A Mapping System for the Integration of OWL-DL Ontologies

Peter Haase
Institute AIFB
University of Karlsruhe, Germany
haase@aifb.uni-karlsruhe.de

Boris Motik
FZI Karlsruhe, Germany
motik@fzi.de

ABSTRACT

To enable interoperability between applications in distributed information systems based on heterogeneous ontologies, it is necessary to formally define the notion of a mapping between ontologies. In this paper, we define a mapping system for OWL-DL ontologies, where mappings are expressed as correspondences between conjunctive queries over ontologies. As query answering within such a general mapping system is undecidable, we identify a decidable fragment of the mapping system, which corresponds to OWL-DL extended with DL-safe rules. We further show how the mapping system can be applied for the task of ontology integration and present a query answering algorithm.

Categories and Subject Descriptors

H.2.5 [**Database Management**]: Heterogeneous Databases; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods

General Terms

Algorithms, Theory

Keywords

Ontology Mapping, Ontology Integration, Description Logics

1. INTRODUCTION

To enable interoperability between applications in large distributed information systems based on heterogeneous ontologies, it is necessary to formally define the notion of a mapping between ontologies. An important step in this direction has been taken by adopting the Web Ontology Language (OWL) as a W3C recommendation for building ontologies in the Semantic Web. OWL already provides support to express simple mappings between ontology elements. However, this support is too limited for many practical purposes, since the mapping primitives merely include subsumption and equivalence between classes, properties and individuals.

In this paper, we follow the general framework of [18] to formalize the notion of a mapping system for OWL-DL ontologies, where mappings are expressed as correspondences between queries over ontologies. It is easy to see that query answering within such a mapping system is undecidable. To obtain an alternative more suitable for practical applications, we introduce restrictions required to attain decidability. These restricted, but still very expressive mappings, can be expressed in OWL-DL extended with the so-called *DL-safe* subset of the Semantic Web Rule Language (SWRL) [19]. Furthermore, we note that these restrictions can be relaxed for so-called *tree-like mappings* using query roll-up techniques from [14]. We demonstrate the expressiveness of the mapping system with a practical example.

While possible applications of the mapping system are manifold, including for example data transformation and data exchange [8], in the paper we show how our proposed mapping system can be applied for the task of *ontology integration*, which addresses the problem of integrating a set of local ontologies. Queries are expressed against a global, integrated ontology, which provides a unified view of the local ontologies. Query answering in the ontology integration system (using conjunctive queries) is based on a novel technique for reducing description logic knowledge bases to disjunctive datalog programs [16, 15].

IHIS'05, November 4, 2005, Bremen, Germany.

2. PRELIMINARIES

The OWL ontology language is based on a family of description logics languages. In particular, OWL-DL is a syntactic variant of the $\mathcal{SHOIN}(\mathbf{D})$ description logic [12].

Hence, although several XML and RDF-based syntaxes for OWL-DL exist, in this paper we use the traditional description logic notation since it is more compact. For the correspondence between this notation and various OWL-DL syntaxes, see [12].

2.1 $\mathcal{SHOIN}(\mathbf{D})$ Description Logic

$\mathcal{SHOIN}(\mathbf{D})$ is a very expressive description logic that provides full negation, disjunction and a restricted form of universal form of existential quantification. $\mathcal{SHOIN}(\mathbf{D})$ further supports reasoning with concrete datatypes, such as strings or integers. Instead of axiomatizing concrete datatypes in logic, $\mathcal{SHOIN}(\mathbf{D})$ employs an approach similar to [1], where the properties of concrete datatypes are encapsulated

in so-called *concrete domains*. A *concrete domain* is a pair $(\triangle_{\mathbf{D}}, \Phi_{\mathbf{D}})$, where $\triangle_{\mathbf{D}}$ is the interpretation domain, and $\Phi_{\mathbf{D}}$ is a set of concrete domain predicates that come with an arity n and a predefined interpretation $d^{\mathbf{D}} \subseteq \triangle_{\mathbf{D}}^n$. An *admissible* concrete domain $\mathbf{D}$ is equipped with a decision procedure for the satisfiability of finite conjunctions over concrete domain predicates. Satisfiability checking of admissible concrete domains can successfully be combined with logical reasoning for many description logics.

The main expressive means of description logics are so called concept descriptions, which describe sets of individuals or objects.

Let N_C be a set of *atomic concept names*, N_{R_a} and N_{R_c} sets of *abstract and concrete role names*, respectively, and N_{I_a} and N_{I_c} sets of *abstract and concrete individuals*, respectively. An *abstract role* is an abstract role name or the inverse S^- of an abstract role name S (concrete roles do not have inverses). Finally, let $\mathbf{D}$ be an admissible concrete domain.

An *RBox* $KB_{\mathcal{R}}$ is a finite set of transitivity axioms $\mathsf{Trans}(R)$, and role inclusion axioms of the form $R \sqsubseteq S$ and $T \sqsubseteq U$, where R and S are abstract roles, and T and U are concrete roles. The reflexive-transitive closure of the role inclusion relationship is denoted with $\sqsubseteq^*$. A role not having transitive subroles (w.r.t. $\sqsubseteq^*$, for a full definition see [13]) is called a *simple* role.

The set of $\mathcal{SHOIN}(\mathbf{D})$ *concepts* is defined by the following syntactic rules, where A is an atomic concept, R is an abstract role, S is an abstract simple role, $T_{(i)}$ are concrete roles, d is a concrete domain predicate, a_i and c_i are abstract and concrete individuals, respectively, and n is a non-negative integer:

$$\begin{aligned} C &\rightarrow A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists R.C \mid \forall R.C \mid \\ &\quad \geq n\, S \mid \leq n\, S \mid \{a_1, \ldots, a_n\} \mid \geq n\, T \mid \leq n\, T \mid \\ &\quad \exists T_1, \ldots, T_n.D \mid \forall T_1, \ldots, T_n.D \\ D &\rightarrow d \mid \{c_1, \ldots, c_n\} \end{aligned}$$

A *TBox* $KB_{\mathcal{T}}$ is a finite set of concept inclusion axioms $C \sqsubseteq D$, for C and D concepts; an *ABox* $KB_{\mathcal{A}}$ is a finite set of concept and role assertions and individual (in)equalities $C(a)$, $R(a,b)$, $a \approx b$ and $a \not\approx b$, respectively. A $\mathcal{SHOIN}(\mathbf{D})$ *knowledge base* KB is a triple $(KB_{\mathcal{T}}, KB_{\mathcal{R}}, KB_{\mathcal{A}})$.

The semantics of a $\mathcal{SHOIN}(\mathbf{D})$ knowledge base KB is given by the mapping π which transforms KB axioms into a first-order formula, as shown in Table 1. Each atomic concept is mapped into a unary predicate and each abstract role is mapped into a binary predicate.

The $\mathcal{SHIQ}(\mathbf{D})$ description logic is obtained from $\mathcal{SHOIN}(\mathbf{D})$ by disallowing nominal concepts of the form $\{a_1, \ldots, a_n\}$ and $\{c_1, \ldots, c_n\}$, and by allowing qualified number restrictions of the form $\geq n\, S.C$ and $\leq n\, S.C$, for C a $\mathcal{SHIQ}(\mathbf{D})$ concept and S a simple role.

2.2 Rules and Conjunctive Queries

We now introduce the notion of conjunctive queries over a $\mathcal{SHOIN}(\mathbf{D})$ knowledge base KB. This notion is used in Section 3 to define the mapping formalism.

Definition 1 (Conjunctive Queries) *Let KB be a $\mathcal{SHOIN}(\mathbf{D})$ knowledge base, and let N_P be a set of predicate symbols, such that all $\mathcal{SHOIN}(\mathbf{D})$ concepts and all abstract and concrete roles are in N_P. An* atom *has the form $P(s_1, \ldots, s_n)$, often denoted as $P(\mathbf{s})$, where $P \in N_P$, and s_i are either variables or individuals from KB. An atom is called* ground atom*, if it is variable-free. An atom is called a* DL-atom *if P is a $\mathcal{SHOIN}(\mathbf{D})$ concept, or an abstract or a concrete role.*

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ be sets of distinguished *and* non-distinguished *variables, denoted as $\mathbf{x}$ and $\mathbf{y}$, respectively. A* conjunctive query *over KB, written as $Q(\mathbf{x}, \mathbf{y})$, is a conjunction of atoms $\bigwedge P_i(\mathbf{s_i})$, where all $\mathbf{s_i}$ together exactly contain $\mathbf{x}$ and $\mathbf{y}$.*

A conjunctive query $Q(\mathbf{x}, \mathbf{y})$ is DL-safe *if each variable occurring in a DL-atom also occurs in a non-DL-atom in $Q(\mathbf{x}, \mathbf{y})$.*

We extend the operator π from Section 2.1 to translate $Q(\mathbf{x}, \mathbf{y})$ into a first-order formula with free variables $\mathbf{x}$ as follows:

$$\pi(Q(\mathbf{x}, \mathbf{y})) = \exists \mathbf{y} : \bigwedge \pi(P_i(\mathbf{s_i}))$$

For $Q_1(\mathbf{x}, \mathbf{y_1})$ and $Q_2(\mathbf{x}, \mathbf{y_2})$ conjunctive queries, a query containment *axiom $Q_2(\mathbf{x}, \mathbf{y_2}) \sqsubseteq Q_1(\mathbf{x}, \mathbf{y_1})$ has the following semantics:*

$$\begin{aligned} &\pi(Q_2(\mathbf{x}, \mathbf{y_2}) \sqsubseteq Q_1(\mathbf{x}, \mathbf{y_1})) = \\ &\quad \forall \mathbf{x} : \pi(Q_1(\mathbf{x}, \mathbf{y_1})) \leftarrow \pi(Q_2(\mathbf{x}, \mathbf{y_2})) \end{aligned}$$

The main inferences for conjunctive queries are:

- Query answering. *An* answer *of a conjunctive query $Q(\mathbf{x}, \mathbf{y})$ w.r.t. KB is an assignment θ of individuals to distinguished variables, such that $\pi(KB) \models \pi(Q(\mathbf{x}\theta, \mathbf{y}))$.*
- Checking query containment. *A query $Q_2(\mathbf{x}, \mathbf{y_2})$ is contained in a query $Q_1(\mathbf{x}, \mathbf{y_1})$ w.r.t. KB, if $\pi(KB) \models \pi(Q_2(\mathbf{x}, \mathbf{y_2}) \sqsubseteq Q_1(\mathbf{x}, \mathbf{y_1}))$.*

We now define the notion of rules and combined knowledge bases extended with rules.

Definition 2 (Rules) *A* rule *over a $\mathcal{SHOIN}(\mathbf{D})$ knowledge base KB has the form $H \leftarrow Q(\mathbf{x}, \mathbf{y})$ where H is an atom and $Q(\mathbf{x}, \mathbf{y})$ a query over KB. As usual, we assume rules to be* safe*, i.e. that each variable from H occurs in $\mathbf{x}$ as well. A rule is* DL-safe *if and only if $Q(\mathbf{x}, \mathbf{y})$ is DL-safe. We extend the operator π to translate rules into first-order formulas as follows:*

$$\pi(H \leftarrow Q(\mathbf{x}, \mathbf{y})) = \forall \mathbf{x} : \pi(H) \leftarrow \pi(Q(\mathbf{x}, \mathbf{y}))$$

A program *P is a finite set of rules; P is DL-safe if all rules are DL-safe. A* combined knowledge base *is a pair (KB, P); we define $\pi((KB, P)) = \pi(KB) \cup \pi(P)$. The main inference in (KB, P) is* query answering*, i.e. deciding whether $\pi((KB, P)) \models A$ for a ground atom A.*

To simplify the presentation, in the above definition we assume that H is a single atom, and not a conjunction of atoms. This is without loss of generality: it is well-known that a rule of the form $A_1 \wedge \ldots \wedge A_n \leftarrow B_1 \wedge \ldots \wedge B_m$ is equivalent to the set of rules $A_i \leftarrow B_1 \wedge \ldots \wedge B_m$, for $1 \leq i \leq n$.

Notice that without DL-safety, the above definition matches that of the SWRL rules [11]. Intuitively, DL-safety restricts the applicability of a query or a rule only to individuals explicitly named in a knowledge base KB. To automatically

Table 1: Translation of $\mathcal{SHOIN}(\mathbf{D})$ into FOL

Mapping Concepts to FOL	
$\pi_y(\top, X)=\top$	$\pi_y(\bot, X)=\bot$
$\pi_y(A, X)=A(X)$	$\pi_y(\neg C, X)=\neg\pi_y(C, X)$
$\pi_y(C \sqcap D, X)=\pi_y(C, X) \wedge \pi_y(D, X)$	$\pi_y(C \sqcup D, X)=\pi_y(C, X) \vee \pi_y(D, X)$
$\pi_y(\forall R.C, X)=\forall y : R(X, y) \rightarrow \pi_x(C, y)$	$\pi_y(\exists R.C, X)=\exists y : R(X, y) \wedge \pi_x(C, y)$
$\pi_y(\{a_1 \ldots, a_n\}, X)=X \approx a_1 \vee \ldots \vee X \approx a_n$	$\pi_y(\{c_1^{\mathsf{c}}, \ldots, c_n^{\mathsf{c}}\}, X)=X \approx_{\mathbf{D}} c_1^{\mathsf{c}} \vee \ldots \vee X \approx_{\mathbf{D}} c_n^{\mathsf{c}}$
$\pi_y(d, X_1, \ldots, X_m)= d(X_1, \ldots, X_m)$	
$\pi_y(\leq n\, R.C, X)= \forall y_1, \ldots, y_{n+1} : \bigwedge R(X, y_i) \wedge \bigwedge \pi_x(C, y_i) \rightarrow \bigvee y_i \approx y_j$	
$\pi_y(\geq n\, R.C, X)= \exists y_1, \ldots, y_n : \bigwedge R(X, y_i) \wedge \bigwedge \pi_x(C, y_i) \wedge \bigwedge y_i \not\approx y_j$	
$\pi_y(\forall T_1, \ldots, T_m.D, X)= \forall y_1^{\mathsf{c}}, \ldots, y_m^{\mathsf{c}} : \bigwedge T_i(X, y_i^{\mathsf{c}}) \rightarrow \pi_x(D, y_1^{\mathsf{c}}, \ldots, y_m^{\mathsf{c}})$	
$\pi_y(\exists T_1, \ldots, T_m.D, X)= \exists y_1^{\mathsf{c}}, \ldots, y_m^{\mathsf{c}} : \bigwedge T_i(X, y_i^{\mathsf{c}}) \wedge \pi_x(D, y_1^{\mathsf{c}}, \ldots, y_m^{\mathsf{c}})$	
$\pi_y(\leq n\, T, X)= \forall y_1^{\mathsf{c}}, \ldots, y_{n+1}^{\mathsf{c}} : \bigwedge T(X, y_i^{\mathsf{c}}) \rightarrow \bigvee y_i^{\mathsf{c}} \approx_{\mathbf{D}} y_j^{\mathsf{c}}$	
$\pi_y(\geq n\, T, X)= \exists y_1^{\mathsf{c}}, \ldots, y_n^{\mathsf{c}} : \bigwedge T(X, y_i^{\mathsf{c}}) \wedge \bigwedge y_i^{\mathsf{c}} \not\approx_{\mathbf{D}} y_j^{\mathsf{c}}$	
Mapping Axioms and KB to FOL	
$\pi(C \sqsubseteq D)= \forall x : \pi_y(C, x) \rightarrow \pi_y(D, x)$	
$\pi(R \sqsubseteq S)= \forall x, y : R(x, y) \rightarrow S(x, y)$	
$\pi(\mathsf{Trans}(R))= \forall x, y, z : R(x, y) \wedge R(y, z) \rightarrow R(x, z)$	
$\pi(C(a))= \pi_y(C, a)$	
$\pi(R(a, b))= R(a, b)$	
$\pi(a^{(\mathsf{c})} \circ b^{(\mathsf{c})})=a \circ_{(\mathbf{D})} b$ for $\circ \in \{\approx, \not\approx\}$	
$\pi(KB)= \bigwedge_{R \in N_R} \forall x, y : R(x, y) \leftrightarrow R^-(y, x) \wedge \bigwedge_{\alpha \in KB_{\mathcal{R}} \cup KB_{\mathcal{T}} \cup KB_{\mathcal{A}}} \pi(\alpha)$	

X is a meta variable and is substituted with the actual variable. π_x is obtained from π_y by simultaneously substituting all $y_{(i)}$ with $x_{(i)}$ and π_y with π_x, and vice versa.

convert a non-DL-safe query into a DL-safe one, we assume a special non-DL predicate $\mathcal{O}$ such that, for each individual α occurring in KB, it contains a fact $\mathcal{O}(\alpha)$. Then, a non-DL-safe conjunctive query $Q(\mathbf{x}, \mathbf{y})$ can be easily converted into a DL-safe query by appending to it an atom of the form $\mathcal{O}(z)$, for each variable z occurring only in a DL-atom of $Q(\mathbf{x}, \mathbf{y})$. For an in-depth discussion about the consequences that this transformation has on the semantics, please refer to [19].

2.3 Reducing DLs to Disjunctive Datalog

The algorithm we present in Section 4 is based on the correspondence between description logics and disjunctive datalog from [16]. Given a $\mathcal{SHIQ}(\mathbf{D})$ knowledge base KB, this algorithm produces a positive disjunctive datalog program $\mathsf{DD}(KB)$ which entails exactly the same set of ground facts as KB, i.e. $KB \models A$ if and only if $\mathsf{DD}(KB) \models A$, for A a ground fact. Thus, query answering in KB is reduced to query answering in $\mathsf{DD}(KB)$, which can be performed efficiently using the techniques of (disjunctive) deductive databases. For example, the magic sets transformation [2] or various statistics-based join-order optimizations can be applied to $\mathsf{DD}(KB)$ to optimize query answering.

Due to some technical particularities, this algorithm requires KB to be a $\mathcal{SHIQ}(\mathbf{D})$ knowledge base; all OWL-DL constructs apart from nominals are supported. It is thus not possible to define a class completely with an enumeration of all of its class members. Another restriction is that the ground fact A is not allowed to contain complex roles (i.e. roles with transitive subroles). This is due to the approach used to handle transitivity axioms; for details, refer to [16]. However, it is still possible to axiomatize transitivity using DL-safe rules.

In [19] it was shown that the above algorithm can be used to answer queries in a combined knowledge base (KB, P), where P is a DL-safe program: $(KB, P) \models A$ if and only if $\mathsf{DD}(KB) \cup P \models A$, for A a ground atom. Assuming a bound on the arity of the predicates in P, query answering can be performed in time exponential in the size of KB and P. Furthermore, as shown in [17], the data complexity of these algorithms (i.e. the complexity assuming the size of the schema is fixed) is NP-complete, or even P-complete if disjunctions are not used.

3. A MAPPING SYSTEM FOR OWL-DL

Based on the definitions from [18], we now introduce the notion of an OWL-DL mapping system. The components of this mapping system are the the source ontology, the target ontology, and the mapping between the two.

Definition 3 (OWL-DL Mapping System) *An OWL-DL mapping system $\mathcal{MS}$ is a triple $(\mathcal{S}, \mathcal{T}, \mathcal{M})$, where*

- *$\mathcal{S}$ is the* source OWL-DL ontology*,*
- *$\mathcal{T}$ is the* target OWL-DL ontology*,*
- *$\mathcal{M}$ is the mapping between $\mathcal{S}$ and $\mathcal{T}$, i.e. a set of assertions $q_S \rightsquigarrow q_T$, where q_S and q_T are conjunctive queries over $\mathcal{S}$ and $\mathcal{T}$, respectively, with the same set of distinguished variables $\mathbf{x}$, and $\rightsquigarrow \in \{\sqsubseteq, \sqsupseteq, \equiv\}$.*

An assertion $q_S \sqsubseteq q_T$ is called a sound *mapping, requiring that q_S is contained by q_T w.r.t. $S \cup \mathcal{T}$; an assertion $q_S \sqsupseteq q_T$ is called a* complete *mapping, requiring that q_T is contained by q_S w.r.t. $\mathcal{S} \cup \mathcal{T}$; and an assertion $q_S \equiv q_T$ is called an* exact *mapping, requiring it to be sound and complete.*

A sound mapping $q_S \sqsubseteq q_T$ is equivalent to an axiom $\forall \mathbf{x} : q_T(\mathbf{x}, \mathbf{y}_T) \leftarrow q_S(\mathbf{x}, \mathbf{y}_S)$, while a complete mapping $q_T \sqsubseteq q_S$ is equivalent to an axiom $\forall \mathbf{x} : q_S(\mathbf{x}, \mathbf{y}_S) \leftarrow q_T(\mathbf{x}, \mathbf{y}_T)$. We call these assertions *general implication mappings* to distinguish them from special types of mappings that we define later.

The generality of the above definition captures a broad class of approaches for mapping systems. Let us discuss the expressiveness in terms of the ontology language, the query

language and the assertions. The source and target ontology are $\mathcal{SHOIN}(\mathbf{D})$ ontologies, i.e. logical theories that can have multiple models. In contrast, mapping systems in databases typically rely on simple relational schemas to describe the source and target, and each source is assumed to be one database (with a single model). The expressiveness of conjunctive queries corresponds to that of the well-known select-project-join queries in relational databases. Two typical approaches to specify mappings are the *global-as-view* (GAV) approach, where elements of the target are described in terms of queries over source, and the *local-as-view* (LAV) approach, where elements of the source are described in terms of queries over target. Our mapping system subsumes the approaches of GAV, LAV. In fact, it corresponds to the GLAV approach, which is more expressive than GAV and LAV combined [9].

We now define the semantics of the mapping system by translation into first-order logic.

Definition 4 (Mapping System Semantics) *For a mapping system $\mathcal{MS} = (\mathcal{S}, \mathcal{T}, \mathcal{M})$, let*

$$\pi(\mathcal{MS}) = \pi(\mathcal{S}) \cup \pi(\mathcal{T}) \cup \pi(\mathcal{M}).$$

The main inference for $\mathcal{MS}$ is computing answers of $Q(\mathbf{x}, \mathbf{y})$ w.r.t. $\mathcal{MS}$, for $Q(\mathbf{x}, \mathbf{y})$ a conjunctive query.

The intuitive reading of this semantics is that an answer of a query needs to be entailed by the source ontology $\mathcal{S}$, the target ontology $\mathcal{T}$ and the mappings $\mathcal{M}$. This semantics is equivalent to the usual model theoretic semantics (e.g. in [5]) based on local and global models, where a query answer must be an answer in *every* global model.

Query answering in such a mapping system of this general form is undecidable and requires a theorem prover. In the following we introduce special types of mappings that lead to decidable query answering and for which practical query answering algorithms exist.

3.1 Full Implication Mappings

The first class of mappings captures the mappings that can be directly expressed in OWL-DL. This is the case if q_s and q_t are DL-atoms of the form $P_s(\mathbf{x})$ and $P_t(\mathbf{x})$.

Concept Mappings. If q_s and q_t are of the form $P_s(x)$ and $P_t(x)$ and P_s, P_t are DL concepts, the mapping corresponds to the equivalent concept inclusion axiom.

Role Mappings. If q_s and q_t are of the form $P_s(x_1, x_2)$ and $P_t(x_1, x_2)$, with P_s and P_t are abstract or concrete roles, the mapping corresponds to the equivalent role inclusion axiom.

3.2 Restricted Implication Mappings

It is well-known that query answering for general implication mappings is undecidable due to the unrestricted use of non-distinguished (i.e. existentially bound) variables in either q_S or q_T. In the following, we define restrictions that reduce the expressivity of the mappings, but provide for a decidable query answering procedure.

DL-safe Mappings. Let us consider a sound mapping $q_S \sqsubseteq q_T$[1] with the assertion $\forall \mathbf{x} : q_T(\mathbf{x}, \mathbf{y}_T) \leftarrow q_S(\mathbf{x}, \mathbf{y}_S)$. In order to avoid introducing new objects in the interpretation domain, we disallow the use of non-distinguished variables in the query q_T, i.e. restrict the assertions to the form $\forall \mathbf{x} : q_T(\mathbf{x}) \leftarrow q_S(\mathbf{x}, \mathbf{y}_S)$. Please note that these assertions directly corresponds to SWRL rules. Analogously to safe rules, we call these mappings safe mappings. Query answering with such mappings is still undecidable in the general case. Therefore, we require the query q_S to be DL-safe (c.f. Definition 1), thus limiting the applicability of the rules to known individuals. Thus obtained mappings correspond to (one or more) DL-safe rules from Definition 2, for which efficient algorithms for query answering are known [19].

Mappings with Tree-like Query Parts. The restrictions introduced by DL-safety may appear rather strong. In the following we show how to relax the above restriction for a certain class of so-called tree-like queries. Using the query roll-up technique from [14], we can eliminate non-distinguished variables by reducing a tree-like part of a query to a concept, without loosing semantic consequences.

Definition 5 (Tree-Like Query Parts) *For a set of unary and binary literals S, the* coincidence graph *of S is a directed graph with the following structure:*

- *Each variable from S is associated with a unique node.*
- *Each occurrence of a constant in S is associated with a unique node, i.e. occurrences of the same constant are associated with distinct nodes.*
- *For each literal $C(s) \in S$, the node s is labeled with C.*
- *For each literal $R(s,t) \in S$, the nodes s and t are connected with a directed arc labeled R.*

The subset Γ of DL-atoms of a conjunctive query $Q(\mathbf{x}, \mathbf{y})$ is called a tree-like part *of $Q(\mathbf{x}, \mathbf{y})$ with a root s if*

- *no variable from Γ occurs in $Q(\mathbf{x}, \mathbf{y}) \setminus \Gamma$,*
- *the coincidence graph of Γ is a connected tree with a root s,*
- *all nodes apart from s are non-distinguished variables of $Q(\mathbf{x}, \mathbf{y})$.*

For details of the roll-up technique for tree-like queries, please refer to [14]; here we explain this technique on the following conjunctive query:

$$\exists y, z, w : R(x, y) \wedge A(y) \wedge S(y, z) \wedge B(z) \wedge T(y, w) \wedge C(w)$$

Since the entire query is tree-like with the root x, the existential quantifiers over z and w can be moved to the atoms where z and w first occur, yielding

$$\exists y : R(x, y) \wedge A(y) \wedge \quad [\exists z : S(y, z) \wedge B(z)] \wedge \; [\exists w : T(y, w) \wedge C(w)]$$

which is obviously equivalent to

$$\exists y : R(x, y) \wedge A(y) \wedge \exists S.B(y) \wedge \exists T.C(y).$$

Now the same procedure can be applied again, yielding the formula

$$\exists R.[A \wedge \exists S.B \wedge \exists T.C](x).$$

By these transformations we have eliminated the non-distinguished variables, i.e. we have "pushed them into description logic."

[1]For a complete mapping $q_S \sqsupseteq q_T$, the situation is analogous, with the roles of q_S and q_T reversed.

4. QUERY ANSWERING IN AN ONTOLOGY INTEGRATION SYSTEM

In this section we show how to use an OWL-DL mapping system from Section 3 for query answering in an *ontology integration system*, whose main task is to provide integrated access to a set of information sources, each expressed with a local source ontology. The integration is realized via a mediated, global ontology through which we can query the local ontologies.

For a set of local source ontologies $\mathcal{S}_1, \ldots, \mathcal{S}_n$, a global ontology $\mathcal{T}$ and corresponding mapping systems $\mathcal{MS}_1, \ldots, \mathcal{MS}_n$ with $\mathcal{MS}_i = (\mathcal{S}_i, \mathcal{T}, \mathcal{M}_i)$, an *ontology integration system* $\mathcal{IS}$ is again a mapping system $(\mathcal{S}, \mathcal{T}, \mathcal{M})$ with $\mathcal{S} = \bigcup_{i\in\{1\ldots n\}} \mathcal{S}_i$ and $\mathcal{M} = \bigcup_{i\in\{1\ldots n\}} \mathcal{M}_i$. The main inference task for $\mathcal{IS}$ is to compute answers of $Q(\mathbf{x}, \mathbf{y})$ w.r.t. $\mathcal{S} \cup \mathcal{T} \cup \mathcal{M}$, for $Q(\mathbf{x}, \mathbf{y})$ a conjunctive query over $\mathcal{T}$.

Algorithm 1 shows how to compute answers to a conjunctive query $Q(\mathbf{x}, \mathbf{y})$ in an ontology integration system $\mathcal{IS}$. It is based on the algorithm outlined in Section 2.3, from which it inherits certain limitations: $\mathcal{IS}$ is required to be based on $\mathcal{SHIQ}(\mathbf{D})$ knowledge bases, and the conjunctive query $Q(\mathbf{x}, \mathbf{y})$ and the queries in mappings are not allowed to contain transitive roles. The algorithm starts by eliminating non-distinguished variables from $Q(\mathbf{x}, \mathbf{y})$ and the mappings using the query roll-up technique. After roll-up, the obtained mappings and queries are required to be DL-safe, which is needed for decidable query answering. If this precondition is fulfilled, then the source ontology, target ontology and the mappings are converted into a disjunctive datalog program, and the original query is answered in the obtained program. By the results from [16, 19], it is easy to see that the algorithm exactly computes the answer of $Q(\mathbf{x}, \mathbf{y})$ in $\mathcal{IS}$.

Algorithm 1 Algorithm for Answering Queries in an Ontology Integration System

Require: ontology integration system $\mathcal{IS}$ and a conjunctive query $Q(\mathbf{x}, \mathbf{y})$

1: Roll-up tree-like parts of $Q(\mathbf{x}, \mathbf{y})$
2: Roll-up tree-like parts of query mappings in $\mathcal{M}$
3: Stop if $Q(\mathbf{x}, \mathbf{y})$ or some mapping from $\mathcal{M}$ is not DL-safe
4: $\Gamma \leftarrow \mathsf{DD}(\mathcal{S} \cup \mathcal{T} \cup \mathcal{M})$
5: Compute the answer of $Q(\mathbf{x}, \mathbf{y})$ in Γ

Let us contrast our approach to query answering with typical approaches in GAV and LAV data integration with respect to how queries against the target are reformulated to queries against the sources. In GAV systems, the problem reduces to simple view unfolding, as the reformulation is explicit in the mappings. In LAV, the problem requires more complex reasoning steps. In contrast, our approach does not require an explicit reformulation step. Instead, query answering here operates on a combined knowledge base consisting of source ontology, target ontology and mappings between them.

From the above definition, one might get the impression that the above algorithm requires that all source and target ontologies must be physically integrated into one mapping system in order to answer queries. This is, of course, not the case. More concretely, to compute $\mathsf{DD}(\mathcal{S} \cup \mathcal{T} \cup \mathcal{M})$, it is necessary to physically integrate the TBox and the RBox part of $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{M}$. Since the TBox and RBox are typically much smaller than the data, this does not pose practical problems. Accessing actual data sources (i.e. the ABoxes) is then governed by the chosen strategy for evaluating the disjunctive program.

5. EXAMPLE

We now present an example of a mapping system to illustrate the rather formal definitions from the previous sections. Let us assume that we need to establish semantic correspondences between two heterogeneous ontologies modeling the bibliography domain. Table 2 shows the definition of the source ontology $\mathcal{S}$, and the target ontology $\mathcal{T}$. The corresponding mappings $\mathcal{M}$ are shown in Table 3 and visualized in Figure 1. In the mappings we use the namespace prefixes *s:* and *t:* to denote elements of the source and target ontology, respectively.

Table 2: Source Ontology $\mathcal{S}$ and Target Ontology $\mathcal{T}$

Source Ontology	Target Ontology
$Person \sqsubseteq \top$	$Author \sqsubseteq \top$
$Publication \sqsubseteq \top$	$Entry \sqsubseteq \top$
$Article \sqsubseteq Publication$	$Article \sqsubseteq Entry$
$Thesis \sqsubseteq Publication$	$MasterThesis \sqsubseteq Entry$
	$PhDThesis \sqsubseteq Entry$
$Topic \sqsubseteq \top$	
$Person \sqsubseteq \forall name.String$	$Author \sqsubseteq \forall name.String$
$Topic \sqsubseteq \forall name.String$	
$\top \sqsubseteq \forall author.Person$	$\top \sqsubseteq \forall author.Author$
$Publication \sqsubseteq \forall title.String$	$Entry \sqsubseteq \forall title.String$
$Publication \sqsubseteq \forall isAbout.Topic$	$Entry \sqsubseteq \forall subject.String$

Table 3: Mapping $\mathcal{M}$

Correspondences
$Q_{s,1}(x, y) : s{:}Publication(x) \wedge s{:}title(x, y)$
$Q_{t,1}(x, y) : t{:}Entry(x) \wedge t{:}title(x, y)$
$m_1 : Q_{s,1} \sqsubseteq Q_{t,1}$
$Q_{s,2}(x) : s{:}Article(x)$
$Q_{t,2}(x) : t{:}Article(x)$
$m_2 : Q_{s,2} \sqsubseteq Q_{t,2}$
$Q_{t,3}(x) : s{:}Thesis(x)$
$Q_{s,3}(x) : (t{:}MasterThesis \sqcup t{:}PhDThesis)(x)$
$m_3 : Q_{s,3} \sqsubseteq Q_{t,3}$
$Q_{s,4}(x, y) : s{:}author(x, y)$
$Q_{t,4}(x, y) : t{:}author(x, y)$
$m_4 : Q_{s,4} \sqsubseteq Q_{t,4}$
$Q_{s,5}(x) : s{:}Person(x) \wedge s{:}author(y, x)$
$Q_{t,5}(x) : t{:}Author(x)$
$m_5 : Q_{s,5} \sqsubseteq Q_{t,5}$
$Q_{s,6}(x, z) : s{:}Publication(x) \wedge s{:}isAbout(x, y) \wedge s{:}name(y, z)$
$Q_{t,6}(x, z) : t{:}Entry(x) \wedge t{:}subject(x, z)$
$m_6 : Q_{s,6} \sqsubseteq Q_{t,6}$

Mapping m_1 maps publications from the source ontology along with their title to the corresponding entries of the target ontology. The sound mapping is expressed via the assertion:

$$\forall x, y : t{:}Entry(x) \wedge t{:}title(x, y) \leftarrow s{:}Publication(x) \wedge s{:}title(x, y)$$

This general implication mapping contains no non-distinguished variable in $Q_{t,1}$, so it can be expressed in a SWRL rule. However, the mapping is not DL-safe, as both x and y do not occur in non-DL predicates in $Q_{s,1}$. The mapping can

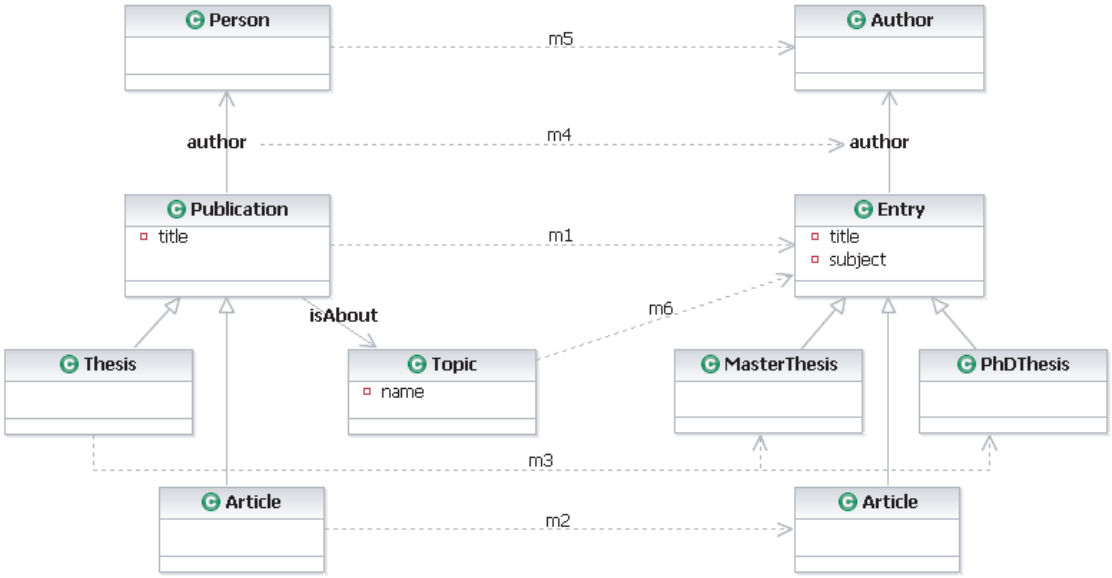

Figure 1: Example Mapping

be made DL-safe (as explained previously) by binding these variables with the special non-DL predicate $\mathcal{O}$ to individuals that actually occur in the source ontology:

$$\forall x, y : t{:}Entry(x) \wedge t{:}title(x, y) \leftarrow s{:}Publication(x) \wedge s{:}title(x, y) \wedge \mathcal{O}(x) \wedge \mathcal{O}(y)$$

Mapping m_2 maps the articles in the source ontology to articles in the target ontology, which can be expressed with a simple concept mapping:

$$s{:}Article \sqsubseteq t{:}Article$$

Mappings m_3 demonstrates the use of complex concepts in a concept mapping; it maps the concept Thesis in the source ontology to the union of the concepts PhDThesis and MasterThesis:

$$s{:}Thesis \sqsubseteq (t{:}MasterThesis \sqcup t{:}PhDThesis)$$

It shows that because of the expressiveness of the ontology language, we are able to express disjunctions in mappings (despite the fact that the query language only allows conjunctive queries).

Mapping m_4 simply maps the author property of the source ontology to that of the target ontology. It can be expressed with a simple role mapping:

$$s{:}author \sqsubseteq t{:}author$$

Mapping m_5 maps persons that are authors of a publication in the source ontology to authors in the target ontology. For the query $Q_{s,5}$ we can apply the query roll-up technique presented in the previous chapter. $Q_{s,5}(x) : s{:}Person(x) \wedge s{:}author(y, x)$ is a tree-like query with the distinguished variable x as its root. It can thus be rolled-up to the semantically equivalent query $Q_{s,5}(x) : (\exists s{:}author^{-}.s{:}Person)(x)$. We can therefore express the mapping as a concept mapping:

$$\exists s{:}author^{-}.s{:}Person \sqsubseteq t{:}Author$$

Let us compare this mapping with a DL-safe mapping obtained without query roll-up:

$$\forall x : t{:}Author(x) \leftarrow s{:}Person(x) \wedge s{:}author(y, x) \wedge \mathcal{O}(x) \wedge \mathcal{O}(y)$$

With the latter assertion, only those persons whose publications are explicitly named in the source ontology will be mapped to authors of the target ontology, whereas the prior mapping only requires a publication to exist, but it does not require it to be explicitly named. To illustrate the difference, consider the following ABox of the source ontology as example:

$$s{:}Person(peter),\ s{:}Person(boris),\ (\exists s{:}author^{-})(peter),\ s{:}author(pub, boris)$$

A query against the target ontology $q_t(x) : t{:}Author(x)$ would return both individuals *peter* and *boris* as result for the mapping obtained with query roll-up, as for both individuals authored publications are known to exist. The same query, but evaluated with the latter mapping, would return only the individual *boris* as query result, as the the publication of which *peter* is an author, is not explicitly named in the ontology.

Finally, mapping m_6 maps the topic classification of the publications. Please note that the topics in the source ontology are modeled as a separate concept, whereas in the target ontology the entries carry the name of the topic as a property: The name of the topic in the source ontology maps to the subject of the entry in the target ontology. The mapping can be expressed with the following assertion:

$$\forall x, z : t{:}Entry(x) \wedge t{:}subject(x, z) \leftarrow s{:}Publication(x) \wedge s{:}isAbout(x, y) \wedge s{:}name(y, z)$$

This mapping is again not DL-safe. Also, neither $Q_{s,6}$ nor $Q_{t,6}$ are tree-like queries, so query roll-up can not be applied. To make the mapping DL-safe, we again require the

variables to be bound to explicitly named individuals:

$$\forall x, z : t{:}Entry(x) \wedge t{:}subject(x, z) \leftarrow s{:}Publication(x) \wedge s{:}isAbout(x, y) \wedge s{:}name(y, z) \wedge \mathcal{O}(x) \wedge \mathcal{O}(y) \wedge \mathcal{O}(z)$$

6. RELATED WORK

Representation of Mappings. Our work is based on the formalization of a mapping system introduced in [18], which, because of its generality, subsumes a large class of mapping representations. Aspects that are not captured by this model include probabilistic or fuzzy mappings such as in [3] and the notion of context as e.g. introduced in C-OWL [4].

Data Integration. The task of ontology integration as application of the mapping system is very related to that of data integration in databases. In [18] the author introduces a general framework for data integration and compares existing approaches to data integration (GAV, LAV) along this framework. He also discusses *query processing approaches* for GAV and LAV, as well as the topic of *inconsistencies between sources*, and *reasoning on queries*. In [10] Halevy gives a status report on data integration, describing the recent progress on (i) schema mediation languages (LAV, GAV, GLAV), (ii) query answering algorithms (*view unfolding in GAV*, *answering queries using views in LAV*), (iii) query optimization, (iv) query execution , and (v) industry development. [6] presents a description logic based approach to data integration. In this framework, the sources are described using views over a mediated schema (LAV). The Description Logic DLR is used to model the components of the data integration system. As query expressions, non-recursive datalog queries are allowed. Views are defined based on these query expressions. A query answering algorithm based on reduction of answering queries using views to unsatisfiability is presented.

Ontology Integration. The work on data integration has been extended and re-applied to ontology integration in [5]. Here the authors follow the classical distinction between LAV and GAV approaches and outline query answering algorithms for these specific settings. In contrast to this work, query answering in our ontology integration system is not bound to these restricted forms of mappings. The main distinction between ontology integration and the existing approaches in data integration in databases is that in data integration one assumes that the sources basically are one *database*, whereas in ontology integration a source ontology is an *arbitrary logical theory*, which can have multiple models. Further, in data integration the languages to describe the sources and targets are typically very restricted (e.g. express the schemas as plain relations). Finally, the approaches are often limited to either LAV or GAV, whereas we do not make restrictions here.

Mapping Discovery. A final related problem is that of ontology matching and ontology alignment in the line of [7] and [20], where the goal is to manually or (semi-)automatically identify correspondences between ontologies, which finally result in mappings expressed in some formalism. Our work can be seen as complementary in the sense that the identified correspondences can be expressed in our mapping system and applied for tasks such as ontology integration.

7. CONCLUSIONS

We have presented the formalization of a general mapping system for OWL-DL ontologies. In this mapping system, the mappings between source and target ontology are specified as correspondences between conjunctive queries against the ontologies. The expressiveness of the mapping system is embodied in the ontology language ($\mathcal{SHOIN}(\mathbf{D})$), the supported query language (conjunctive queries), and the flexibility of assertions (GLAV approach). We have further identified a decidable fragment of mappings and a practical query answering algorithm for the task of ontology integration. All components of the mapping system can be fully expressed in OWL-DL extended with DL-safe rules. It thus integrates well with current efforts for rule extensions to OWL. The presented algorithms are implemented in the KAON2 ontology management system (`http://kaon2.semanticweb.org/`).

Acknowledgments

Research reported in this paper has been partially financed by the EU in the IST projects SEKT (IST-2003-506826, `http://www.sekt-project.com/`) and DIP (IST-2003-507483, `http://dip.semanticweb.org/`).

8. REFERENCES

[1] F. Baader and P. Hanschke. A Scheme for Integrating Concrete Domains into Concept Languages. In *Proc. of the 12th Int'l Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 452–457, Sydney, Australia, 1991.

[2] C. Beeri and R. Ramakrishnan. On the power of magic. In *Proceedings of the Sixth ACM Symposium on Principles of Database Systems*, pages 269–293, San Diego, CA, March 1987.

[3] P. Bouquet, M. Ehrig, J. Euzenat, E. Franconi, P. Hitzler, M. Krötzsch, L. Serafini, G. Stamou, Y. Sure, and S. Tessaris. Specification of a common framework for characterizing alignment, knowledgeweb deliverable d2.2.1v2. Technical report, Institut AIFB, Universität Karlsruhe, 2004.

[4] P. Bouquet, F. Giunchiglia, F. Van Harmelen, L. Serafini, and H. Stuckenschmidt. C-owl: contextualizing ontologies. In *Proc. of the 2nd International Semantic Web Conference (ISWC 2003)*, Sanibel Island (Fla.), 20-23 October 2003.

[5] D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration. In *Proceedings of the First Semantic Web Working Symposium*, pages 303–316, 2001.

[6] D. Calvanese, G. De Giacomo, and M. Lenzerini. Description logics for information integration. In A. Kakas and F. Sadri, editors, *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski*, volume 2408 of *Lecture Notes in Computer Science*, pages 41–60. Springer, 2002.

[7] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 662–673, New York, NY, USA, 2002. ACM Press.

[8] R. Fagin, P. Kolaitis, R. Miller, and L. Popa. Data exchange: Semantics and query answering. In *ICDT 2003, Siena, Italy*, 2003.

[9] M. Friedman, A. Levy, and T. Millstein. Navigational plans for data integration. In *Proceedings of AAAI '99/IAAI '99*, pages 67–73, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence.

[10] A. Y. Halevy. Data integration: A status report. In *BTW 2003, Datenbanksysteme für Business, Technologie und Web, Tagungsband der 10. BTW-Konferenz, 26.-28. Februar 2003, Leipzig*, pages 24–29, 2003.

[11] I. Horrocks and P. F. Patel-Schneider. A proposal for an OWL rules language. In *Proc. of the Thirteenth International World Wide Web Conference (WWW 2004)*, pages 723–731. ACM, 2004.

[12] I. Horrocks and P. F. Patel-Schneider. Reducing OWL Entailment to Description Logic Satisfiability. *Journal of Web Semantics*, 1(4), 2004.

[13] I. Horrocks, U. Sattler, and S. Tobies. Practical Reasoning for Very Expressive Description Logics. *Logic Journal of the IGPL*, 8(3):239–263, 2000.

[14] I. Horrocks and S. Tessaris. Querying the semantic web: a formal approach. In Ian Horrocks and James Hendler, editors, *Proc. of the 13th Int. Semantic Web Conf. (ISWC 2002)*, number 2342 in LNCS, pages 177–191. Springer-Verlag, 2002.

[15] U. Hustadt, B. Motik, and U. Sattler. Reasoning for Description Logics around $\mathcal{SHIQ}$ in a Resolution Framework. Technical Report 3-8-04/04, FZI, Karlsruhe, Germany, April 2004. http://www.fzi.de/wim/publikationen.php?id=1172.

[16] U. Hustadt, B. Motik, and U. Sattler. Reducing $\mathcal{SHIQ}^-$ Description Logic to Disjunctive Datalog Programs. In *Proc. of the 9th Conference on Knowledge Representation and Reasoning (KR2004)*. AAAI Press, June 2004.

[17] U. Hustadt, B. Motik, and U. Sattler. Data Complexity of Reasoning in Very Expressive Description Logics. In *Proc. IJCAI 2005*, Edinburgh, UK, 2005. Morgan-Kaufmann Publisher.

[18] M. Lenzerini. Data integration: a theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM Press, 2002.

[19] B. Motik, U. Sattler, and R. Studer. Query Answering for OWL-DL with Rules. In *Proceedings of the 3rd International Semantic Web Conference (ISWC 2004)*, Hiroshima, Japan, NOV 2004.

[20] N. Fridman Noy. Tools for mapping and merging ontologies. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 365–384. Springer, 2004.