QUERY PROCESSING IN ONTOLOGY-BASED PEER-TO-PEER SYSTEMS

Heiner Stuckenschmidt, Frank van Harmelen and Fausto Giunchiglia

December 2003

Technical Report # DIT-03-083

Query Processing in Ontology-Based Peer-to-Peer Systems

Heiner Stuckenschmidt[1], Frank van Harmelen[1], and Fausto Giunchiglia[2]

Contents

Abstract. The unstructured, heterogeneous and dynamic nature of the Web poses a new challenge to query-answering over multiple data sources. The so-called Semantic Web aims at providing more and semantically richer structures in terms of ontologies and meta-data. A problem that remains is the combined use of heterogeneous sources. In a dynamic environment, it is no longer realistic to assume that the involved data sources act as if they were a single (virtual) source, modelled as a global schema, as is done in classical data integration approaches. In this paper, we propose an alternative approach where we replace the role of a single virtual data source schema with a peer-to-peer approach relying on limited shared (or: overlapping) vocabularies between peers. Since overlaps between vocabularies of peers will be limited and the dynamic nature of the system prohibits the design of accurate mappings, query processing will have to be approximate. We provide a formal model for such approximate query processing based on limited shared vocabularies between peers, and we show how the quality of the approximation can be adjusted in a gradual manner. The result is a flexible architecture for query-processing in heterogenous and dynamic environments, based on a formal foundation. We present the approach and discuss it on the basis of a case study.

Keywords: Semantic Web, Methods and Formalisms for Knowledge Sharing, Knowledge-Based Mediation Architectures

1. Introduction

1.1. Semantic Web and Peer-to-Peer

The approach to query-processing that we present in this paper is strongly motivated by the peer-to-peer (P2P) architecture [12] that we expect for the Semantic Web. In this section we will argue why we expect the Semantic Web to have such a peer-to-peer architecture.

When we look at the current World Wide Web, we see in fact a mixed architecture, that is partly client/server-based, and partly P2P. On the one hand, each node in the network can directly address every other node in the network in a single, flat, world-wide address space, giving it the structure typical of many P2P networks. On the other hand, in practice there is currently a strong asymmetry between nodes in this address space that act as content-servers, and nodes that act as clients. Recent estimates indicate the presence of 50 million web-servers, but as many as 150 million clients. On the scale of the World Wide Web, any form of centralization would create immediate bottlenecks, in terms of network throughput and server capacity.

This need for a flat, non-server-centered architecture will be even stronger on the Semantic Web. Of course, the same physical load-balancing arguments hold as on the current Web, but the Semantic Web adds a new argument in favor of a P2P-style argument. On the Semantic Web, any server-centered architecture will not only create physical bottlenecks, but as communication relies on the use of ontologies will also create *semantic bottlenecks*. Since the semantics of information will be explicit (or at least: more explicit) on the Semantic Web, any single server will in a way "impose" a particular semantic view on all its clients. This will have undesirable consequences, both in terms of the pluriformity of the available information, as well as in terms of the size of the central ontology that such information-servers would have to maintain.

Instead, a P2P-style architecture will be able to avoid both the physical and the semantic bottlenecks. Different semantic views, expressed in terms of different ontologies, will be provided by many peers in a flat network of peers, each employing their own local, small ontology. Of course, this increased flexibility comes at a price: such "different semantic views, in terms of different ontologies" creates a significant data-integration problem: how will these peers be able to communicate if they do not share the same view on their data? In the remainder of this paper, we propose an approach where the communication between peers relies on a limited shared vocabulary between them. This replaces the role of the single virtual database schema that is the traditional basis for solving data integration problems.

In the following, we will briefly point to existing work on integration of heterogeneous databases, and we will see that this work is predominantly based on the notion of a global schema that is connected to the heterogeneous schemas to be integrated. Subsequently, we argue why this traditional approach is no longer viable in a peer-to-peer style network as the Semantic Web will be. The remainder of the paper will then be devoted to describing our proposal for such new approaches that will enable us to do query processing in a peer-to-peer setting without the need for global integrating schemas.

1.2. The Need for New Approaches

The problem of integrating heterogeneous database schemas [10] has been addressed by many researchers.

The integration is normally done using a global schema that is connected to the heterogeneous schemas to be integrated by a number of views. We can distinguish two general approaches [7]:

- *Global-As-View:* In the global-as-view approach every relation in the global schema is defined as a view over the different schemas to be integrated.
- *Local-As-View:* In the local-as-view approach, views are used to define the schemas of local information sources in terms of the global one.

The benefits of using explicit semantic models, i.e. ontologies, has been recognized in many approaches. A survey of approaches using ontologies is provided by [19]. Description logics have been proven to be a useful formalism for specifying and reasoning about semantic models [3] to support information integration. It has been shown that results from the database area provide solutions for the integration of semi-structured information (see e.g. [1]).

However, some peculiar characteristics of P2P networks and, in particular, the fact that they are characterized by strong dynamics, require the development of new solutions, which substantially extend the current data integration technology. This issue has been discussed in length in [5]. We report here only the main ideas. Consider the situation where John, a person living in Toronto, is described in the database F of his family doctor, and also in the database H of the hospital where he once received medical treatment.

Example: John goes to another country, for instance Trentino in Italy. Unluckily, here he has an accident; he breaks a leg, and he must get medical aid. The medical office has its own database M which now needs to query H for the purpose of retrieving previous treatment details. Furthermore, a new record from M should appear in F. However the acquaintance between M and F does not need to be maintained for ever, since the two databases will probably not need to coordinate again.

1.2.1. Dropping the global schema In situations like that described in the example, the design and development of data integration mechanisms for randomly acquainted databases which may need to communicate only a few times, becomes impractical. In particular, it makes little sense to speak of a global schema [7], as we cannot think of a set of P2P databases just as an implementation of a single virtual database (this being the assumption which motivates the definition of a global schema). It is no longer possible to see the global schema as a view of the local database (global-as-view approach) or, vice versa, the local databases as views of the global database (local-as-view approach). For instance, we can no longer assume that there is a unique universe, containing all the elements of the single databases, but rather many overlapping domains. From a foundational point of view, any theory developed under the assumption of a global schema, and under the implicit assumption that the global schema is fixed, prevents us from the studying the dynamics of a P2P network. As far as we know, in the data integration literature, these two assumptions have never been relieved, see for instance [7, 10]. As a consequence, the problem handling heterogeneous information sources becomes a coordination task that reoccurs whenever two peers want to cooperate.

1.2.2. Good enough answers In a P2P network, it becomes hard to maintain high quality answers to queries, for instance the fact that data can flow among the databases preserving soundness and completeness. In this context, soundness means that the data provided by the local databases satisfy the global schema (but they are not necessarily complete, some of them can get lost in the coordination). Completeness has the dual meaning. In the data integration literature, completeness is often given up, still maintaining the request of soundness. In a P2P environment, it will be possible to have completeness and soundness only in limit cases, for instance with low dynamics or simplified interaction among the databases.

One area where there will often be interest in getting very high quality data integration is the medical care domain. There are however many other application domains where this is not the case. One such example is tourism. This domain is not life critical, and in many cases the small dimension of a single business (e.g., hotels) does not justify big investments. Consider the following example.

Example: When planning his vacation in Trentino, John goes to a local agency. The agency searches for single operators (hotels, for instance), and queries them for the necessary information (e.g., prices and availability).

In this, the dynamics will have a high impact on the quality of the answer. We have network variance: the relevant databases are much more unstable in their being active and coordinated in the network, nodes come and go (for instance depending on the season), and so on. We have database variance: John travels around and queries different databases. The same query will get different results since each database will implement different degrees of coordination with the others, and so on. Thus, for instance, a query about hotels made to a hotel database will likely get an answer that is better than the answer obtained from a campsite database. We also have query variance: if you ask a query about campsites to a campsite database you will likely get a better quality answer than if you ask this database a query about hotels. Depending on the query, certain coordination mechanisms may or may not be activated. However, in this application, the agency doesn't need the best possible answer. It simply needs some answer. As long as, for instance, it gets a hotel John likes, this is good enough.

Compared to the previous medical example, in the tourism example much lower quality data coordination will suffice. The medical care and tourism domains are just examples. Things can get even more radical and complex when one thinks of applications where some of the nodes are mobile and where coordination happens on an even more occasional basis, for instance due to the physical proximity of two mobile peers. In these situations, and for certain kinds of applications, almost any answer will suffice. In terms of quality of answers, we can go from one extreme to the other. On one extreme, it may be usual to get poor quality answers. This may happen because the databases interact partially or do not interact at all or, even worse, they pass around data which are wrong (for instance because of unsolved problems of semantic heterogeneity). On the other extreme, there will be a tight coordination and it will be possible to achieve or, at least, approximate soundness and completeness. Between these two extremes there is a continuum of answers of different quality. This observation coincides with the ideas of the Semantic Web, where it is widely agreed that completeness and correctness in a logical sense can not be reached in many cases.

2. Ontology-Based Peer-to-peer Systems

Before we can present our approach to query processing in ontology-based peer-to-peer systems, we have to specify the systems we are talking about in more detail. We assume a system of independent peers that encapsulate the (possibly redundant) information of the whole system. Each peer uses one or more ontologies to model the information. These ontologies are used as a conceptual schema of the actual information that can be seen as an instantiation of the ontology. Peers exchange knowledge by formulating queries using the vocabulary defined in the ontologies they use and sending them to other peers in the network. The task of the receiving peer is to determine the answers to these queries relative to its own vocabulary and information. This leads to a situation where we are rather concerned with heterogeneous knowledge bases that plain data sources in the conventional sense.

In order to get a clearer notion of the problem of processing queries in such systems we make some simplifying assumptions. First of all we will only consider two peers that want to communicate. Then we assume that there are only two ontologies involved, a shared one and a private one of the peer trying to communicate. We further assume that both ontologies are encoded on the same language, preventing us from the problem of integrating the ontology languages.

This simplified communication problem can easily be extended to more realistic scenarios as communication is mostly bi-lateral even in complex systems. There might be more than two ontologies involved in the communication, but they will all either be shared or private to one of the peers. The assumption that there are actually ontologies being shared by peers in the system is backed by the observation, that real-world ontologies are in most cases not build from scratch. It is rather common to at least start with an existing ontology (see for example `http://www.daml.org/` for a library of ontologies about various domains. The existence of an internal mapping between the ontologies used by an individual peer is likely because if the peer wants to use more than one ontology as a basis for its information, it has to know about their relation. As a single peer is a rather static system compared to the overall network, we can use schema matching techniques that have been developed in the database community in order to find correspondences (see [14] for an overview). The only assumption that really is a simplification is the existence of a single ontology language. Investigating this problem, however, is out of the scope of this paper.

In the following we give formal definitions for the parts of an ontology-based peer-to-peer system that are concerned with query processing. These parts include the definition of ontologies and mappings as well as the notion of queries and answers in the setting of ontology-based information.

2.1. Ontological Knowledge

A number of languages for encoding ontologies on the Web have been proposed (see [6] for an overview). In order to get a general notion of ontological knowledge, we define the general structure of a terminological knowledge base (ontology) and its instantiation independent of a concrete language.

Terminological knowledge usually groups objects of the world that have certain properties in common. A description of the shared properties is called a class definition. Classes can be arranged into a subclass-superclass hierarchy. Classes can be defined in two ways, by enumeration of its members or by stating that it is a refinement of a complex logical expressions. The specific logical operators to express such logical definitions can vary between ontology languages; the general definitions we give here abstract from these specific operators. Further relations can be specified in order to establish structures between classes. Terminological knowledge considers binary relations that can either be defined by restricting their domain and range or by declaring them to be a sub-relation of an existing one. In order to capture the actual information content of a knowledge base we allow to specify single objects, also called instances. In our view on terminological knowledge, instances can be defined by stating their membership in a class. Further, we can define instances of binary relations by stating that two objects form such a pair.

Definition 1 (Terminological Knowledge Base)**.** *A Terminological Knowledge Base $\mathcal{T}$ is a triple $\mathcal{T} = \langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle$ where $\mathcal{C}$ is a set of class definitions of the form:*

- $c \equiv (o_1, \cdots, o_n)$ *where c is a class definition and $o_1, \cdots, o_n$ are object definitions.*
- $c_1 \sqsubseteq c_2$ *where c_1 and c_2 are class definitions.*

$\mathcal{R}$ is a set of relation definitions of the form:

- $r \sqsubseteq (c_1, c_2)$ *where r is a role definition and c_1 and c_2 are class definitions.*
- $r_1 \sqsubseteq r_2$ *where r_1 and r_2 are role definitions.*

and $\mathcal{O}$ is a set of object definitions of the form:

- $o : c$ *where c is a class definition and o is an individual.*
- $(o_1, o_2) : r$ *where r is a relation definition and o_1, o_2 are object definitions.*

In the following, we will consider terminological knowledge bases that consist of such axioms. Of course, any specific ontology language will have to further instantiate these definitions to specify logical operators between classes etc, but for the purposes of this paper, these general definitions are sufficient. Further, we define the signature of a terminological knowledge base $\langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle$ to be a triple $\langle \mathcal{CN}, \mathcal{RN}, \mathcal{ON} \rangle$, where $\mathcal{CN}$ is the set of all names of classes defined in $\mathcal{C}$, $\mathcal{RN}$ the set of all relation names in $\mathcal{O}$ and $\mathcal{ON}$ the set of all object names occurring in $\mathcal{O}$.

We define a Tarski style semantics for the notion of terminological knowledge that is very much inspired by Description Logics (compare [3]). The formal definition of this semantics and the notion of logical consequence is omitted due to lack of space and can be found in an extended version of this paper [18].

2.2. Inter-Ontology Mappings

We assume that each peer has an integrated view on the ontologies it uses as a semantic foundation for its information. This integrated view is created by the mappings relates elements from different ontologies. The creation of these mappings is discussed in other work (see e.g. [14] for an overview) and is not further discussed in this paper. As our methods for query processing rely on these internal mappings of individual peers, we have to define the nature of the mappings. For this work, we adopted the mapping framework proposed in [11] summarized in the following.

Madhavan et.al. define mappings in terms of operations between expressions in two different domain models, in our case ontologies. They further demand that the resulting expression is consistent with the logical interpretation of the individual ontologies (we cover this point in the next section). As this framework is very generic, we instantiate it in the context of ontological knowledge as defined in the last section. In particular, we need to define the kinds of expressions, we consider and the operations used to relate them.

The expressions, we are considering here are the definitions of classes, relations and objects, respectively. Further, we define the following mappings between the different types of knowledge:

$$\begin{aligned} c_1 &\xleftrightarrow{m_C} c_2, m_C \in \{\sqsubseteq, \sqsupseteq, \equiv\} \\ r_1 &\xleftrightarrow{m_R} r_2, m_R \in \{\sqsubseteq, \sqsupseteq, \equiv\} \\ o_1 &\xleftrightarrow{\equiv} o_2 \end{aligned} \tag{1}$$

Intuitively the mappings marked with the operator $\equiv$ state that definitions in the different ontologies refer to the same class, relation or object. Mappings marked with $\sqsubseteq$ (and $\sqsupseteq$) state that the definition in the one ontology is a special (more general) case of the other definition. Further, Madhavan et.al. consider the use of a helper model individual models are mapped to. The helper model is then used to derive composed mappings between these models. In our view on ontology-based peer-to-peer systems, the shared ontologies can be seen as such helper models.

The formal semantics for terminological knowledge can easily be extended to cover mappings between different models. Here we assume a set of interpretation mappings into partially overlapping domains. Mappings impose constraints on these interpretations. A formal definition of the semantics of mappings can be found in [18].

2.3. Semantics and Logical Consequence

We can define semantics and logical consequence of a terminological knowledge base using an interpretation mapping $.^{\Im}$ into an abstract domain Δ such that:

- $c^{\Im} \subseteq \Delta$ for all class definitions c in the way defined above
- $r^{\Im} \subseteq \Delta \times \Delta$ for all relation definition r
- $o^{\Im} \in \Delta$ for all object definitions o

This type of denotational semantics is inspired by description logics [4], however, we are not specific about operators that can be used to build class definitions which are of central interest of these logics. Using the interpretation mapping, we can define the notion of a model in the following way:

Definition 2 (Model of a Terminological Knowledge Base)**.** *An interpretation* $\Im$ *is a model for the knowledge base* $\mathcal{T}$ *if* $\Im \models A$ *for every axiom* $A \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{O})$ *where* $\models$ *is defined as follows.*

- $\Im \models c \equiv (o_1, \cdots, o_n)$, *iff* $c^{\Im} = \{o_1^{\Im}, \cdots, o_n^{\Im}\}$
- $\Im \models c_1 \sqsubseteq c_2$, *iff* $c_1^{\Im} \subseteq c_2^{\Im}$
- $\Im \models r \sqsubseteq (c_1, c_2)$, *iff* $r^{\Im} \subseteq c_1^{\Im} \times c_2^{\Im}$
- $\Im \models r_1 \sqsubseteq r_2$, *iff* $r_1^{\Im} \subseteq r_2^{\Im}$
- $\Im \models o : c$, *iff* $o^{\Im} \in c^{\Im}$
- $\Im \models (o_1, o_2) : r$, *iff* $(o_1^{\Im}, o_2^{\Im}) \in r^{\Im}$

In order to be able to handle multiple ontologies, we have to define the interpretation mapping over different models and mappings between them. In the following, we only consider an interpretation for two ontologies and mappings between them. The definitions, however, can easily be extended to more than two ontologies.

First of all, we divide the interpretation mapping $\Im$ into two sub-mappings $\Im^1$ and $\Im^2$ each defining the interpretation for one of the two ontologies in the way described above. Further, we define the interpretation of mappings between the ontologies in the following way:

$$\Im \models (c_1 \xleftrightarrow{m_C} c_2) \text{ iff } \begin{cases} c_1^{\Im^1} = c_2^{\Im^2} \; for\, m_C = \equiv \\ c_1^{\Im^1} \subseteq c_2^{\Im^2} \; for\, m_C = \sqsubseteq \\ c_1^{\Im^1} \supseteq c_2^{\Im^2} \; for\, m_C = \sqsupseteq \end{cases} \tag{2}$$

$$\Im \models (r_1 \xleftrightarrow{m_R} r_2) \text{ iff } \begin{cases} r_1^{\Im^1} = r_2^{\Im^2} \; for\, m_R = \equiv \\ r_1^{\Im^1} \subseteq r_2^{\Im^2} \; for\, m_R = \sqsubseteq \\ r_1^{\Im^1} \supseteq r_2^{\Im^2} \; for\, m_R = \sqsupseteq \end{cases} \tag{3}$$

$$\Im \models (c_1 \xleftrightarrow{\equiv} c_2) \text{ iff } o_1^{\Im^1} = o_2^{\Im^2} \tag{4}$$

These definitions enable us to perform reasoning across different ontologies using the notion of logical consequence:

Definition 3 (Logical Consequence). *An axiom A logically follows from a set of axioms $\mathcal{S}$ if $\Im \models \mathcal{S}$ implies $\Im \models A$ for every model $\Im$. We denote this fact by $\mathcal{S} \models A$.*

In order to perform logical reasoning, we can use existing reasoning systems that have been build for reasoning about description logic knowledge bases. The system that has been used in the case study is the FaCT system [8].

2.4. Ontology-Based Queries

As mentioned in the beginning of this section, the peers in a system will exchange information by querying information from other peers in the network using terms from their own ontology. In the following we first define ontology based queries as well as the notion of answers to and relations between queries. We formalize queries in the following way: conjuncts of a query are predicates that correspond to classes and relations of the ontology. Further, variables in a query may only be instantiated by constants that correspond to objects in that ontology.

Definition 4 (Terminological Queries). *Let V be a set of variables disjoint from $\mathcal{ON}$ then an terminological query Q over a knowledge base $\mathcal{T} = \langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle$ is an expressions of the form $Q(\bar{X}) \leftarrow q_{1_i} \wedge \cdots \wedge q_{m_i}$ where q_i are query terms of the form $x : c$ or $(x, y) : r$ such that $x, y \in V \cup \mathcal{ON}$, $c \in \mathcal{CN}$ and $r \in \mathcal{RN}$ or are of the form $x = o$ where $x \in V$ and $o \in \mathcal{ON}$*[1].

The following expression in an example query based on an ontology used in the case study described later. It asks for hotels in castles that are located in towns in Mecklenburg and have less than 25 rooms:

$$\begin{aligned} Q(X) \quad \leftarrow \quad & X : Hotel \wedge (X, Z) : hat - Zimmer \wedge \; Z \leq 25 \wedge \\ & (X, W) : ist - in - Schloss \wedge W = ja \wedge (X, Y) : liegt - in - Ort \wedge \\ & (Y, V) : liegt - in - Land \wedge V = mecklenburg \end{aligned} \tag{5}$$

[1]Note that this may include data-type expressions as the type itself is can be considered to be a class, the actual value an instance of that class and the comparison operator a special relation.

The fact that all conjuncts relate to elements of the ontology allows us to determine the answer to terminological queries in terms of instantiations of the query that are logical consequences of the knowledge base. The computation of query answers in the sense being defined above is the main inference task of peers within a system. In the next section we will discuss a logically well-founded approach for computing such answers.

3. Query Processing

After having introduced some basic notions related to ontology-based peer-to-peer systems we now turn our attention to the problem of query processing. The definition of query answers provides us with a deductive definition that describes correct answers with respect to a query over a single specific ontology. In the case of an ontology-based peer-to-peer system, however, we face a situation where we have to deal with more than one ontology. In many cases, the answering peer does not know all terms used in the query expression, because they are taken from the local ontology of the asking peer. In order to overcome this problem, we have to align the vocabularies the asking and the answering peer. In [16] we describe an approach for approximately translating classes from one ontology into another. We briefly summarize this approach in the next section and extend it to conjunctive queries. Further, we discuss the issue of the quality of approximations generated by our approach.

3.1. Approximating Class Descriptions

The notion of an interpretation given above is a very general one and does not restrict the nature of members of a concept. This is done by the use of operators for defining classes. These kinds of operators restrict the possible members of a class using an interpretation mapping to an abstract domain Δ. Figure 1 defines some operators we use in the following in order to define classes. These operators include the definition of a class by enumerating its members, the conjunction of two classes interpreted as the intersection of their members and existential restriction on a certain property defining that all members of that class have to be related to at least one object of a certain type C by the relation P (compare figure 1).

Operator	**Extension** $.^{\Im}$
$C_1 \sqcap C_2$	$C_1^{\Im} \cap \cdots \cap C_n^{\Im}$
$\{x_1, \cdots, x_n\}$	$\{x_1, \cdots, x_n\} \subset \Delta$
$(\exists P.C)$	$\{y \in \Delta \| \exists x((y,x) \in P^{\Im}) \wedge x \in C^{\Im}\}$
$(\leq n)$	$\{x^{\Im} \in N \| x^{\mathcal{E}} \leq n\}$

FIGURE 1. Some operators for Constraining Classes

These kinds of restriction are the basis for deciding whether a class definition is equivalent, more specialized or more general than another. Formally, we can decide whether one of the following relations between two classes can be deduced from the ontology:

subsumption:: $C_1 \sqsubseteq C_2 \iff C_1^{\Im} \subseteq C_2^{\Im}$
membership:: $x : C \iff x^{\Im} \in C^{\Im}$

The classes in an ontology form a hierarchy with respect to the subsumption relation. In the case of multiple ontologies connected by mappings such a hierarchy can also be computed for the united set of classes. Therefore, we will always have a set of direct super- and a set of direct subclasses of a class c_1 from the private ontology. We can use those direct sub- and super classes that belong to the shared ontology as upper and lower approximation for c_1 in the shared ontology:

Definition 5 (Lower Approximation)**.** *Let $\mathcal{C}_1$ be the set of classes of a private ontology, $\mathcal{C}_2$ the set of classes of a shared ontology and $c \in C_1$, then a class $c_{glb} \in \mathcal{C}_2$ is called a lower approximation of c in $\mathcal{C}_2$, if the following assertions hold:*

1. $c_{glb} \sqsubseteq c$
2. $(\exists c' \in \mathcal{C}_2 : c' \sqsubseteq c) \implies (c' \sqsubseteq c_{glb})$

The greatest lower bound $glb_{\mathcal{C}_2}(c)$ denotes the set of all lower approximations of c in $\mathcal{C}_2$.

Definition 6 (Upper Approximation)**.** *Let $\mathcal{C}_1$ be the set of classes of a private ontology, $\mathcal{C}_2$ the set of classes of a shared ontology and $c \in \mathcal{C}_1$, then a class $c_{lub} \in \mathcal{C}_2$ is called an upper approximation of c in $\mathcal{C}_2$, if the following assertions hold:*

1. $c \sqsubseteq c_{lub}$
2. $(\exists c' \in \mathcal{C}_2 : c \sqsubseteq c') \implies (c_{lub} \sqsubseteq c')$

The least upper bound of $lub_{IS_2}(c)$ is the set of all upper approximations of c in $\mathcal{C}_2$.

The rational of using these approximations is that we can decide whether an entity x is a member of a class in the private ontology based on its membership in classes of the shared ontology. This decision in turn provides us with an approximate result on deciding whether x is the result of a query stated in terms of the private ontology, based on the following observation:

- If x is member of a lower bound of c_1 then it is also in c_1
- If x is not member of all upper bounds of c_1 then it is not in c_1

In [15] Selman and Kautz propose to use this observation about upper and lower boundaries for theory approximation. We adapt the proposal for defining an approximate classifier M' that assigns members of shared classes to private ones in the following way:

Definition 7 (Class Approximation)**.** *Let $\mathcal{C}_1$ be the set of classes of a private ontology, $\mathcal{C}_2$ the set of classes of a shared ontology and x member of a class in $\mathcal{C}_2$, then for every $c_1 \in \mathcal{C}_1$ we define M' such that:*

- $M'(x, c_1) = 1$ *if* $x : \left(\bigvee_{c \in glb_{IS_2}(c_1)} c \right)$
- $M'(x, c_1) = 0$ *if* $x : \neg \left(\bigwedge_{c \in lub_{IS_2}(c_1)} c \right)$
- $M'(x, c_1) = ?$*, otherwise*

Where the semantics of disjuction and conjunction is defined in the obvious way using set union and intersection.

Based on the observation about the upper and lower bounds, we can make the following assertion about the correctness of the proposed approximate classification:

Proposition 1 (Correctness of Approximation)**.** *The approximation from definition 7 is correct in the sense that:*

1. *If* $M'(x, c_1) = 1$ *then* $x^{\Im} \in c_1^{\Im}$
2. *If* $M'(x, c_1) = 0$ *then* $x^{\Im} \notin c_1^{\Im}$

Using the definition of upper and lower bounds the correctness of the classification can be established using the model-based semantics of ontologies and mappings (see [17] for a proof).

3.2. Queries as Classes

The result of the last section provides us with the possibility to compute a set of objects that are definitely members of a class expression and a set of objects that are possibly members of a class. This approach can directly be used to answer trivial queries that only ask for members of a particular class. We have shown that a slight variation of the mechanism can also be used to approximate Boolean queries over class names [16]. In order to compute (approximate) answers for ontology-based conjunctive queries, however, we also have to deal with binary relations in the query expression. In order to cope with relations as well, we use a method for translating conjunctive queries into class expressions that has been proposed by Horrocks and Tessaris

[9]. The idea of the approach of Horrocks and Tessaris now to translate the query into an equivalent class expression, classify this new class and use standard inference methods to check whether an object is an instance of the query expression. This approach makes use of the fact that binary relations in a conjunctive query can be translated into an existential restriction in such a way that logical consequence is preserved after a minor modification of the object definition part of the ontology. Details are given in the following theorem.

Proposition 2 (Role Roll-Up [9])**.** *Let* $\mathcal{T} = \langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle$ *be an ontology. Let further* R *be a role,* C_I *a set of class names in* $\mathcal{CN}$ *and* $a, b \in \mathcal{ON}$ *individual names. Given a new class name* P_b *not appearing in* $\mathcal{CN}$*, then* $\langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle \models (a, b) : R \wedge b : C_1 \wedge \cdots \wedge b : C_k$ *if and only if* $\langle \mathcal{C}, \mathcal{R}, \mathcal{O} \cup \{b : P_b\} \rangle \models a : \exists R(P_b \sqcap C_1 \sqcap \cdots \sqcap C_k)$

The transformation of a complete query is more difficult due to the dependencies between the variables that occur in the query expression. In order to keep track of these dependencies during the transformation Horrocks and Tessaris introduce the notion of a query graph.

Definition 8 (Query Graph (Horrocks and Tessaris 2000))**.** *The graph induced by a query is a directed graph with a node for every variable and individual name in the query and an directed edge from node x to node y for every role term $(x, y) : R$ in the query.*

The correct transformation of a query into a class expression depends on the kinds of dependencies between the variables in the query which is reflected in the structure of the query graph. While the approach of Horrocks and Tessaris is more general, we restrict ourselves to queries where the query graph is a (directed) tree and its root node corresponds to the variable we are interested in. In especially, this requires that none of the roles used in the query is declared to be functional and that each constant only appears once in a query. While using this simplification, we would like to emphasize that the translation can be done for unions of conjunctive queries with an arbitrary number of result variables and a very expressive logical language for defining class expressions. Our simplifying assumptions lead to a simple method for transforming a query graph into a class expression. The result of applying this translation technique to our example query in equation 5 is the following expression:

$$\begin{aligned}(Hotel \quad \sqcap \quad & (\exists\, liegt-in-Ort.(\exists\, liegt-in-Land.\{mecklenburg\})) \sqcap \\ & (\exists\, hat-Zimmer.(\leq 25)) \sqcap \\ & (\exists\, ist-in-Schloss.\{ja\}))\end{aligned} \tag{6}$$

As this expression defines a new class in the overall ontology we can now apply the approximation techniques described in the last section in order to compute the sets of possible and the set of definite answers to the query. We will use this query as a running example in the following discussion on approximations.

3.3. Quality of Approximation

Unfortunately, proving the correctness of the approximation says nothing about the quality of the approximation. In the worst case, the upper and lower boundaries of concepts in the other hierarchy are always $\top$ and $\bot$ respectively. In this case the translated query always returns the empty set as result. We were not able to investigate the quality of approximations on theoretical level, however, we can provide some rules of thumb that can be used to predict the quality of an approximation:

Depths of hierarchies:: The first rule of thumb, we can state is that deeper class hierarchies lead to better approximations. For hierarchies of depth one it is easy to see that we will not be able to find good upper and lower bounds. We can also assume that deeper hierarchies provide finer grained distinctions between concepts that in turn often produce closer approximations.

Degree of overlap:: Our approach assumes a shared ontology, however, we cannot guarantee that different systems indeed use the same parts of this shared vocabulary. Therefore, the actual overlap of terms used in the existing definitions that are compared is important for predicting the quality of approximations. In general, we can assume that a high degree of overlap leads to better approximations.

Both criteria used in the rules of thumb above strongly depend on the application and on the creator of the corresponding models. At least for the degree of overlap, we can assume that hierarchies that are concerned with the same domain of interest will share a significant part if the vocabulary, thus enabling us to compute reasonable approximations.

In the course of a case study it turned out that in most cases the approximation of concept expressions returns good results, because people tend to share a reasonable number of concept names across different ontologies that provide a basis for creating mappings. These mappings can often be found using stemming and simple string matching. On the other hand, it turned out that it is much harder to come up with reasonable mapping between the relations used in different ontologies leading to a situation where we only have very sparse mappings between these relations. This in turn has a major impact on the quality of approximation applied to conjunctive queries. In fact the lack of mappings between relations often leads to a situation, where answers could not be found, because names of relations in the query were not known in the ontology in the answering peer. In the next section, we discuss an approach to overcome the problem of sparse mappings between relation names.

4. Query Relaxation

In the presence of sparse mappings, we face a situation where the descriptions of different peers referring to the same real-world object can be significantly different. In most cases, the descriptions are different in the sense that different relations are used to related same object to other objects in the domain. These relations may refer to the same properties of the object that cannot be matched due to a missing mapping or the set of properties itself used might be different. As a consequence, real-world objects that are meant to be an answer to a query are not returned because their description does not match the query that is formulated using terms form a different ontology. We address this problem by relaxing the query, i.e. by weakening those constraints from the query expression that are responsible for the failure. In order to be useful, this weakening process has to fulfill certain formal properties. In especially, we want to make sure that we do not loose any answers when modifying the query. We can guarantee this using the notion of query subsumption as described by Halevy:

Definition 9 (Query Containment and Equivalence (Halevy 2001))**.** *Let $\mathcal{T} = \langle \mathcal{C}, \mathcal{R}, \mathcal{O} \rangle$ and Q_1, Q_2 conjunctive queries over $\mathcal{T}$. Q_1 is said to be contained in another query Q_2 denoted by $Q_1 \sqsubseteq Q_2$ if for all possible sets of object definitions of a terminological knowledge base the answers for Q_1 is a subset of the answers for Q_2 : $(\forall \mathcal{O} : res(Q_1) \subseteq res(Q_2))$. The two queries are said to be equivalent, denoted as $Q_1 \equiv Q_2$ iff $Q_1 \sqsubseteq Q_2$ and $Q_2 \sqsubseteq Q_1$*

Based on these notions we compute a sequence of queries $Q_0, \cdots, Q_n$ such that the following properties hold:

1. $Q_0 \equiv Q$
2. $i < j \Longrightarrow Q_i \sqsubseteq Q_j$

The intuition behind this approach is to start with the original query and generate queries where each is more general than the one before, i.e. each query following in the sequence returns all results of the previous one, but might return more results. Our hope is that these new results contain the description of some real-world objects that should be answers, but were not found due to their description.

There are many different ways of making a query more general in order to increase the chance of matching a potential answer. In the following we discuss relaxation heuristics we consider useful for the purpose of query processing in a peer-to-peer setting.

4.1. Variable Elimination

The first heuristic is based on the fact that each variable in a conjunctive might fail to match a specific object if the object does not satisfy the constraints. Therefore, a way of increasing the chance of matching the target object in the head of the query is to successively eliminate non-answer variables from the query. In the example query in equation 5 for example, we have the variables V,W,X,Y and Z where X is the answer variable. Therefore we can weaken query by eliminating the variables V,W,Y and Z. This can be done by removing all conjuncts containing a specific variable from the query expression. It is easy to see that successively removing conjuncts from the query leads to a sequence of queries with the desired properties.

The main question that arises when adopting the variable elimination approach is the order in which the variables should be removed from the query. This order is partially constrained by the dependencies between the different variables. Removing the wrong variable first can break these dependencies and make remaining conjuncts useless. Looking at the example query this would happen if we first removed the variable Y. In this case the conjunct $V = \mathit{mecklenburg}$ would be isolated, because the variable V only occurred in the removed conjuncts that connected it to the answer variable. In order to avoid breaking dependencies

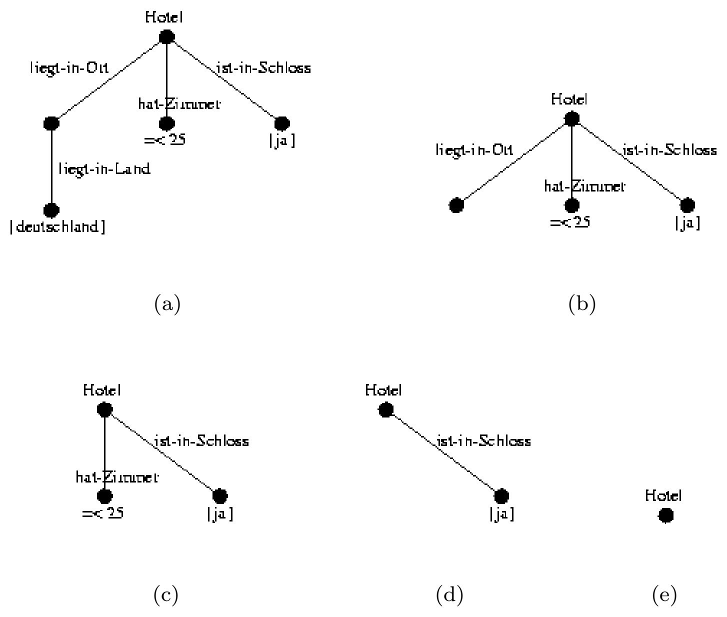


FIGURE 2. A possible sequence of query-graphs

when removing conjuncts, we can use the query graph of the query to be relaxed (compare definition 8) as it explicates existing dependencies. In the query-graph dependencies between variables are represented by arcs between nodes. Therefore we have to ensure that the query graphs remains connected when removing the node that represents the variable we want to eliminate. Obviously, this is only the case if we eliminate variables that correspond to leaf nodes in the graph. Figure 2 illustrates the successive elimination of the variables V, Y, Z and W from the example query, showing the corresponding sequence of query graphs.

4.2. Guided Elimination

The major drawback of the variable elimination heuristic as explained so far is the high number of arbitrary choices that still exist in the order of elimination. More specifically, whenever the query tree has more than one leaf node, we have no strategy yet to decide which one to eliminate. In general, there are many possibilities for defining ordering heuristics, based on:

1. The nature of the domain
2. The preferences of the user
3. The task to be solved

As our approach does not aim at a specific domain, user or task, we will have to rely on rather general heuristics being aware that they will never be optimal. In our case, the only information we can use to decide on an elimination order is the existence of local mappings that relate the query vocabulary to the shared one that is actually used to compute the answer. The general idea is that we would rather drop conjuncts that represent concepts or relations without a suitable mapping into the shared ontology, because it can never be satisfied by any object classified according to that ontology. We have seen that for the case of concepts, we can often find a suitable approximation even if there is no direct counterpart in the shared ontology. Therefore, we focus on conjuncts representing relations and eliminate such variables first that are constrained by a relational conjunct that has no direct mapping to the shared ontology. The effect of this strategy is illustrated in the next section where we describe some experiments with approximating concepts and relaxing queries in a case study.

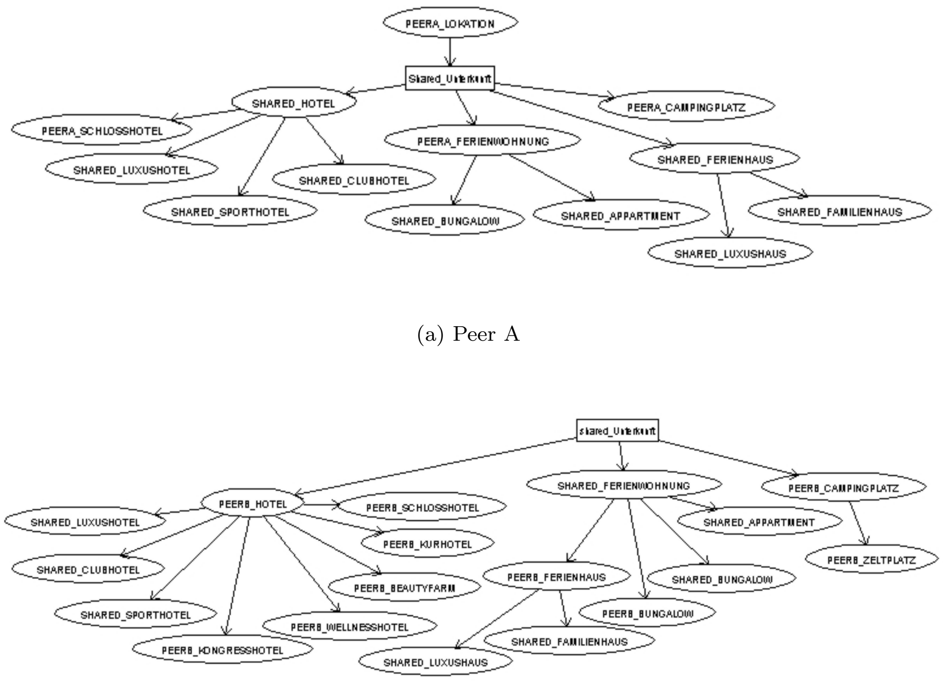
5. Examples from a Case Study

We performed a first case study in order to validate the methods described in this paper. The case study is based on three different ontologies in the domain of tourism. The ontologies are available in the DAML ontology library (`www.daml.org`) and have been created by independent groups of students at the university of Karlsruhe. All ontologies aim at describing the conceptualization of an internet site that is advertising tourism in north-east Germany. All ontologies contain information about accommodation, tourist attractions and transportation facilities. While sharing these general topics, the different ontologies describe them in a very different way focusing on different parts of the overall domain. We chose these ontologies, because they very closely resemble the situation we expect in a peer-to-peer network, where peers model information about the same domain in different ways.

In the course of our case study, we imported the ontologies, each containing about 300 classes and 50 to 70 relations into an ontology editor using some syntactic transformations. We then analyzed the ontologies and created about 150 obvious mappings, mainly between classes that have exactly the same name and between classes where one name is the plural form of the other. Based on these mappings we computed two overlapping class hierarchies consisting of about 600 classes each. These hierarchies served as the basis for evaluating our class approximation and query relaxation techniques. In the following, we describe examples of class approximation and of query relaxation with respect to these hierarchies.

As an example of class approximation we use the class 'Ferien-Wohnung' (a flat used as accommodation during holidays). The relevant parts of the hierarchies can be seen in figure 5. We can see that classes from private (identified by the prefix 'PEERA' or 'PEERB' respectively and shared (identified by the prefix 'SHARED') ontologies occur in both hierarchies.

The approximations we are interested in are the direct sub- and super-classes of our example class form the shared ontology. We can see in the figure that these are: 'Bungalow' and 'Appartment'. If we look at the view of PeerB on the world we see that also the class 'Ferienhaus' (house used during holiday) would fall under this category. While this result is not completely true, because houses are not flats, it still serves the purpose very well, because all classes describe accommodations that are reasonable replacements in the case that no flat is available.



(a) Peer A

(b) Peer B

FIGURE 3. The Views of two different peers on the same domain

If we determine the upper approximation of the example class, we get the general class 'Unterkunft' (accommodation). Our method now determines all instances of this general class to be potential members of the example class. Besides the members of the already mentioned classes, this also includes objects that are members of the classes 'Hotel' and 'Campingplatz' (camp-site) in the view of the answering peer B. We see, that these results are still closely related to the example classes, because they are all accommodations mainly used during holiday, however, hotels and camp-sites are not really the kind of answer the user would assume to get when asking for a flat. Still, returning hotels and camp-sites as answers to a query for a flat is still better than not returning any result, because the user might want to change her choice in favor of other preferences (e.g. the location).

As an example for query relaxation, we take the example query introduced in equation 5. If we transform this query in to a class expression (equation 6) and classify it into the overall class hierarchy of the case study, it end up as a sub-class of 'Schlosshotel' (castle acting as a hotel). Computing the answer to the query we get an empty set, because there are no instances of 'Schlosshotel' satisfying all properties of the query class. Using the upper bound, however, we already get the members of the class 'Schlosshotel'. Looking at the ontologies in the case study, we see that none of the ontologies except for one the query is based on contains information about the number of rooms of a hotel which makes it impossible to prove that a specific Hotel is an answer to the query. As a response to this observation, the query is relaxed by removing the restriction on the number of rooms. This leads to a situation, where we already get some definite results, namely those members of the class 'Schlosshotel' that satisfy the requirement of being in the federal state of Mecklenburg. Note that this provides us with a better result than the use of the upper bound, because we already have a pre-selection of results according to the geographic criterion.

The ability to retrieve relevant information using this second query relied on the fact that the ontology describing the information defines the class 'Schlosshotel' as the set of all hotels for which the property 'liegt-in-schloss' (is located in a castle) is true. We were able to use this implicit information about the specific relation in order to retrieve information without having an explicit assertion stating that a hotel has this specific property. In a case where the ontology does not contain the necessary information, we would still get no results for the relaxed query, because the property 'liegt-in-Schloss' is not satisfied by any information item. In this case we can again use the upper bound for answering the query, which would now be the class 'Hotel'. Consequently, we would get all Hotels as potential answers. Again, this result is too general, as we want to preserve at least the geographic constraint. A solution is to further relax the query by removing the 'liegt-in-Schloss' property from the query. The resulting query will match all Hotels in the federal estate of Mecklenburg.

6. Conclusions

The existing data integration technology is based on the assumption that it is possible to define a virtual global schema. A query is therefore posed to this schema and then suitably translated in local queries to the information bases being integrated. As hinted in this paper and more extensively argued elsewhere, this approach hardly scales to a highly dynamic P2P network.

In this paper we have proposed a novel approach based on the following key ideas:

1. We shouldn't think in terms of a global schema but, rather, in terms of independent autonomous nodes which, at run time, depending on the query, provide local answers which must then be integrated.
2. In most cases, the system will not be able to provide the best possible answer. It is more an issue of providing an answer which is good enough.

The proposed approach exploits ontologies as a conceptual "high level" schema which allows to hide local implementation details, and to exchange information using the vocabulary defined by the ontologies themselves. In this setting, good enough answers are obtained by posing queries to local ontologies, by allowing queries to be propagated by using inter- ontology mappings, and by using approximation techniques in order to avoid the problem of very sparse mappings.

We are only at the beginning and a lot of work is still to be done before integrating Semantic Web and P2P network technology. We list below some of the open problems we can foresee:

Foundations. Once we assume that we have no global schema we must also assume, among other things, that we no longer have a single domain of interpretation (a single set of models). To take this into account we must define a new semantics which allows for multiple interpretation domains, and for mappings which tell how ontology elements, and also domain elements, are mapped. In this framework, queries, query answers, and the mappings defined in this paper can be formally characterized and various (partial) completeness and correctness results can be provided (characterizing, for instance, to which extent, a query has been answered). Some preliminary ideas in this direction can be found in [2].

Semantic Routing. In a P2P network, in most if not all cases, a node has very little knowledge of its peers, and this knowledge becomes obsolete very quickly. Before worrying about how to answer a query (the problem we have mainly dealt with in this paper) a node has to worry about which nodes should be queried. This requires some mechanisms for peer discovery that, in general, will have again to deal with the problem of semantic heterogeneity (each node will advertise itself using some local vocabulary). A preliminary description of a possible solution has been provided in [5], which proposes "Interest Groups" as a way of collecting peers having knowledge about same or similar topics and a "locally centralized" mechanism for handling such groups.

Quality of Answers. We have already given up the hope for complete or, more simply, correct answers. But how do we judge when an answer is good answer? Whether an answer is good enough depends on many things: what the user wants, the status of the network, its connectivity, the topic of the query, and so on. If we want to make the approximation independent of human judgement, we have to provide formal quality measures that can be used to judge the quality of an approximation and provide a basis for deciding between alternative approximations. A promising proposal for such a quality measure is described in [13]. A particular question concerns the trade-off between the accuracy of an approximation and the complexity of computing it. One of the strengths of our approach is that determining the next approximation does not require logical reasoning itself due to the structure based relaxation criterion. It is very likely that we could further improve the accuracy of the approximation by using logical reasoning for finding alternatives for class and relation names rather than removing them from the query.

References

[1] S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semi-structured and structured data sources. *SIGMOD Records*, 28(1):54–59, March 1999.

[2] P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. Technical Report DIT-02-0013, Depertment of Information Technologies, University of Trento, 2002. Also appears in Proceedings of Web DB 2002.

[3] D. Calvanesea, G. De Giacomo, and M. Lenzerini. Description logics for information integration. In *Computational Logic: From Logic Programming into the Future*, Lecture Notes in Computer Science. Springer Verlag, 2001.

[4] F. M. Donini, M. Lenzerini, D. Nardi, , and A. Schaerf. Reasoning in description logics. In G. Brewka, editor, *Principles of Knowledge Representation*, Studies in Logic, Language and Information, pages 193–238. CSLI Publications, 1996.

[5] F. Giunchiglia and I. Zaihrayeu. Making peer databases interact - a vision for an architecture supporting data coordination. In *CIA 2002: Cooperative Information Agents*, Lecture Notes in AI. Springer, 2002.

[6] A. Gomez-Perez and O. Corcho. Ontology langauges for the semantic web. *IEEE Intelligent Systems*, January/February:54–60, 2002.

[7] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001.

[8] I. Horrocks. The FaCT system. In H. de Swart, editor, *Automated Reasoning with Analytic Tableaux and Related Methods: International Conference Tableaux'98*, number 1397 in Lecture Notes in Artificial Intelligence, pages 307–312. Springer-Verlag, Berlin, May 1998.

[9] I. Horrocks and S. Tessaris. A conjunctive query language for description logic aboxes. In *AAAI/IAAI*, pages 399–404, 2000.

[10] M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.

[11] J. Madhavan, P. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. In *Eighteenth National Conference on Artificial Intelligence (AAAI'2002)*, Edmonton, Canada, 2002.

[12] M. Parameswaran, A. Sursala, and A. Winston. P2p networking: An information sharing alternative. *IEEE Computing*, 34(7), 2001.

[13] M. Peim, E. Franconi, and N. Paton. Estimating the quality of answers when querying description logic ontologies. *Data and Knowledge Engineering*, 2002. (submitted).

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

[15] B. Selman and H. Kautz. Knowledge compilation and theory approximation. *Journal of the ACM*, 43(2):193–224, March 1996.

[16] H. Stuckenschmidt. Approximate information filtering with multiple classification hierarchies. *International Journal of Computational Intelligence and Applications*, 2002. to appear.

[17] H. Stuckenschmidt. *Ontology-Based Information Sharing in Weakly-Structured Environments*. PhD thesis, Faculty of Sciences, Vrije Universiteit Amsterdam, 2002.

[18] H. Stuckenschmidt, F. van Harmelen, and F. Giunchiglia. Query processing in ontology-based peer-to-peer systems. Technical report, Department of Department of Information and Communication Technology, University of Trento, November 2002.

[19] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S.Huebner. Ontology-based integration of information - a survey of existing approaches. In *Ontologies and Information Sharing*, number 47, pages 108–117, Seattle, USA, August 2001.