}_{min}} = (\Sigma, \Gamma)$ is built as follows : The set of vertices is $\Sigma = \{V_{S_1 FlightToEU}, V_{S_3.USAFligntWithStop}\}$ and the set of edges is :

$\Gamma = \{\epsilon_{(Flight)}, \epsilon_{(\forall Departure.USACity)}, \epsilon_{(\geq 1 Departure)}, \epsilon_{(\forall Destination.EuropeanCity)}, \epsilon_{(\geq 1 Destination)}, \epsilon_{(\geq 1 Stop)}\}$.

Lemma 4. *Let $M_{min}(\mathcal{O}, \mathcal{M}, Q)$ be a P2P generation problem expressed in the context of a DL with structural subsumption. Then, Q a conjunction of $C_i, i \in \{1,..,n\}$ is a solution of $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$ iff $\mathcal{R}_Q = \{V_{C_i}, i \in \{1,..,n\}$ is a minimal transversal of the hypergraph $\mathcal{H}_{\mathcal{M}_{min}}$.*

Proof :$Q \equiv \sqcap_{i=1}^{n} C_i, Q_{input} \equiv \sqcap_{j=1}^{m} A_j$. Q is a solution of $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$. $\forall A_j \in \hat{Q} \cap \hat{Q_{input}}, \exists C_i \mid A_j \in \hat{C_i} \cap \hat{Q_{input}} \Leftrightarrow \forall \epsilon_{A_j} \in \mathcal{H}_{\mathcal{M}_{min}}, \exists V_{C_i} \in \mathcal{R}_Q \mid V_{C_i} \in \epsilon_{A_j} \Leftrightarrow \mathcal{R}_Q$ is a transversal of the hypergraph $\mathcal{H}_{\mathcal{M}_{min}}$. Q is a minimal rewriting of $Q_{input} \Leftrightarrow \nexists Q' \mid Q_{input} \sqsubseteq_{\mathcal{M} \cup \mathcal{O}} Q' \sqsubseteq_{\emptyset} Q \Leftrightarrow \mathcal{R}_Q$ is a minimal transversal of $\mathcal{H}_{\mathcal{M}_{min}}$

This lemma provides a practical method to solve $M_{min}(\mathcal{O}, \mathcal{S}, \mathcal{M}, Q_{input})$. Indeed, it enables to reuse and adapt known techniques and algorithms for computing minimal transversals [5] to our context.

Continuing with the example, considering the hypergraph $\mathcal{H}_{\mathcal{M}_{min}}$, the minimal transversals are $\{V_{S_1.FlightToEU}, V_{S_3.USAFligntWithStop}\}$. So, in this case we have only one minimal rewriting that subsume Q_{input} which is $Q \equiv S_1.FlightToEU \sqcap S_3.USAFligntWithStopS$.

5 Conclusion

In this paper, we considered the problem of the automatic generation of semantic P2P mappings between autonomous data source schemas. We formalized this problem using description logics and we showed that it can be reduced to a problem of rewriting queries using views. We have developed a prototype that implements our approach. The algorithms of query rewriting and the computation of the minimal transversals of the hypergraph are implemented using the JAVA language and it is based on the OWL language. The prototype is composed of three modules : Parser, GUI modules and Hypergraph. This latter module is devoted to the computation of minimal rewritings. Our future work will be devoted to exploration of other family of logics for which the investigated problem can be solved. We will also investigate the possibly of automatic generation of *input description*. Finally, quantitative experimentation and algorithmic optimization constitute also interesting future research directions.

References

1. Abiteboul, S., et al.: The lowell database research self-assessment (2003)
2. Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.-C., Simon, L.: SomeWhere in the semantic web. In: Fages, F., Soliman, S. (eds.) PPSWR 2005. LNCS, vol. 3703, pp. 1–16. Springer, Heidelberg (2005)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. University Press, Cambridge (2003)
4. Beeri, C., Levy, A.Y., Rousset, M.C.: Rewriting queries using views in description logics. In: PODS 1997, pp. 99–108. ACM Press, New York (1997)
5. Berge, C.: Hypergraphs (1989)
6. Eiter, T., Gottlob, G.: Identifying the minimal transversals of an hypergraph and related problems. SIAM of Computing 24(6), 1278–1304 (1995)
7. Goasdoué, F., Rousset, M.C.: Answering queries using views: A KRDB perspective for the semantic web. ACM Transactions on Internet Technology 4(3), 255–288 (2004)
8. Halevy, A.Y., Ives, Z.G., Suciu, D., Tatarinov, I.: Schema mediation in peer data management systems. In: ICDE, pp. 505–516 (2003)
9. Halevy, A.Y., Ives, Z.G., Madhavan, J., Mork, P., Suciu, D., Tatarinov, I.: The piazza peer data management system. IEEE Transactions on Knowledge and Data Engineering 16(7), 787–798 (2004)
10. Lenzerini, M.: Data integration: a theoretical perspective. In: PODS 2002, pp. 233–246. ACM Press, New York (2002)
11. Levy, A.Y.: Answering queries using views: A survey. The VLDB Journal 10, 270–294 (2001)
12. Sarma, A.D., Dong, X., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 861–874. ACM, New York (2008)
13. Teege, G.: Making the difference: A subtraction operation for description logics. In: Doyle, J., Sandewall, E., Torasso, P. (eds.) Principles of Knowledge Representation and Reasoning: Proc. of the 4th International Conference (KR 1994), pp. 540–550. Morgan Kaufmann Publishers, San Francisco (1994)