Querying Articulated Sources

Tzitzikas, Yannis; Meghini, Carlo

Published in:
Proceedings of the third International Conference on Ontologies, Databases and Applications of Semantics for Large Scale Information Systems, ODBASE'2004

Publication date:
2004

Querying Articulated Sources

Carlo Meghini[1] and Yannis Tzitzikas[2]

[1] ISTI – CNR, Pisa, Italy meghini@isti.cnr.it
[2] Institut d'Informatique, University of Namur, Belgium ytz@info.fundp.ac.be

Abstract. In this study we address the problem of answering queries over information sources storing objects which are indexed by terms arranged in a taxonomy. We examine query languages of different expressivity and sources with different kinds of taxonomies. In the simplest kind, the taxonomy includes just term-to-term subsumption links. This case is used as a basis for further developments, in which we consider taxonomies consisting of term-to-queries links. An algorithm for query evaluation is presented for this kind of taxonomies, and it is shown that the addition of negation to the query language leads to intractability. Finally, query-to-query taxonomies are considered.

1 Introduction

In semantic-based retrieval on peer-to-peer (P2P) networks, the language that can be used for indexing the domain objects and for formulating queries, can be either *free* (e.g natural language) or *controlled*, i.e. object descriptions and queries may have to conform to a specific vocabulary and syntax. The former case resembles distributed Information Retrieval (IR) and is applicable when the domain objects have a textual content (e.g. [1, 2]). In the latter case, the objects of a peer are indexed according to a specific conceptual model (e.g. relational, object-oriented, logic-based, etc), and content searches are formulated using a specific query language. An approach falling into this category, in which the objects of the domain are indexed in terms of *taxonomies* and *inter-taxonomy mappings* are employed for bridging the inevitable naming, granularity and contextual heterogeneities that may exist between the taxonomies of the peers, was proposed in [3]. The difference between the P2P architecture and the classical two-tiered mediator approach (like the one presented in [4]) is that in a P2P system the mappings between the peers may lead to cyclic dependencies between the query evaluation tasks of the peers. Such cases require special treatment in order to avoid endless query evaluation and to optimize the evaluation of queries. The work presented in [5] gave the foundations of query answering in this kind of systems and presented four algorithms for query evaluation. However, that work considered a very simple form of articulated source, namely one whose articulations relate just terms, and a negation-free query language.

In this paper, we make a step forward, by considering term to query articulations, that is articulations relating queries of one source to terms in another source, and provide an algorithm for handling query evaluation in this context.

The algorithm is then extended to the case of queries including negation, borrowing the semantics from datalog, by establishing a mapping from a source to a datalog program. We then consider term to query articulations whose queries include negation, and show that query evaluation becomes a coNP-hard problem. We finally move on to consider query to query articulations, showing that the usage of negation-free DNF queries in articulations make the object retrieval problem intractable.

The next two Sections lay down the basic framework. With Section 4, we move towards more sophisticate scenarios, starting with the addition of negation to the language for querying simple sources. Section 5 deals with term to query articulations, while Section 6 considers query to query articulations. Related work is reported in Section 7. For reasons of space, we have included in the paper only the most important proofs.

2 Simple Sources

Let Obj denote the set of all objects of a domain common to several information sources.

Definition 1 (Simple Source). A *simple source* S is a pair $S = (A, I)$ where

- A, the *taxonomy,* is a pair $(T, \preceq)$ where T, the *terminology,* is a finite and non-empty set of names, or *terms*, and $\preceq$ is a reflexive and transitive relation over T, modeling *subsumption* between terms.
- I, the *interpretation,* is a total function $I : T \rightarrow 2^{Obj}$ that associates each term in the terminology with a set of objects. □

Figure 1 presents the taxonomy of a simple source. For readability, only and the transitive reduction of the subsumption relation is given, leaving out reflexive and transitive relationships.

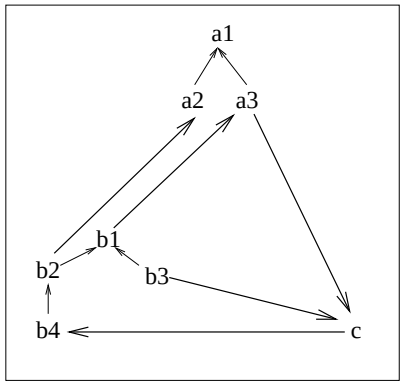

Fig. 1. A taxonomy

Not all interpretations of a source are the same; those that satisfy the subsumption relationships, better reflect the application semantics and are therefore factored out as *models,* following a common terminology.

Definition 2 (Model). An interpretation I of a terminology T is a *model* of a taxonomy $A = (T, \preceq)$ if $t \preceq t'$ implies $I(t) \subseteq I(t')$. Given two interpretations I, I' of the same terminology T, I is less than or equal to I', in symbols $I \leq I'$, if $I(t) \subseteq I'(t)$ for each term $t \in T$. An interpretation J of a terminology T is a *model* of a simple source $S = (A, I)$ if it is a model of A and $I \leq J$. □

To query a simple source, we next introduce a query language allowing negation-free Boolean combinations of terms as queries. These queries are captured in DNF expressions.

Definition 3 (Query). Let T be a terminology. The *query language* associated to T, $\mathcal{L}_T$, is the language defined by the following grammar, where t is a term of T : $q ::= d \mid q \vee d$, $d ::= t \mid t \wedge d$. An instance of q is called a *query,* while an instance of d is called a *disjunct*. □

The semantics of the query language maps each query into a set of objects, based on a specific interpretation of the underlying terminology.

Definition 4 (Extension). Given a simple source $S = (A, I)$, where $A = (T, \preceq)$, and a query $q \in \mathcal{L}_T$, the *extension of q in I,* q^I, is defined as follows:

1. $(q \vee d)^I = q^I \cup d^I$
2. $(d \wedge t)^I = d^I \cap t^I$
3. $t^I = I(t)$. □

Since the function $\cdot^I$ is an extension of the interpretation function I, we will abuse notation by using the latter in place of the former.

Definition 5 (Answer). Given a simple source $S = (A, I)$, the *answer of q in S,* $ans(q, S)$, is given by:

$$ans(q, S) = \{o \in Obj \mid o \in J(q) \text{ for all models } J \text{ of } S\}$$

that is, the set of objects that are in the extension of q in all the models of A which are greater than I. □

We can now state query evaluation.

Proposition 1. For all simple sources $S = (A, I)$, where $A = (T, \preceq)$, and queries $q \in \mathcal{L}_T$, $ans(q, S)$ is given by:

1. $ans(q \vee d, S) = ans(q, S) \cup ans(d, S)$,
2. $ans(d \wedge t, S) = ans(d, S) \cap ans(t, S)$,
3. $ans(t, S) = \bar{I}(t)$

where $\bar{I}(t) = \bigcup\{I(s) \mid s \preceq t\}$ is the unique minimal model $\bar{I}$ of the simple source $S = (A, I)$.

Proof: We first show that $\bar{I}$ is the unique minimal model of the source S. To this end, it must be proved that (a) $\bar{I}$ is a model of A; (b) $I \leq \bar{I}$; and (c) $\bar{I}$ is the smallest model for which (a) and (b) hold. (a) $t \preceq t'$ implies $\{s \mid s \preceq t\} \subseteq \{s \mid s \preceq t'\}$, hence $\bigcup\{I(s) \mid s \preceq t\} \subseteq \bigcup\{I(s) \mid s \preceq t'\}$, *i.e.* $\bar{I}(t) \subseteq \bar{I}(t')$. Thus $\bar{I}$ is a model of $(T, \preceq)$. (b) trivially follows from the definition of $\bar{I}$ and from the reflexivity of $\preceq$. To see (c), let I' be a model of $(T, \preceq)$ which is greater than I. We prove that $\bar{I} \leq I'$. By the definition of $\bar{I}(t)$, if $o \in \bar{I}(t)$ then $o \in I(s)$ for a term s such that $s \preceq t$. Then $o \in I'(t)$ too because I' is a model of T. We conclude that for every $o \in \bar{I}(t)$ it holds $o \in I'(t)$, which means that $\bar{I} \leq I'$. As for the rest of the Proposition, let us start from the last clause. $o \in ans(t, S)$ implies $o \in \bar{I}(t)$, since $I \leq \bar{I}$, so $o \in ans(t, S)$ implies $o \in \bar{I}(t)$. Conversely, $o \in \bar{I}(t)$ implies, by Proposition 1, that $o \in J(t)$, for all models J of A such that $I \leq J$, *i.e.* $o \in ans(t, S)$. As for the second clause:

$$\begin{aligned} ans(d \wedge t, S) &= \{o \in Obj \mid o \in (d \wedge t)^J, \forall \text{ mod.} J \text{ of } S\} \\ &= \{o \in Obj \mid o \in d^J \cap t^J, \forall \text{ mod.} J \text{ of } S\} \\ &= \{o \in Obj \mid o \in d^J, o \in J(t), \forall \text{ mod.} J \text{ of } S\} \\ &= ans(d, S) \cap ans(t, S). \end{aligned}$$

The argument for the first clause is analogous. □

We call $\bar{I}$ the model of A *generated* by I. The procedure δ_t, presented in Figure 2, computes the model generated by a given interpretation on a term x implementing the following definition, provably equivalent to the one in the last Proposition:

$$\bar{I}(x) = I(x) \cup \bigcup\{\bar{I}(v) \mid v \preceq^r x\}$$

where $\preceq^r$ is the transitive reduction of the subsumption relation $\preceq$, encoded in the graph G_A. $\bar{I}(c)$ for the taxonomy in Figure 1 is computed by invoking $\delta_t(c, \{c\})$, and yields $I(c) \cup I(a3) \cup I(b1) \cup I(b2) \cup I(b4) \cup I(b3)$.

```
procedure δ_t (t : term ; A : set of terms);
1. begin
2. R ← I(t)
3. for each edge ⟨u, t⟩ in G_A do
4.   if u ∉ A then begin
5.     A ← A ∪ {u}
6.     R ← R ∪ δ_t(u, A)
7.     end
8. return R
9. end
```

Fig. 2. The procedure δ_t

3 Networks of Articulated Sources

Articulated sources are simple sources whose terms have subsumption relationships with the terms of other terminologies. These inter-terminology relationships are called articulations, to distinguish them from those within single taxonomies, which are of an intra-terminology nature. Formally,

Definition 6 (Articulated Source). An *articulation* $\preceq_{ij}$ from a terminology T_i to a terminology T_j, is any non-empty set of relationships $t_j \preceq_{ij} t_i$ where $t_i \in T_i$ and $t_j \in T_j$. An *articulated source* M over $k \geq 1$ disjoint terminologies $T_1, ..., T_k$, is a pair $M = (S_M, R_M)$, where: $S_M = (A_M, I_M)$ is a simple source such that $A_M = (T_M, \preceq_M)$ and T_M is disjoint from $T_1, ..., T_k$; and R_M is a set $R_M = \{a_{M,1}, ..., a_{M,k}\}$, where for all $i \in [1, k]$, $a_{M,i}$ is an *articulation* from T_M to T_i. □

In what follows we will tacitly consider only articulated sources over disjoint terminologies. An articulated source M with an empty interpretation, *i.e.* $I_M(t) = \emptyset$ for all $t \in T_M$, is also called a *mediator.*

Definition 7 (Network). A *network of articulated sources,* or simply a *network*, N is a non-empty set of sources $N = \{S_1, \ldots, S_n\}$, where each source S_i is either simple, or is articulated over the terminologies of the sources in a proper, non-empty subset of $N \setminus \{S_i\}$. □

Figure 3 shows a network of 3 articulated sources. Articulations are highlight by a surrounding circle.

One way of interpreting a network is to view it as a simple source which happens to be distributed along several simple sources, each dealing with a specific sub-terminology of the network terminology. The relationship between Figures 1 and 3 evidently suggests this view. The global source can be logically re-constructed by removing the barriers which separate local sources, as if (virtually) collecting all the network information in a single repository. The notion of *network source,* defined next, captures this interpretation of a network.

Definition 8 (Network source). The *network source* S_N of a network of articulated sources $N = \{S_1, \ldots, S_n\}$, is the simple source $S_N = (A_N, I_N)$, where $A_N = (T_N, \sqsubseteq)$ and:

$$T_N = \bigcup_{i=1}^{n} T_i, \qquad I_N = \bigcup_{i=1}^{n} I_i, \qquad \sqsubseteq = (\bigcup_{i=1}^{n} \sqsubseteq_i)^*$$

where $\sqsubseteq_i$ is the *total subsumption* of the source S_i, given by the union of the subsumption relation $\preceq_i$ with all articulations of the source, that is:

$$\sqsubseteq_i = \preceq_i \cup a_{i,1} \cup \ \ldots \ \cup a_{i,n}$$

and A^* denotes the transitive closure of the binary relation A. A *network query* is a query over T_N. □

Note that this global simple source does not pre-exist. It emerges in a bottom-up manner by the articulations of the peers. This is one difference that distinguishes peer-to-peer systems from federated distributed databases.

Following the model developed so far, the answer to a network query q, or *network answer,* is given by $ans(q, S_N)$, which relies on the model of A_N generated by I_N, that is, for each term t in T_N :

$$\bar{I}_N(t) = \bigcup\{\ I_N(t') \mid t' \sqsubseteq t\}.$$

In order to evaluate a network query, a distributed process is required, which uses the query evaluators on the local simple sources as sub-processes. The topology of this global process strictly reflects that of the network subsumption relation $\sqsubseteq$. For instance, in order to evaluate the query a_1 in the network of Figure 3, the query a_1 must be evaluated on source S_1, b_1 must be evaluated on source S_2, c on source S_3, and so on, following articulations backwards, so as to compute $\bar{I}_N(a_1)$. In order to avoid an endless query evaluation, a 2-level cycle management is required: local query evaluators must take care of the intra-terminology cycles (typically, by using the procedure δ_t), while the global query evaluator must properly handle the inter-terminology cycles, *i.e.*the cycles in the network subsumption relation $\sqsubseteq$ which involve at least one articulation. For simple sources, we have studied the problem elsewhere [5].

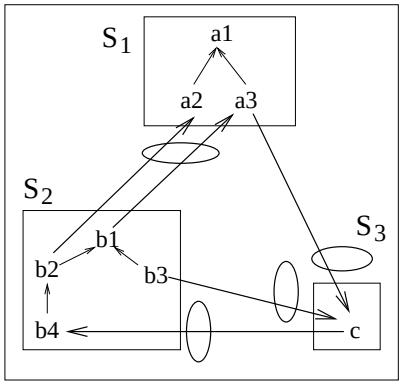

Fig. 3. A network of articulated sources

4 Adding negation in queries

We now extend the query language by allowing negation. That is, we consider the queries belonging to the language $q ::= \ t \mid q \wedge q' \mid q \vee q' \mid \neg\ q$. Also these queries can be translated into DNF form, yielding the language defined next.

Definition 9 (Extended Query). Let T be a terminology. An *extended query* over T is any string derived by the following grammar, where t is a term of T: $q ::= d \mid q \vee d$ where a disjunct d is given by $d ::= l \mid l \wedge d$, and l is a *literal,* defined as $l ::= t \mid \neg t$. We call the language so defined $\mathcal{L}_T^{\neg}$. □

The extension of a negative literal in an interpretation I of T is defined, in the obvious way, as follows: $I(\neg t) = Obj \setminus I(t)$, while the notion of an answer remains unchanged, that is $o \in ans(q, S)$ iff o is in the extension of q in all models of the source S.

By extending the model in this apparently intuitive way, however, a negative literal in a query is equivalent to the *false* clause, because there is not enough information in the taxonomy of a source to support a negative fact. In order to derive an intuitive and, at the same time, logically well-grounded evaluation procedure for extended queries, we need an alternative query semantics (*i.e. ans*). In order to define it, let us consider a logical reformulation of the problem in terms of datalog.

Intuitively, the translation from a simple source to a datalog program should be straightforward: terms are unary predicate symbols, as they are interpreted by sets of objects; consequently, each subsumption relationship is mapped into a rule on the predicate symbols corresponding to the involved terms; and the interpretation of each term is mapped into a set of ground facts on the corresponding predicate symbol. In so doing, however, there could be predicate symbols occurring both in rule heads and in facts. In datalog terms, these predicate symbols would be both intensional and extensional, and this is not allowed by the datalog syntax. This problem is solved by mapping each term t_i into two predicate symbols: (a) an extensional one, denoted $\mathtt{C_i}$, representing the interpretation of t_i, *i.e.* $I(t_i)$; and (b) an intensional one, denoted $\mathtt{Y_i}$, representing t_i in the rules encoding the subsumption relation. The obvious connection between $\mathtt{C_i}$ and $\mathtt{Y_i}$ is that all facts expressed via the former are also true of the latter, and this is captured by stating a rule (named "extensional" below) of the form $\mathtt{C_i}(x) \rightarrow \mathtt{Y_i}(x)$ for each term t_i.

Notice that not every subsumption relationship needs to be mapped into a rule: since reflexivity and transitivity are embodied into logical consequence, only the transitive reduction $\preceq^r$ of the subsumption relation needs to be encoded into the program.

Definition 10 (Source program). Given a simple source $S = (A, I)$, where $A = (T, \preceq)$, the *source program* of S is the set of clauses P_S given by $P_S = TR_S \cup ER_S \cup F_S$, where:

- $TR_S = \{\mathtt{Y_i(x)} :- \mathtt{Y_j(x)} \mid t_j \preceq^r t_i\}$ are the *terminological rules* of P_S;
- $ER_S = \{\mathtt{Y_i(x)} :- \mathtt{C_i(x)} \mid t_i \in T\}$ are the *extensional rules* of P_S;
- $F_S = \{\mathtt{C_i(o)} \mid o \in I(t_i)\}$ are the *facts* of P_S, stated in terms of constants $\mathtt{o}$ which are one-to-one with the elements of Obj (unique name assumption). □

Next, we translate queries in the language $\mathcal{L}_T$.

Definition 11 (Query program). Given a query $q \in \mathcal{L}_T$ to a simple source $S = (A, I)$, where $A = (T, \preceq)$, the *query program* of q is the set of clauses P_q given by:

$$\{\mathtt{q(x)} :- \mathtt{Y_1(x)}, \ldots, \mathtt{Y_k(x)} \mid t_1 \wedge \ldots \wedge t_k \text{ is a disjunct of } q\}.$$

where $\mathtt{q}$ is a new predicate symbol. □

In order to show the quivalence of the original model with its datalog translation, we state the following:

Proposition 2. For each simple source $S = (A, I)$, where $A = (T, \preceq)$, and query $q \in \mathcal{L}_T$ to S, $ans(q, S) = \{o \in Obj \mid P_S \cup P_q \models \mathtt{q(o)}\}$. □

Let us consider this mapping in light of the new query language. For a source $S = (A, I)$, the source program P_S remains a pure datalog program, while the query program P_q of any query q agaist S becomes:

$$\{\mathtt{q(x)} :- \mathtt{L_1(x)}, \ldots, \mathtt{L_k(x)} \mid t_1 \wedge \ldots \wedge t_k \text{ is a disjunct of } q\}$$

where each $\mathtt{L_i}$ can now be either $\mathtt{Y_i}$ or $\neg\ \mathtt{Y_i}$.

We can now re-phrase in logical terms the problem with negative literals in queries stated at the beginning of this Section, namely that negative facts cannot be logical consequences of a datalog program, hence a query evaluation procedure based on logical consequence, would treat negative literals as *false* clauses. To circumvent this problem, while retaining an intuitive query-answering behaviour, the notion of logical consequence is extended so as to allow the inference of negative literals. In datalog, the extension which is typically used is an approximation of CWA, and can be characterized either procedurally, in terms of program stratification, or declaratively, in terms of perfect model. We will adopt the former characterization.

In fact, P_q is a datalog$^\neg$ program, and so is the program $P_S \cup P_q$. The latter program is stratified, by the level mapping l defined as follows:

$$l(pred) = \begin{cases} 1 \text{ if } pred \text{ is } \mathtt{q} \\ 0 \text{ otherwise} \end{cases}$$

It follows that $P_S \cup P_q$ has a minimal Herbrand model M_S^q given by ([6]) the least fixpoint of the transformation $T'_{P_q \cup M_{P_S}}$ where M_{P_S} is the least Herbrand model of the datalog program P_S, and T'_P is the (obvious) extension to datalog$^\neg$ of the T_P operator, on which the standard semantics of pure datalog is based. The model M_S^q is found from M_{P_S} in one iteration since only instances of $\mathtt{q}$ are added at each iteration, and $\mathtt{q}$ does not occur in the body of any rule. The following definition establishes an alternative notion of answer for queries including negation.

Definition 12 (Extended answer). Given an extended query q to a simple source $S = (A, I)$, the *extended answer to q in S,* denoted $\varepsilon(q, S)$, is given by: $\varepsilon(q, S) = \{o \in Obj \mid M_S^q \models \mathtt{q(o)}\}$ □

We conclude by showing how extended answers can be computed.

Proposition 3. For each simple source $S = (A, I)$, where $A = (T, \preceq)$, and query $q \in \mathcal{L}_T^{\neg}$, $\varepsilon(q, S)$ is given by:

1. $\varepsilon(q \vee d, S) = \varepsilon(q, S) \cup \varepsilon(d, S)$,
2. $\varepsilon(l \wedge d, S) = \varepsilon(l, S) \cap \varepsilon(d, S)$,
3. $\varepsilon(t, S) = \bar{I}(t)$,
4. $\varepsilon(\neg t, S) = Obj \setminus \varepsilon(t, S)$. □

From a practical point of view, computing $\varepsilon(\neg t_1 \wedge \ldots \wedge \neg t_k)$ requires computing:

$$Obj \setminus (\bar{I}(t_{i_1}) \cup \ldots \cup \bar{I}(t_{i_k}))$$

which in turn requires knowing Obj, *i.e.* the whole set of objects of the network. As this knowledge may not be available, or may be too expensive to obtain, one may want to resort to a query language making a restricted usage of negation, for instance by forcing each query disjunct to contain at least one positive term.

5 Term to query articulations

Here we study the more general case where an articulation can contain subsumption relationships between terms and *queries*. We call such articulations *term-to-query* (t2q), to be distinguished from the articulations introduced previously, which we term *term-to-term* (t2t) articulations. t2t articulations are clearly special cases of t2q articulations.

First, we introduce the basic building block of t2q articulations, that is subsumption relationships between queries and terms.

Definition 13 (Extended source). An *extended taxonomy* is a pair $(T, \preceq_e)$ where T is a terminology and $\preceq_e \subseteq (\mathcal{L}_T \times \mathcal{L}_T)$, reflexive and transitive. An *extended source* S is a pair (A, I), where A is an extended taxonomy $(T, \preceq_e)$ and I is an interpretation of T. □

Notice that since a term is a query, an extended taxonomy does in fact extend a taxonomy, by allowing subsumption relationships also between disjunctions of conjunctions of terms (*i.e.*, non-term queries) and terms. Figure 4 presents the taxonomy of an extended source.

Next, we introduce the notion of model of an extended source.

Definition 14 (Model). An interpretation I of a terminology T is a *model* of an extended taxonomy $(T, \preceq_e)$ if $q \preceq_e t$ implies $I(q) \subseteq I(t)$. An interpretation J of a terminology T is a *model* of an extended source $S = (A, I)$ if it is a model of A and $I \leq J$. □

The answer to a query $q \in \mathcal{L}_T$ to an extended source $S = (A, I)$, is the same as that for the 2t2 case, *i.e.*:

$$ans(q, S) = \{o \in Obj \mid o \in J(q) \text{ for all models } J \text{ of } S\}.$$

The analogous of Proposition 1 is the following.

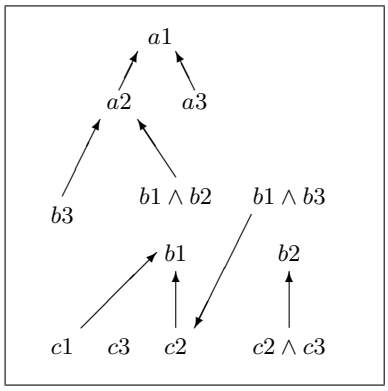

Fig. 4. An extended taxonomy

Proposition 4. For all extended sources $S = (A, I)$, where $A = (T, \preceq_e)$, and queries $q \in \mathcal{L}_T$, $ans(q, S)$ is given by:

1. $ans(q \vee d, S) = ans(q, S) \cup ans(d, S)$,
2. $ans(d \wedge t, S) = ans(d, S) \cap ans(t, S)$,
3. $ans(t, S) = \bar{I}_e(t)$

where $\bar{I}_e(t) = \bar{I}(t) \ \cup \ \bigcup\{\bar{I}(q) \mid q \preceq_e t$ and for no $u \in T$, $q = u\}$ is the unique minimal model of S. □

In order to perform query evaluation on an extended source, our starting point is the method for the t2t case, in which the interpretation $\bar{I}(t)$ for each query term t is computed by the procedure δ_t, which navigates the graph G_A. In order to have the same kind of navigation for an extended source, the graph representing $\preceq_e^r$, having either terms or queries as nodes such as the one in Figure 4, is unsuitable. To see why, let us consider the following equivalent rewriting of $\bar{I}_e(t)$

$$\bar{I}_e(t) = \bigcup\{I(s) \mid s \preceq_e t\} \ \cup \\ \bigcup\{\bar{I}(q) \mid q \preceq_e t \text{ and for no } u \in T,\ q = u\}.$$

According to this expression, in order to compute $\bar{I}_e(t)$ one starts from t and moves backward to find all terms and queries that are reachable through subsumption links; when a node with a term s is found, the extension $I(s)$ must be fetched, and then search proceeds normally; but when a node with a query q is reached, $\bar{I}(q)$ must be computed, and this requires to "jump" to the terms composing q. In order to avoid this problem, we use an hypergraph to represent the taxonomy of an extended source.

In an hypergraph, an edge can connect a node to an arbitrary subset of nodes, and is therefore called hyperedge. In order to generate the hypergraph representing $\preceq_e^r$, we first trasform $\preceq_e^r$ into the equivalent relation $\preceq'$ by replacing each

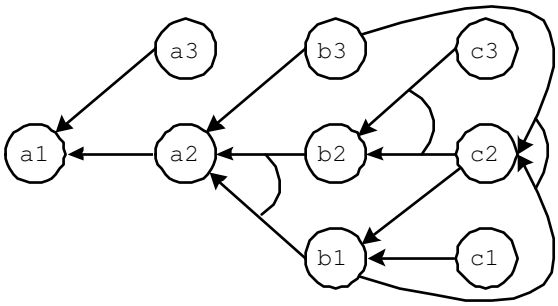

Fig. 5. The hypergraph of the taxonomy in Figure 4

relationship $(q_1 \vee \ldots \vee q_k, t)$ in $\preceq_e^r$, with the k relationships $(q_1, t), \ldots, (q_k, t)$. Then, the hypergraph H_A is constructed from $\preceq_e'$ by introducing an hyperedge $\langle \{u_1, \ldots, u_m\}, t \rangle$ if and only if $(u_1 \wedge \ldots \wedge u_m, t) \in \preceq_e'$. Figure 5 shows the hypergraph associated to the taxonomy shown in Figure 4. Hyperedges are represented by joint edges.

The procedure δ_q, presented in Figure 6, computes $\bar{I}_e(t)$ for a given term t, by navigating the hypergraph just introduced. To this end, it must be invoked as:

$$\delta_q(t, \{t\})$$

where the second parameter is the set of terms on the *path* from t to the current term. This set is used to correctly terminate the evaluation in presence of loops in the hypergraph H_A. In fact, the management of the termination condition is one of the two differences between δ_q and δ_t. The other difference concerns the computation of R (line 5 of δ_q), which in the present case must reflect the structure of the considered hyperedge, which in turns reflect the fact that we are dealing with t2q articulations. The reason why termination is checked on the basis of the membership of a term in the path from the original term to the current one, is that a term may belongto several queries, thus simply the fact that the term has been already encountered is not sufficient to decide that the

procedure δ_q (t : **term** ; A : **set of terms**);
1. **begin**
2. $R \leftarrow I(t)$
3. **for each** hyperdge $\langle \{u_1, ..., u_r\}, t \rangle$ in H_A **do**
4. **if** $\{u_1, ..., u_r\} \cap A = \emptyset$ **then**
5. $R \leftarrow R \cup (\delta_q(u_1, A \cup \{u_1\}) \cap \ldots \cap \delta_q(u_r, A \cup \{u_r\})$
6. **return** R
7. **end**

Fig. 6. The procedure δ_q

current hyperedge does not contribute to the result, as instead it was the case for δ_t. Instead, if the current hyperedge h connects the current input term t to a term x belonging in the current path A, then x is being encountered upon computing $\bar{I}_e(x)$, therefore the current hyperedge does not give any contribution to the result. An example of application of δ_q can be found in the appendix.

Let us now proceed to define t2q articulated sources.

Definition 15 (t2q articulated source). A *term-to-query articulation* $\preceq_{ij}$ from a terminology T_i to a terminology T_j, is any nonempty set of relationships $q_j \preceq_{ij} t_i$ where $t_i \in T_i$ and $q_j \in \mathcal{L}_{T_j}$. A *t2q articulated source* M over $k \geq 1$ disjoint terminologies $T_1, ..., T_k$, is a pair $M = (S_M, R_M)$, where: $S_M = (A_M, I_M)$ is an extended source such that $A_M = (T_M, \preceq_M)$ and T_M is disjoint from $T_1, ..., T_k$; and R_M is a set $R_M = \{a_{M,1}, ..., a_{M,k}\}$, where for all $i \in [1, k]$, $a_{M,i}$ is a *t2q articulation* from T_M to T_i. □

Networks of t2q articulated sources (or, simply 2tq networks), are defined in the obvious way.

Definition 16 (t2q network). A *t2q network of articulated sources,* or simply a *t2q network*, N is a non-empty set of sources $N = \{S_1, \ldots, S_n\}$, where each source S_i is either simple, or is a t2q articulated source over the terminologies of a proper, non-empty subset of the sources in $N \setminus \{S_i\}$. □

Figure 7 presents a t2q network consisting of 3 sources.

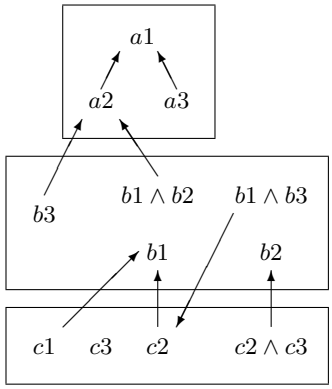

Fig. 7. A t2q network with 3 sources

The source corresponding to a t2q network, defined as in the t2t case (Definition 8), is now an extended source, against which queries can be posed. Figure 7 shows a network of t2q articulated sources, following the same conventions as in Figure 3.

5.1 Adding negation to the taxonomy

If the queries on the left-hand side of articulations have negation, then the network corresponds to a Datalog program with rules that contain negation in their bodies, and it is well known (e.g. see [7]) that such programs may not have a unique minimal model. This is also illustrated by the example shown in Figure 8, in which the interpretation function is also given as term superscript (that is, $I(a2) = I(b2) = \{o\}$, while $I(a1) = I(b1) = \emptyset$).

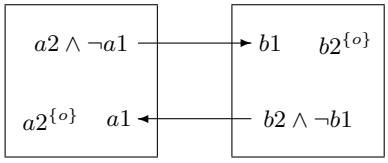

Fig. 8. A network with no unique minimal model

term/query	I	I_a	I_b
$a1$	$\emptyset$	$\{o\}$	$\emptyset$
$a2$	$\{o\}$	$\{o\}$	$\{o\}$
$b1$	$\emptyset$	$\emptyset$	$\{o\}$
$b2$	$\{o\}$	$\{o\}$	$\{o\}$
$b2 \wedge \neg b1$	$\{o\}$	$\{o\}$	$\emptyset$
$a2 \wedge \neg a1$	$\{o\}$	$\emptyset$	$\{o\}$

Table 1. Models of the network shown in Figure 8

Table 1 shows the interpretation I of the network and two interpretations, I_a and I_b, which are both models and minimal. This turns out to be a serious drawback.

Proposition 5. A *neg-extended taxonomy* is a pair $(T, \preceq_e^\neg)$ where T is a terminology and $\preceq_e^\neg \subseteq (\mathcal{L}_T^\neg \times T)$, reflexive and transitive. A *neg-extended source* S is a pair (A, I), where A is a neg-extended taxonomy $(T, \preceq_e^\neg)$ and I is an interpretation of T. Deciding whether an object $o \in Obj$ is in the answer of an extended query q in a neg-extended source S, $o \in ans(q, S)$, is a coNP-hard problem.

The proof is based on the following polynomial reduction from SAT. Let α be a CNF formula of propositional logic over an alphabet V, that is:

$$\alpha = \bigwedge_{i=1}^{n} \alpha_i \qquad \alpha_i = \bigvee_{j=1}^{m_i} l_{ij}$$

where l_{ij} is either a positive literal, that is a letter $v \in V$, or a negative literal, that is $\neg u$ where $u \in V$. We map α into a neg-extended source $S_\alpha = (A_\alpha, I_\alpha)$, where $A_\alpha = (V, \preceq_\alpha)$, and an extended query q_α as follows: let o be any object in Obj; then:

- the query q_α is given by

$$\bigvee \{v_1 \wedge \ldots \wedge v_k \mid \neg v_1 \vee \ldots \vee \neg v_k \text{ is a conjunct in } \alpha\}$$

  If there is no such conjunct $\neg v_1 \vee \ldots \vee \neg v_k$ in α, then let α_1 be $l_1 \vee \ldots \vee l_k$; we then set $q_\alpha = \overline{l_1} \wedge \ldots \wedge \overline{l_k}$, where $\overline{\neg u} = u$ and $\overline{v} = \neg v$.
- for each remaining conjunct α_i in α,
  1. if α_i is a letter v, then $I_\alpha(v) = \{o\}$
  2. if α_i is $l_1 \vee \ldots \vee l_k$ for $k \geq 2$, where at least one literal is positive, say w.l.o.g. that l_1 is the positive literal u, then the subsumption relationship $(\overline{l_2} \wedge \ldots \wedge \overline{l_k}, u)$ is in $\preceq_\alpha$.
- nothing else is in I_α, q_α or $\preceq_\alpha$.

For instance, the propositional formula

$$\begin{aligned}\alpha = {} & a2 \wedge b2 \wedge \\ & (a1 \vee \neg a2 \vee b1) \wedge (a1 \vee b1 \vee \neg b2) \wedge \\ & \neg a1 \wedge \neg b1\end{aligned}$$

is mapped into the source shown in Figure 8 and the query $a1 \vee b1$. We now show the following

Lemma $o \in ans(q_\alpha, S_\alpha)$ iff α is unsatisfiable.
In fact, we prove the equivalent form: $o \notin ans(q_\alpha, S_\alpha)$ iff α is satisfiable.
($\rightarrow$) Suppose α is satisfiable, and let f be a truth assignment over V satisfying it. Let J be the interpretation of the terminology V such that, for each term $t \in V$,

$$J(t) = \begin{cases} \{o\} \text{ if } f(t) = T \\ \emptyset \quad otherwise \end{cases}$$

We have that $I_\alpha \leq J$, since for each $t \in V$, either $I_\alpha(t)$ is empty, or $I_\alpha(t) = \{o\}$. In the former case, $I_\alpha(t) \subseteq J(t)$ for any $J(t)$. In the latter case, we have that $\alpha_j = t$ for some $1 \leq j \leq n$, which implies $f(t) = T$ (since f satisfies α) which implies $J(t) = \{o\}$ and again $I_\alpha(t) \subseteq J(t)$. Moreover, $(q, u) \in \preceq_\alpha$ implies $J(q) \subseteq J(u)$. In proof, $(q, u) \in \preceq_\alpha$ iff $\alpha_k = \neg q \vee u$ for some $1 \leq k \leq n$, which implies $f(\neg q \vee u) = T$ (since f satisfies α) and therefore: either $f(\neg q) = T$ and by construction $J(q) = \emptyset$, or $f(u) = T$ and by construction $J(u) = \{o\}$; in both cases $J(q) \subseteq J(u)$. Hence J is a model of A_α. However, $o \notin J(q_\alpha)$. In fact, by construction, for any disjunct d in q_α, there exists $\alpha_j = \neg d$ for some $1 \leq j \leq n$. Since f satisfies α, it follows that f satisfies $\neg d$ so $f(d) = F$. But then $J(d) = \emptyset$ for each conjunct d in q_α, which implies $J(q_\alpha) = \emptyset$. So, $o \notin J(q)$ for a model J of A_α, that is $o \notin ans(q_\alpha, S_\alpha)$.

($\leftarrow$) Suppose $o \notin ans(q_\alpha, S_\alpha)$, and let J be a model of A_α such that $o \notin J(q_\alpha)$. Let f be the truth assignment over V defined as follows, for each letter $t \in V$,

$$f(t) = \begin{cases} T \text{ if } o \in J(t) \\ F \textit{ otherwise} \end{cases}$$

By a similar argument to the one developed in the *if* part of the proof, it can be proved that f satisfies α, and this completes the proof of the Lemma.

From the last Lemma and the NP-completeness of SAT, the coNP-hardness of deciding query answers in neg-extended sources follows. □

6 Query to query articulations

Query to query (q2q) articulations establish subsumption relationships between queries, and are the most sophisticate representation scheme for data integration. Query answering in this context requires deciding query containment, a notoriously difficult task from the computational point of view [8].

We will address two different kinds of q2q articulations, leaving negation out of the considered languages, in light of the negative results reported in the previous Section.

A conjunctive articulation has the form $q \preceq r$ where q is a negation-free DNF query, *i.e.* an expression of the language $\mathcal{L}_T$, while r is a conjunction of terms. A conjunctive taxonomy $(T, \preceq_c)$ is just a terminology and set of conjunctive articulations. From a logical point of view, a conjunctive taxonomy is just a notational variant of an extended (*i.e.*, t2q) taxonomy. In fact, it can be shown that an interpretation of a terminology T is a model of a conjunctive taxonomy $(T, \preceq_c)$ if and only if it is a model of the taxonomy $(T, \preceq)$, where $\preceq$ is obtained from $\preceq_c$ by replacing each subsumption relationship $(q, t_1 \wedge \ldots \wedge t_m)$ in $\preceq_c$, with the m relationships $(q, t_1), \ldots, (q, t_m)$. Then all the results reported in Section 5 carry over conjunctive articulations.

A disjunctive articulation has the form $q \preceq q'$ where both q and q' are negation-free DNF queries, *i.e.* an expression of the language $\mathcal{L}_T$. Disjunction in the right-hand side of sumsumption relationships cannot be reduced, and, as expected, is expressive enough to allow the existence of sources which do not have a unique minimal model. As an example, the source $S = (A, I)$, where $A = (\{a, b, c\}, \{(a, b \vee c)\})$ and $I = \{(b, \{1\}), (c, \{2\})\}$ has two minimal models, $I_1 = I \cup \{(a, \{1\})\}$ and $I_2 = I \cup \{(a, \{2\})\}$.

Even though articulations are negation-free, loosing the uniqueness of the minimal model is enough to make query evaluation for this kind of sources computationally difficult.

Proposition 6. A *disjunctive taxonomy* is a pair $(T, \preceq_d)$ where T is a terminology and $\preceq_d \subseteq (\mathcal{L}_T \times \mathcal{L}_T)$, reflexive and transitive. A *disjunctive source* S is a pair (A, I), where A is a disjunctive taxonomy $(T, \preceq_d)$ and I is an interpretation of T. Deciding whether an object $o \in Obj$ is in the answer of an extended query q in a disjunctive source S, $o \in ans(q, S)$, is a coNP-hard problem.

The proof is similar to that of the previous Proposition. For brevity, we just show the reduction from SAT. Let α be as in the proof of Proposition 5. Let o be any object in Obj; then:

- the query q_α is given by

$$\bigvee\{v_1 \wedge \ldots \wedge v_k | \neg v_1 \vee \ldots \vee \neg v_k \text{ is a conjunct in } \alpha\} \vee \\ \bigvee\{\neg u_1 \wedge \ldots \wedge \neg u_k \mid u_1 \vee \ldots \vee u_k \text{ is a conjunct in } \alpha\}$$

  If there are no such conjuncts $\neg v_1 \vee \ldots \vee \neg v_k$ or $\neg u_1 \wedge \ldots \wedge \neg u_k$ in α, then let α_1 be $l_1 \vee \ldots \vee l_k$; we then set $q_\alpha = \overline{l_1} \wedge \ldots \wedge \overline{l_k}$, where $\overline{\neg u} = u$ and $\overline{v} = \neg v$.
- for each remaining conjunct α_i in α,
  1. if α_i is a letter v, then $I_\alpha(v) = \{o\}$
  2. if α_i is $\neg u_1 \vee \ldots \vee \neg u_j \vee v_1 \vee \ldots \vee v_m$ where $j, m \geq 1$ then the subsumption relationship $(u_1 \wedge \ldots \wedge u_j, v_1 \vee \ldots \vee v_m)$ is in $\preceq_\alpha$.
- nothing else is in I_α, q_α or $\preceq_\alpha$.

In the present case, the propositional formula

$$\alpha = a2 \wedge b2 \wedge \\ (a1 \vee \neg a2 \vee b1) \wedge (a1 \vee b1 \vee \neg b2) \wedge \\ \neg a1 \wedge \neg b1$$

is mapped into the source shown in Figure 9 and the query $a1 \vee b1$. It can be shown that:

Lemma $o \in ans(q_\alpha, S_\alpha)$ iff α is unsatisfiable. □

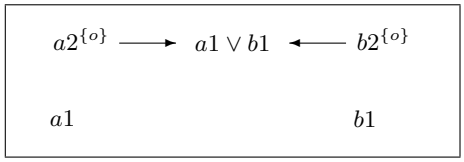

Fig. 9. A disjunctive source

7 Related Work

The approach to information retrieval on P2P networks considered in this study, starts to receive noteworthy attention by the researchers, as is believed that the database and knowledge base research has much to contribute to the P2P grand challenge through its wealth of techniques for sophisticated semantics-based data models and query processing techniques (e.g. see [9–11]). Of course, a P2P system might impose a single conceptual model on all participants to enforce uniform, global access, but this will be too restrictive. Alternatively, a limited number

of conceptual models may be allowed, so that traditional information mediation and integration techniques will likely apply (with the restriction that there is no central authority), e.g. see [12, 13]. The case of fully heterogeneous conceptual models makes uniform global access extremely challenging and this is the case that we are interested in.

From a data modeling point of view several approaches for P2P systems have been proposed recently, including relational-based approaches [10], XML-based approaches [14] and RDF-based [13]. In this paper we consider a taxonomy-based conceptual modeling approach. This approach has three main advantages (for more see [3]): (a) it is very easy to create the conceptual model of a source, (b) the integration of information from multiple sources can be done easily, and (c) automatic articulation using data-driven methods (like the one presented in [15]) are possible.

From an architectural point of view, and according to the SIL (Search Index Link) model presented in [16], our networks falls into the case of P2P systems which have only forwarding search links. Specifically, our work specializes content-based queries to taxonomy-based queries. Another distinguishing characteristic, is that in our model a peer does not just forward the received queries to its neighbors, it first translates them. Also note that the relationships stored in the articulations not only determine query translation but also query propagation. Of course, work done on P2P architectures, e.g. [17, 16], could be also exploited in our setting in order to enhance the efficiency of a taxonomy-based P2P system. Our approach has some similirities with Edutella [12, 13], an RDF-based metadata infrastructure for P2P systems. However, the mediators of Edutella distribute a query to a peer only if the query can be answered completely by the peer. In contrast, in our model the answers of queries are formed collaboratively. Moreover, in Edutella special servers are devoted for registering the schema that each peer supports. In our model we do not make any such assumption.

An approach for supporting object queries appropriate for domains where no accepted naming standards exist (and thus it generalizes the functionality provided by systems like Napster and Gnutella) is described in [11]. The mapping tables employed there can express only exact mappings, however the open/closed-world semantics that are given are quite interesting and their application to our setting is one topic of our research agenda.

8 Conclusions

We have addressed the problem of evaluating queries stated against information sources storing objects indexed according to a taxonomies. Different representation schemes and query languages have been examined, with the objective of tracing the boundaries between cases in which query evaluation is tractable from those in which it is intractable. To this end, we have focused more on the analysis of the problems from a computational point of view than on the aspects related to the peer-to-peer architecture. In spite of this, our model is clearly con-

ceived with these architectures in mind, so the results that have been derived in this paper constitute a necessary foundational step towards the development of peer-to-peer information systems based on taxonomical classification schemata.

References

1. Ling, B., Lu, Z., Ng, W.S., Ooi, B., Tan, K.L., Zhou, A.: "A Content-Based Resource Location Mechanism in PeerIS". In: Proc. of the 3rd International Conference on Web Information Systems Engineering, WISE 2002, Singapore (2002)
2. Koubarakis, M., Tryfonopoulos, C.: "Peer-to-Peer Agent Systems for Textual Information Dissemination: Algorithms and Complexity". In: Proceedings of the UK Workshop on Multiagent Systems,UKMAS'02, Liverpool, UK (2002)
3. Tzitzikas, Y., Meghini, C., Spyratos, N.: "Taxonomy-based Conceptual Modeling for Peer-to-Peer Networks". In: Proceedings of 22th Int. Conf. on Conceptual Modeling, ER'2003, Chicago, Illinois (2003)
4. Tzitzikas, Y., Spyratos, N., Constantopoulos, P.: "Mediators over Taxonomy-based Information Sources". VLDB Journal (2004) (to appear).
5. Meghini, C., Tzitzikas, Y.: Query evaluation in peer-to-peer networks of taxonomy-based sources. In: Proceedings of CooPIS-2003, the Tenth International Conference on Cooperative Information Systems. LNCS 2888, Springer Verlag (2003) 263–281
6. Ceri, S., Gottlob, G., Tanca, L.: Logic Programming and Databases. Springer Verlag (1990)
7. Ullman, J.D.: "Principles of Database and Knowledge-Base Systems, Vol. I". Computer Science Press (1988)
8. Lenzerini, M.: Data integration: A theoretical perspective. In: Proceedings of PODS 2002, the twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Madison, Winsconsin, USA (2002)
9. Gribble, S., Halevy, A., Ives, Z., Rodrig, M., Suiu, D.: "What can Databases do for Peer-to-Peer?". In: Proceedings of WebDB01, Santa Barbara, CA (2001)
10. Bernstein, P.A., Giunchiglia, F., Kementsietsidis, A., Mylopoulos, J., Serafini, L., Zaihrayeu, I.: "Data Management for Peer-to-Peer Computing: A Vision". In: Proceedings of WebDB02, Madison, Wisconsin (2002)
11. Kementsietsidis, A., Arenas, M., Miller, R.J.: "Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues". In: Int. Conf. on Management of Data, SIGMOD'2003, San Diego, California (2003)
12. Nejdl, W., Wolf, B., Staab, S., Tane, J.: "EDUTELLA: Searching and Annotating Resources within an RDF-based P2P Network". In: Semantic Web Workshop 2002, Honolulu, Havaii (2002)
13. Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmer, M., Risch, T.: "EDUTELLA: A P2P networking infrastructure based on RDF". In: WWW'2002. (2002)
14. Halevy, A., Ives, Z., Mork, P., Tatarinov, I.: "Piazza: Data Management Infrastructure for Semantic Web Applications". In: Proceedings of WWW'2003. (2003)
15. Tzitzikas, Y., Meghini, C.: "Ostensive Automatic Schema Mapping for Taxonomy-based Peer-to-Peer Systems". In: Seventh International Workshop on Cooperative Information Agents, CIA-2003, Helsinki, Finland (2003)
16. Cooper, B., Garcia-Molina, H.: "Modeling and Measuring Scalable Peer-to-peer Search Networks". Technical report, University of Stanford (2002)
17. Yang, B., Garcia-Molina, H.: "Comparing Hybrid Peer-to-Peer Systems". In: The VLDB Journal. (2001) 561–570