

# A Unifying Framework for Flexible Information Access in Taxonomy-based Sources<sup>1</sup>

Yannis Tzitzikas<sup>2</sup>, Carlo Meghini<sup>3</sup> and Nicolas Spyrtatos<sup>4</sup>

<sup>2</sup>*Information Technology, VTT Technical Research Centre of Finland*  
`ext-yannis.tzitzikas@vtt.fi`

<sup>3</sup>*Istituto di Scienza e Tecnologie dell' Informazione [ISTI], CNR, Pisa, Italy*  
`meghini@isti.cnr.it`

<sup>4</sup>*Laboratoire de Recherche en Informatique, Universite de Paris-Sud, France*  
`spyrtatos@lri.fr`

**Abstract.** A taxonomy-based source consists of a taxonomy and a database storing objects that are indexed in terms of the taxonomy. For this kind of sources, we describe a *flexible* interaction scheme that allows users to retrieve the objects of interest without having to be familiar with the terms of the taxonomy or with the supported query language. Specifically we describe an *interaction manager* whose functionality unifies several well-known interaction schemes including query by example, answer enlargement/reduction, query relaxation/restriction, index relaxation/contraction, feedback and adaptation mechanisms.

## 1 Introduction

A taxonomy-based source consists of a taxonomy and a set of objects that are indexed with respect to the taxonomy. Given a taxonomy-based source a user can find the desired objects by either browsing or querying the source. In this paper we propose an interaction scheme whose objective is to make the desired objects easy to find for the user, even if the source has a query language which is *unknown* to the user. The proposed scheme is actually a generalization of the interaction schemes that are currently used. The proposed interaction manager supports several kinds of interaction, including query by example, index relaxation/contraction, query relaxation/restriction, answer enlargement/reduction, "relevance" feedback, and adaptation facilities, in a uniform manner. This interaction manager is actually a specialization of the interaction manager of the generalized interaction scheme that was proposed in [16]. According to [16], the interaction of a user with an information source is viewed as a sequence of *transitions* between *contexts* where a context is a consistent "interaction state". The user has at his/her disposal several ways to express the desired transition. The traditional query-and-answer interaction scheme is only one kind of transition. Then, it is the interaction manager that has to find (and drive the user to) the new context. Methods allowing the user to specify a transition relatively to the

---

<sup>1</sup> Work supported by ERCIM and the DELOS NoE in Digital Libraries, Contract No G038-507618

current context are also provided. Furthermore, methods for restricting the set of transitions to those that can indeed lead to a context were described. That work [16] gave the foundations of this interaction manager from a mathematical point of view, in terms of an abstract view of an information source that generalizes information retrieval bases [1], databases and knowledge bases [17]. The current paper specializes this framework for taxonomy-based sources and shows how the corresponding computational tasks can be performed.

The paper is organized as follows. Section 2 introduces taxonomy-based sources and Section 3 recalls the more essential parts of the generalized interaction scheme proposed in [16]. Subsequently, Section 4 describes how this scheme can be applied on taxonomy-based sources. Finally, Section 5 concludes the paper and identifies issues for further research.

## 2 Taxonomy-based Sources

Taxonomies is probably the oldest and most widely used conceptual modeling tool still used in Web directories (e.g. in Google and Yahoo!), Content Management (hierarchical structures are used to classify documents), Web Publishing (many authoring tools require to organize the contents of portals according to some hierarchical structure), Web Services (services are typically classified in a hierarchical form), Marketplaces (goods are classified in hierarchical catalogs), Personal File Systems, Personal Bookmarks for the Web, Libraries (e.g. Thesauri [9]) and in very large collections of objects (e.g. see [13]). Although more sophisticated conceptual models (including concepts, attributes, relations and axioms) have emerged and are recently employed even for meta-tagging in the Web, almost all of them have a backbone consisting of a subsumption hierarchy, i.e. a taxonomy.

We view a taxonomy-based source  $S$  as a quadruple  $S = \langle T, \preceq, I, Q \rangle$  where:

- $T$  is terminology, i.e. a finite set of names called *terms*, e.g. **Canaries**, **Birds**.
- $\preceq$  is a reflexive and transitive binary relation over  $T$  (i.e. a pre-order) called *subsumption*, e.g. **Canaries**  $\preceq$  **Birds**,
- $I$  is a function  $I : T \rightarrow 2^{Obj}$  called *interpretation* where  $Obj$  is a finite set of objects called *domain*. For example  $Obj = \{1, \dots, 100\}$  and  $I(\text{Canaries}) = \{1, 3, 4\}$ , and
- $Q$  is the set of all queries defined by the grammar  $q ::= t \mid q \wedge q' \mid q \vee q' \mid \neg q \mid (q)$  where  $t$  is a term in  $T$ .

The pair  $(T, \preceq)$  is the taxonomy of  $S$ .

We assume that every terminology  $T$  also contains two special terms, the *top term*, denoted by  $\top$ , and the *bottom term*, denoted by  $\perp$ . The top term subsumes every other term  $t$ , i.e.  $t \preceq \top$ . The bottom term is strictly subsumed by every other term  $t$  different than top and bottom, i.e.  $\perp \preceq \perp$ ,  $\perp \preceq \top$ , and  $\perp \prec t$ , for every  $t$  such that  $t \neq \top$  and  $t \neq \perp$ . We also assume that  $I(\perp) = \emptyset$  in every interpretation  $I$ .

Query answering in a source  $S$  is based on the notion of model. An interpretation  $I$  is a *model* of a taxonomy  $(T, \preceq)$  if for all  $t, t'$  in  $T$ , if  $t \preceq t'$  then  $I(t) \subseteq I(t')$ . Given an interpretation  $I$  of  $T$ , the model of  $(T, \preceq)$  *generated* by  $I$ , denoted  $\bar{I}$ , is given by:  $\bar{I}(t) = \bigcup\{I(s) \mid s \preceq t\}$ .

An interpretation  $I$  can be extended to an interpretation of queries as follows:  $I(q \wedge q') = I(q) \cap I(q')$ ,  $I(q \vee q') = I(q) \cup I(q')$ , and  $I(\neg q) = Obj \setminus I(q)$ .

Given a source  $S = \langle T, \preceq, I, Q \rangle$  and a query  $q \in Q$ , the answer of  $q$  is the set  $\bar{I}(q)$ , also denoted by  $S(q)$ .

Figure 1 (below at Section 4) illustrates two taxonomy-based sources. Objects are represented by natural numbers, the membership of an object to the interpretation of a term is indicated by a dotted arrow from the object to that term, subsumption of terms is indicated by a continuous-line arrow from the subsumed term to the subsuming term. We do not illustrate the top and bottom terms and we illustrate only the transitive reduction of the subsumption relation.

### 3 The Generalized Interaction Scheme for Information Access

This scheme is described in terms of an abstract view of a source. Specifically, a source  $S$  is viewed as a function  $S : Q \rightarrow \mathcal{A}$  where  $Q$  is the set of all queries that  $S$  can answer, and  $\mathcal{A}$  is the set of all answers to those queries, i.e.  $\mathcal{A} = \{S(q) \mid q \in Q\}$ . As we focus on retrieval queries, we assume that  $\mathcal{A}$  is a subset of  $\mathcal{P}(Obj)$ , the powerset of  $Obj$ , where  $Obj$  is the set of all objects stored at the source.

Let  $\mathcal{S}$  be the set of all sources that can be derived by "updating" the source  $S$  (e.g. for adapting it), but for the moment let us suppose that  $\mathcal{S}$  is the set of all functions from  $Q$  to  $\mathcal{P}(Obj)$ .

Let  $\mathcal{U}$  denote the set of all triples in  $\mathcal{S} \times Q \times \mathcal{A}$ , i.e.  $\mathcal{U} = \mathcal{S} \times Q \times \mathcal{A}$ . A triple  $c = (S, q, A) \in \mathcal{U}$  is called an interaction context, or *context* for short, if  $S(q) = A$ . Let  $\mathcal{C}$  denote the set of all contexts, i.e.  $\mathcal{C} = \{(S, q, A) \in \mathcal{U} \mid S(q) = A\}$ . Given a context  $c = (S, q, A)$ ,  $S$  is called the *source view* of  $c$ ,  $q$  is called the *query* of  $c$  and  $A$  is called the *answer* of  $c$ . The interaction between the user and the source is carried out by a software module called *Interaction Manager* (IM). The interaction of a user with the source is viewed as a sequence of *transitions* between contexts. At any given time, the user is in one context, the *focal context*. At the beginning of the interaction with the system the user starts from the initial context  $(S, \epsilon, \emptyset)$  where  $S$  is the stored information source,  $\epsilon$  is the empty query and  $\emptyset$  is the empty answer<sup>2</sup>. However, any context of  $\mathcal{U}$  could serve as the initial context, e.g. the context  $(S, \top, Obj)$ . There are several methods the user can use for moving from one context to another, i.e. for changing the focal context. For example, in the traditional query-and-answer interaction scheme, the user actually "replaces" the current query  $q$  by formulating a new query  $q'$  and the "interaction manager" drives him to the new context  $(S, q', A')$  where

<sup>2</sup> We assume that  $S(\epsilon) = \emptyset$  therefore  $(S, \epsilon, \emptyset)$  is a context.

$A' = S(q')$ . However this is only one way of changing the focal context. Several other ways will be presented below.

### 3.1 Context Transition Specifications (CTSs) through Replacements

Suppose that the user can *replace* one component of the focal context (i.e. either  $S$ ,  $q$  or  $A$ ) by explicitly providing another component (resp.  $S'$ ,  $q'$  or  $A'$ ). So we can have three kinds of replacements: (a) query replacements, denoted by  $[q \rightarrow q']$ , (b) answer replacements, denoted by  $[A \rightarrow A']$ , and (c) source view replacements, denoted by  $[S \rightarrow S']$ .

As the user should always be in a context i.e. in a triple  $(S, q, A)$  where  $S(q) = A$ , after any of the above kinds of replacement, the IM should try to reach a context  $c'$  by changing one (or both) of the remaining two components of the focal context. Instead of leaving the IM to decide, it is the user that indicates to the IM the component(s) to be changed after a replacement. A replacement plus an indication of the above kind, is a *Context Transition Specification (CTS)*. Below we list the CTSs that we consider and discuss in brief the motivation beneath.

- $[q \rightarrow q'/A]$ . Here the answer  $A$  must be changed. This is the classical query-and-answer interaction scheme: when the user replaces the current query  $q$  with a new query  $q'$ , the user is given a new answer (i.e. an  $A'$  such that  $A' = S(q')$ ). Thus we can write:  $[q \rightarrow q'/A](S, q, A) = (S, q', S(q'))$ .
- $[S \rightarrow S'/A]$ . This is again a classical interaction scheme: whenever the source changes (e.g. after an update) the answers of queries change as well, i.e. here we have  $A' = S'(q)$ .
- $[A \rightarrow A'/q]$ . Here the query must be changed. This interaction scheme may help the user to get acquainted with the query language of the source. It can be construed as an alternative query formulation process. The user selects a number of objects (i.e. the set  $A'$ ) and asks from the IM to formulate the query that "describes" these objects. Subsequently the user can change the query  $q'$  in a way that reflects his/her information need. Roughly this resembles the Query By Example (QBE) process in relational databases. It also resembles the relevance feedback mechanisms in Information Retrieval systems. For example, the user selects a subset  $A'$  of the current answer  $A$  consisting of those elements of  $A$  which the user finds relevant to his/her information need. Subsequently, the IM has to change appropriately the query.
- $[S \rightarrow S'/q]$ . Here the query  $q$  must be changed. This resembles the way that a relational database management system changes the query  $q$  that defines a relational view, after an update of the database, in order to preserve the contents (tuples) of the view.
- $[A \rightarrow A'/S]$ . Here the source view  $S$  has to be changed. Here the user wants  $A'$  (instead of  $A$ ) to be the answer of  $q$ . The IM should try to *adapt* to the desire of the user by changing the source view from  $S$  to an  $S'$  such that  $S'(q) = A'$ . So this is a flexible and easy-to-use method that allows users to express their demand for source adaptation.

- $[q \rightarrow q'/S]$  Here again the source view  $S$  must be changed. This means that the user replaces the current query  $q$  by a query  $q'$ , because the user wants the current answer  $A$  to be the answer of  $q'$ , not of  $q$ . The IM should try to *adapt* to the desire of the user by changing the source view from  $S$  to an  $S'$  such that  $S'(q') = A$ .

The second column of Table 1 lists each kind of CTS that can be applied on a focal context  $c = (S, q, A)$ . Given a context  $c$  and a context transition specification  $R$ , the role of the IM is to find the desired target context  $R(c)$ , if one exists. The third column shows the target context after each kind of CTS where boldface is used to indicate the component that the IM has to compute in order to reach that context, assuming that only one of the remaining two components of the focal context can be changed. The notations  $n_S(A')$  and  $n_{S'}(A)$  denote queries that will be explained in detail in the next section.

**Table 1.** Context Transitions Specifications

CTS num	CTS notation	$R(c)$	evaluation of $R(c)$
(1)	$[q \rightarrow q'/A]$	$(S, q', \mathbf{S}(q'))$	relies on query evaluation
(2)	$[S \rightarrow S'/A]$	$(S', q, \mathbf{S}(q))$	relies on query evaluation
(3)	$[A \rightarrow A'/q]$	$(S, \mathbf{n_S}(A'), A')$	relies on naming functions
(4)	$[S \rightarrow S'/q]$	$(S', \mathbf{n_{S'}}(A), A)$	relies on naming functions
(5)	$[A \rightarrow A'/S]$	$(\mathbf{S}', q, A')$	relies on source adaptation
(6)	$[q \rightarrow q'/S]$	$(\mathbf{S}', q', A)$	relies on source adaptation

### 3.2 Finding the Target Context

In CTSs (1) and (2) of Table 1, the IM has to change the answer in order to reach to a context. These cases are relatively simple as the target context always exists and the desired answer  $A'$  can be derived using the query evaluation mechanism of the source. However, the cases in which the IM has to find a new query (i.e. in CTSs (3) and (4)) or a new source view (i.e. in CTSs (5) and (6)) are less straightforward as the target context does not always exist.

In CTS (3) and (4) the IM has to find a  $q \in Q$  such that  $S(q) = A$  for given  $S$  and  $A$ . Supporting these cases requires having a "naming service", i.e. a method for computing one or more queries that describe (name) a set of objects  $A \subseteq Obj$ . Ideally we would like a function  $n : \mathcal{P}(Obj) \rightarrow Q$  such that for each  $A \subseteq Obj$ ,  $S(n(A)) = A$ . Having such a function, we would say that the query  $n(A)$  is an exact name for the object set  $A$ . Note that if  $S$  is an *onto* function then the naming function  $n$  coincides with the inverse relation of  $S$ , i.e. with the relation  $S^{-1} : \mathcal{P}(Obj) \rightarrow Q$ . However, this is not always the case, as more often than not,  $S$  is not an onto function, i.e.  $\mathcal{A} \subset \mathcal{P}(Obj)$ . Furthermore, if  $S$  is onto and one-to-one, then  $S^{-1}$  is indeed a function, thus there is always a

unique  $q$  such that  $S(q) = A$  for each  $A \subseteq Obj$  <sup>3</sup>. As  $S$  is not always an onto function, "approximate" naming functions are introduced, specifically a *lower* naming function  $n^-$  and an *upper* naming function  $n^+$ , defined as follows:

$$\begin{aligned} n^-(A) &= lub\{q \in Q \mid S(q) \subseteq A\} \\ n^+(A) &= glb\{q \in Q \mid S(q) \supseteq A\} \end{aligned}$$

where *lub* stands for least upper bound and *glb* for greatest lower bound. If  $A$  is a subset of  $Obj$  for which both  $n^-(A)$  and  $n^+(A)$  are defined (i.e. the above *lub* and *glb* exist), then  $S(n^-(A)) \subseteq A \subseteq S(n^+(A))$  and  $n^-(A)$  and  $n^+(A)$  are the best "approximations" of the exact name of  $A$ . Note that if  $S(n^-(A)) = S(n^+(A))$  then both  $n^-(A)$  and  $n^+(A)$  are exact names of  $A$ .

Let us now return to CTS (3) and (4). If a naming function  $n_S$  is available for source  $S$ , then the target contexts exist and they are the ones shown in the third column of Table 1. If only approximate naming functions are available, then two "approximate" next contexts exist:

$$\begin{aligned} - [A \rightarrow A'/q](S, q, A) &= \begin{cases} (S, n_S^-(A'), S(n_S^-(A'))) \\ (S, n_S^+(A'), S(n_S^+(A'))) \end{cases} \\ - [S \rightarrow S'/q](S, q, A) &= \begin{cases} (S', n_{S'}^-(A), S'(n_{S'}^-(A))) \\ (S', n_{S'}^+(A), S'(n_{S'}^+(A))) \end{cases} \end{aligned}$$

Notice that in the above cases, IM does not change only  $q$ , but also the answer. In the case  $[A \rightarrow A'/q]$ , the new context has an answer, say  $A''$ , which is the closest possible to the requested (by the user) answer  $A'$ . In the case  $[S \rightarrow S'/q]$ , the new context has an answer  $A'$  which is the closest possible to the current  $A$ .

In CTS (5) and (6) the IM is looking for a  $S \in \mathcal{S}$  such that  $S(q) = A$  for given  $q$  and  $A$ . Clearly, the sought source  $S$  always exists if and only if  $\mathcal{S}$  is the set of all functions from  $Q$  to  $\mathcal{A}$ . The process of finding the desired  $S'$ , which we shall hereafter call *source adaptation*, cannot be described within this abstract framework. However, the restricted relative replacements and the source relaxation/contraction mechanisms that are introduced in the sequel, can be exploited for finding the target context even if source adaptation is not supported.

### 3.3 Relative Replacements and Relative CTSs

Relative replacements allow the user to specify the desired replacement without having to provide explicitly the new component but by defining it relatively to the current. These methods can be very helpful for the user during the interaction with the system. Relative replacements are based on the three partially ordered sets (posets) that are associated with each source, namely:

- The poset of answers  $(\mathcal{P}(Obj), \subseteq)$

<sup>3</sup> Usually, sources are not one-to-one functions. For instance, the supported query language may allow formulating an infinite number of different queries, while the set of all different answers  $\mathcal{A}$  is usually finite (e.g.  $\mathcal{P}(Obj)$ ).

- The poset of queries  $(Q, \leq)$ . Given two queries  $q$  and  $q'$  of  $Q$ , we write  $q \leq q'$  iff  $S(q) \subseteq S(q')$  in every possible source  $S$  in  $\mathcal{S}$ . We write  $q \sim q'$  if both  $q \leq q'$  and  $q' \leq q$  hold. Let  $Q_{\sim}$  denote the set of equivalence classes induced by  $\sim$  over  $Q$ .
- The poset of sources  $(\mathcal{S}, \sqsubseteq)$ . Given two sources  $S$  and  $S'$  in  $\mathcal{S}$ ,  $S \sqsubseteq S'$  iff  $S(q) \subseteq S'(q)$  in every query  $q \in Q$ .

For every element  $x$  of the above lattices, let

$Br(x)$  denote the elements that are greater than or equal to  $x$ ,

$Nr(x)$  the elements that are less than or equal to  $x$ ,

$x^+$  a component that covers  $x$ , and  $x^-$  a component that is covered by  $x$ <sup>4</sup> in the corresponding poset. Note that there may not always exist a unique  $x^+$  or a unique  $x^-$ . Specifically, we may have zero, one, or more  $x^+$ 's and  $x^-$ 's for a given  $x$ .

These partial orders can be exploited by the IM for moving to a component (answer, query, or source) that covers, or is covered by, the current component. Let  $Up(x)$  denote a component among those greater than  $x$  and that the IM can compute (ideally  $Up(x) = x^+$ ), and let  $Down(x)$  denote a component among those less than  $x$  and that the IM can compute (ideally  $Down(x) = x^-$ ). These relative replacements can be used within context transition specifications. A CTS defined by a relative replacement will be called *relative CTS*. Table 2 lists all possible relative CTSs.

**Table 2.** Relative Context Transitions Specifications

CTS num	CTS notation	description
(1r)	$[q \rightarrow Up(q)/A]$	query relaxation resulting in answer enlargement
(1r)	$[q \rightarrow Down(q)/A]$	query contraction resulting in answer reduction
(2r)	$[S \rightarrow Up(S)/A]$	source relaxation resulting in answer enlargement
(2r)	$[S \rightarrow Down(S)/A]$	source contraction resulting in answer reduction
(3r)	$[A \rightarrow Up(A)/q]$	answer enlargement resulting in query relaxation
(3r)	$[A \rightarrow Down(A)/q]$	answer reduction resulting in query contraction
(4r)	$[S \rightarrow Up(S)/q]$	source relaxation resulting in query contraction
(4r)	$[S \rightarrow Down(S)/q]$	source contraction resulting in query relaxation
(5r)	$[A \rightarrow Up(A)/S]$	answer enlargement resulting in source relaxation
(5r)	$[A \rightarrow Down(A)/S]$	answer reduction resulting in source contraction
(6r)	$[q \rightarrow Up(q)/S]$	query relaxation resulting in source contraction
(6r)	$[q \rightarrow Down(q)/S]$	query contraction resulting in source relaxation

Relative CTSs can enhance flexibility during information access by the user. Also note that they can very easily be reflected at the user interface layer of the system. Indeed, the user interface can have two buttons, "Up" and "Down", for every component of the focal context. By pressing a button, the corresponding

<sup>4</sup> An element  $x$  is covered by  $y$  (or  $y$  covers  $x$ ) if  $x < y$  and there is no  $z$  such that  $x < z < y$ .

relative replacement is specified. An additional option control can allow the user to indicate to the IM the component ( $S$ ,  $q$ , or  $A$ ) that should be changed in order to reach the new context.

### 3.4 Restricting the Relative CTSs

The three partial orders mentioned earlier (and their interrelationships) can be exploited in order to restrict the set of relative CTSs to those for which the IM can indeed compute the target context.

In particular, we can restrict  $Up/Down(A)$  so that to support CTS (3) even if naming functions are not available (or if they are available, but there is no exact name for  $A'$ ). This can be achieved by defining:

$$Up(A) = S(Up(q)) \quad \text{and} \quad Down(A) = S(Down(q))$$

We can restrict  $Up/Down(A)$  so that to support CTS (5) even if source adaptation is not available. This can be achieved by defining  $Up(A)$  and  $Down(A)$  as follows:

$$Up(A) = Up(S)(q) \quad \text{and} \quad Down(A) = Down(S)(q)$$

We can restrict  $Up/Down(q)$  so that to support CTS (6) even if source adaptation is not available, but naming functions are available. This can be achieved by defining  $Up(q)$  and  $Down(q)$  as follows:

$$Up(q) = n_{Down(S)}(A) \quad \text{and} \quad Down(q) = n_{Up(S)}(A)$$

## 4 Application on Taxonomy-based Sources

For applying on taxonomy-based sources the interaction scheme just described, we have to find a method for computing the target context after every kind of CTS. The CTSs that require computing a new answer, can be supported using the query evaluation method already described in Section 2. So we only have to focus on the cases where the IM after a CTS has to find a new query or a new source view. The first case relies on naming functions and the second on source adaptation. Below we show that naming functions and source adaptation are possible for taxonomy-based sources.

### Naming functions

Given a source  $S$  and an answer  $A$ , we seek for a query  $q'$  such that  $S(q') = A$ . Finding  $q'$  requires defining the naming functions  $n^-$  and  $n^+$  for taxonomy-based sources. Naming functions for taxonomy-based sources were first described in [15] where they were exploited for creating automatically inter-taxonomy mappings. Before describing them, let us first introduce an auxiliary definition. Given an object  $o \in Obj$ , we shall use  $D_I(o)$  to denote the query obtained by taking the conjunction of all terms  $t$  such that  $o \in I(t)$ , i.e.  $D_I(o) = \bigwedge \{t \in T \mid o \in I(t)\}$ .

For keeping our notations simple, we shall also sometimes use  $D_I(o)$  to denote the set  $\{t \in T \mid o \in I(t)\}$ . It can be easily proved that if we exclude from our consideration the queries that contain negation, then the naming functions are defined as follows:

$$n^-(A) \sim \bigvee \{ D_I(o) \mid o \in A, S(D_I(o)) \subseteq A \}$$

$$n^+(A) \sim \bigvee \{ D_I(o) \mid o \in A \}$$

The proofs of the above two propositions can be found in Appendix A. It is important to note that the evaluation of the above formulas is a computationally tractable task. Indeed, the time complexity for computing  $n^-(A)$  is  $O(|Obj|^2 * |T|)$ , while the time complexity for computing  $n^+(A)$  is  $O(|Obj| * |T|)$ . As an example, consider the source shown in Figure 1.(a). Here we have:  $n^-(\{1, 2, 3\}) = a$  and  $n^+(\{1, 2, 3\}) = a \vee b$ . Now the set  $\{1, 2, 4\}$  has an exact name, i.e.  $n^-(\{1, 2, 4\}) = n^+(\{1, 2, 4\}) = a \vee (b \wedge c)$ .

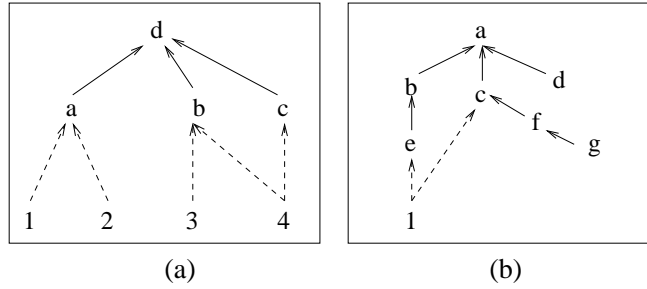


Fig. 1. Two taxonomy-based sources

### Source adaptation

Given a query  $q$  and an answer  $A$ , here we seek for a source  $S$  such that  $S(q) = A$ . We first need to define the set of sources  $\mathcal{S}$  that we assume and their ordering. Let  $S = \langle T, \preceq, I, Q \rangle$  be the original stored source. We will consider that all sources in  $\mathcal{S}$  have the same taxonomy, i.e. the taxonomy  $(T, \preceq)$ , and the same query language  $Q$ . So two sources  $S$  and  $S'$  of  $\mathcal{S}$  may differ only in their interpretation, so we assume that  $S = \langle T, \preceq, I, Q \rangle$  and  $S' = \langle T, \preceq, I', Q \rangle$ . If  $\mathcal{I}$  denotes the set of all interpretations of  $T$  over  $Obj$ , then we can define the set of all sources  $\mathcal{S}$  as follows:  $\mathcal{S} = \{ \langle T, \preceq, I, Q \rangle \mid I \in \mathcal{I} \}$ . For defining the ordering over  $\mathcal{S}$ , let us first introduce an auxiliary definition. Given two interpretations  $I$  and  $I'$  of  $T$  we write  $I \sqsubseteq I'$  if and only if  $I(t) \subseteq I'(t)$  for every  $t \in T$ . One can easily see, that if  $S$  and  $S'$  are sources of  $\mathcal{S}$ , then  $S \sqsubseteq S'$  iff  $I \sqsubseteq I'$ .

Let us now return to the problem at hand. Specifically, let us consider CTS (5), i.e.  $[A \rightarrow A'/S]$ . Here we seek for a  $S'$  such that  $S'(q) = A'$ . Let  $A^+ = A' \setminus A$  and  $A^- = A \setminus A'$ , so  $A^+$  is the set of objects that have to added to the current answer, and  $A^-$  is the set of objects that have to deleted from the current answer in order to reach  $A'$ , i.e. we can write  $A' = (A \cup A^+) \setminus A^-$ . Below we explain

how source adaptation can be achieved by considering each form that the query  $q$  may have.

At first, suppose that  $q = t$  where  $t \in T$ . For reaching the desired  $S'$  we have to update the interpretation  $I$  of  $S$ . Note that  $\bar{I}(t)$  must no longer contain the set  $A^-$ , but it has to contain the set  $A^+$ . Recall that an object  $o$  may belong to  $\bar{I}(t)$  either because  $o \in I(t)$  or because  $o \in I(t')$  where  $t' \preceq t$ . So in order to exclude from  $\bar{I}(t)$  an object  $o$  of  $A^-$  we have to delete it from the interpretations of all terms that are narrower than  $t$ , specifically from the terms  $\{t' \in D_I(o) \mid t' \preceq t\}$ . So for each  $t' \in D_I(o)$  such that  $t' \preceq t$  we must set  $I'(t') := I(t') \setminus \{o\}$ . Now in order to add to  $\bar{I}(t)$  an object  $o$  of  $A^+$  we can just add it to the interpretation of  $t$ . So we must set  $I''(t) := I'(t) \cup A^+$ .

Now suppose that  $q$  is a conjunction  $q = t_1 \wedge \dots \wedge t_k$ . One can easily see that the set  $A^+$  must be somehow added to each  $\bar{I}(t_i)$  for  $i = 1, \dots, k$ . However we don't necessarily have to delete the set  $A^-$  from each  $\bar{I}(t_i)$ , as it suffices to delete each object of  $A^-$  from only one  $\bar{I}(t_i)$ . In this case, we need a criterion for selecting the term whose interpretation we are going to reduce. Among the several possible criteria, we select to delete an object  $o$  of  $A^-$  from the interpretation of the term to which this deletion causes the less perturbation to the interpretation  $I$ . Let us now define formally the minimal perturbation criterion. Let  $Nr(t) = \{t' \mid t' \preceq t\}$  and  $Br(t) = \{t' \mid t \preceq t'\}$ . We can now define the perturbation of deleting  $o$  from the interpretation of a term  $t$  as follows:

$$pertDel(o, t) = |Nr(t) \cap D_{\bar{I}}(o)|$$

For example, consider the source shown in Figure 1.(b). In this source we have  $D_I(1) = \{e, c\}$  and  $D_{\bar{I}}(1) = \{a, b, c, e\}$ . Some examples follow:

$$\begin{aligned} pertDel(1, c) &= |Nr(c) \cap D_{\bar{I}}(1)| = |\{c, f, g\} \cap \{a, b, c, e\}| = |\{c\}| = 1 \\ pertDel(1, b) &= |Nr(b) \cap D_{\bar{I}}(1)| = |\{b, e\} \cap \{a, b, c, e\}| = |\{b, e\}| = 2 \\ pertDel(1, a) &= |Nr(a) \cap D_{\bar{I}}(1)| = |\{a, b, c, d, e, f, g\} \cap \{a, b, c, e\}| = |\{a, b, c, e\}| = 4 \end{aligned}$$

Also note that if  $o \notin \bar{I}(t)$  then  $pertDel(o, t) = 0$ .

Now suppose that  $q$  is a disjunction of terms, i.e.  $q = t_1 \vee \dots \vee t_k$ . It is evident that here the set  $A^-$  must be excluded from each  $\bar{I}(t_i)$ ,  $i = 1, \dots, k$ . On the other hand, we don't have to add the objects of  $A^+$  to all  $\bar{I}(t_i)$ , as it suffices to add each object of  $A^+$  to only one term of  $t$ . We can again adopt a minimal perturbation criterion for selecting the appropriate term of  $q$ . Specifically, we can define the perturbation of adding an object  $o$  to the interpretation of a term  $t$  as follows:

$$pertAdd(o, t) = |Br(t) \setminus D_{\bar{I}}(o)|$$

Some examples over the source of Figure 1.(b) follow:

$$\begin{aligned} pertAdd(1, d) &= |Br(d) \setminus D_{\bar{I}}(1)| = |\{a, d\} \setminus \{a, b, c, e\}| = |\{d\}| = 1 \\ pertAdd(1, f) &= |Br(f) \setminus D_{\bar{I}}(1)| = |\{a, c, f\} \setminus \{a, b, c, e\}| = |\{f\}| = 1 \\ pertAdd(1, g) &= |Br(g) \setminus D_{\bar{I}}(1)| = |\{a, c, f, g\} \setminus \{a, b, c, e\}| = |\{f, g\}| = 2 \end{aligned}$$

Also note that if  $o \in \bar{I}(t)$  then  $pertAdd(o, t) = 0$ .

Now the queries that contain all boolean connectives can be handled analogously. Specifically, Table 3 sketches the needed algorithms for each kind of

queries. The table includes the case of queries in CNF (conjunctive normal form). Note that any query with logical connectives can be converted to CNF by using one of the existing algorithms (e.g. see [8]).

**Table 3.** Source Adaptation

query form	algorithm
$q = t$	(a) for each $o \in A^-$ do <b>Delete</b> ( $o, t$ ) (b) $I'(t) := I(t) \cup A^+$
$q = \neg t$	(a) for each $o \in A^+$ do <b>Delete</b> ( $o, t$ ) (b) $I'(t) := I(t) \cup A^-$
$q = t_1 \vee \dots \vee t_k$	(a) for each $o \in A^-$ do for each $i = 1, \dots, k$ do <b>Delete</b> ( $o, t_i$ ) (b) for each $o \in A^+$ do $m = \arg_i \min \{pertAdd(o, t_i) \mid i = 1, \dots, k\}$ ; $I'(t_m) := I(t_m) \cup \{o\}$ end for
$q = t_1 \wedge \dots \wedge t_k$	(a) for each $i = 1, \dots, k$ do $I'(t_i) := I(t_i) \cup A^+$ (b) for each $o \in A^-$ do $m = \arg_i \min \{pertDel(o, t_i) \mid i = 1, \dots, k\}$ ; <b>Delete</b> ( $o, t_m$ ) end for
$q = d_1 \wedge \dots \wedge d_m$ where $d_j = t_{j1} \vee \dots \vee t_{jn_j}$	(a) add each object of $A^+$ to each disjunction of $q$ (b) delete each object of $A^-$ from the disjunction that causes the less perturbation
	<i>algorithm</i> <b>Delete</b> ( $o, t$ ) for each $t' \in D_I(o)$ such that $t' \preceq t$ do $I'(t') := I(t') \setminus \{o\}$ end algorithm

In order to define relative CTCs for taxonomy-based sources we have to show how source relaxation/contraction and query relaxation/contraction can be achieved. For relaxing and contracting a source  $S$  we shall use two operations: one for relaxing and one for contracting the interpretation of  $S$ . Hereafter with  $I$ , we shall denote a model of the taxonomy. For relaxing (resp. contracting) a model  $I$ , we will use an operation  $\cdot^+$  (resp.  $\cdot^-$ ) defined as follows:

$$I^+(t) = \bigcap \{I(t') \mid t \prec t'\}$$

$$I^-(t) = \bigcup \{I(t') \mid t' \prec t\}$$

The  $\cdot^+$  operation was first introduced in [11] and it is founded on *abduction* [6]. So, if  $S = \langle T, \preceq, I, Q \rangle \in \mathcal{S}$ , then  $Up(S) = \langle (T, \preceq), I^+, Q \rangle$  and  $Down(S) = \langle (T, \preceq), I^-, Q \rangle$ .

Another remark, is that the above operators can give us an alternative solution to the source adaptation problem in the special case where  $A'$  is a subset or a superset of the current answer  $A$ . Specifically, if  $A \subset A'$  then our search space is  $Br(S)$ . We can apply iteratively the operator  $\cdot^+$  on  $S$  until reaching to a source  $S'$  such that  $S'(q) \supseteq A'$  or until reaching the fixed point of  $\cdot^+$ . If  $S'(q) = A'$  then we have found the solution, while if  $S'(q) \supset A'$  then there is no exact solution. In the last case we can define only "approximate" solutions. Analogously we can treat the case where  $A \supset A'$ . In particular, if  $A \supset A'$  then the search space is  $Nr(S)$  and we can search it by applying iteratively the operator  $\cdot^-$  on  $S$  until reaching to a source  $S'$  such that  $S'(q) \subseteq A'$  or until reaching the fixed point of  $\cdot^-$ .

Concerning relative query replacements, given a query  $q \in Q$  we need to define  $Up(q)$  and  $Down(q)$ . Note that  $Up(q)$  and  $Down(q)$  do not necessarily have to correspond to  $q^+$  and  $q^-$  respectively. Due to reasons of space we do not describe these functions in detail. Besides, mechanisms for query relaxation have already been proposed for several kinds of sources, including relational [7], semi-structured [4], Description-Logics-based [12, 3] and Web sources [10]. We only note that the answer of the query  $Up(q)$  (resp.  $Down(q)$ ) should be bigger (resp. smaller) than the current, and that intentional query containment does not always implies extensional subsumption, e.g. if  $I(a) = \{1\}$  and  $I(b) = \{1\}$  then although  $a$  is intentionally contained by  $a \vee b$ , here we have  $I(a) = I(a \vee b)$ .

In conclusion, we have just showed that the generalized interaction scheme is feasible for taxonomy-based sources. Although we have not reported refined complexity results, it is evident that all associated computational tasks have polynomial complexity.

## 5 Concluding Remarks

In this paper we specialized the unified generalized framework for information access that was proposed in [16] for the case of taxonomy-based sources. The resulting model captures several kinds of interaction that are more complex than those that are currently supported. We described the algorithms and showed that these tasks are computationally tractable.

Further research includes specializing the generalized interaction scheme for sources with conceptual models that allow representing the relationships that may hold between the individual objects of the domain. Specifically we plan to elaborate the case where sources employ Description Logics (DL) [5] knowledge bases, as DL is the knowledge representation language of the Semantic Web [2]. We strongly suspect that this specialization is again feasible because we can view a DL source as a taxonomy-based source. Indeed, there are several approaches for constructing in polynomial time the taxonomy of concepts of a DL knowledge base (e.g. see [14]).

## References

1. Ricardo Baeza-Yates and Berthier Ribeiro-Neto. “*Modern Information Retrieval*”. ACM Press, Addison-Wesley, 1999.
2. Tim Berners-Lee, James Hendler, and Ora Lassila. “The Semantic Web”. *Scientific American*, May 2001.
3. Alain Bidault, Christine Froidevaux, and Brigitte Safar. “Repairing Queries in a Mediator Approach”. In *Proceedings of the ECAI’00*, pages 406–410, Berlin, Germany, August 20–25 2000.
4. Lee Dongwon. “*Query Relaxation for XML Model*”. PhD thesis, University of California, 2002.
5. F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. “Reasoning in Description Logics”. In Gerhard Brewka, editor, *Principles of Knowledge Representation*, chapter 1, pages 191–236. CSLI Publications, 1996.
6. T. Eiter and G. Gottlob. The complexity of logic-based abduction. *Journal of the ACM*, 42(1):3–42, January 1995.
7. Terry Gaasterland. “Cooperative Answering through Controlled Query Relaxation”. *IEEE Expert: Intelligent Systems and Their Applications*, 12(5), 1997.
8. Antony Galton. “*Logic for Information Technology*”. John Wiley & Sons, 1990.
9. International Organization For Standardization. “Documentation - Guidelines for the establishment and development of monolingual thesauri”, 1986. Ref. No ISO 2788-1986.
10. Wen-Syan Li, K. Seluk Candan, Quoc Vu, and Divyakant Agrawal. “Query Relaxation by Structure and Semantics for Retrieval of Logical Web Documents”. *IEEE Transactions on Knowledge and Data Engineering*, 14(4), 2002.
11. Carlo Meghini and Yannis Tzitzikas. “An Abduction-based Method for Index Relaxation in Taxonomy-based Sources”. In *Proceedings of MFCS 2003, 28th International Symposium on Mathematical Foundations of Computer Science*, pages 592–601, Bratislava, Slovak Republic, August 2003. Springer Verlag.
12. E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. “Estimating Information Loss for Multi-ontology Based Query Processing”. In *Proceedings of Second International and Interdisciplinary Workshop on Intelligent Information Integration*, Brighton Centre, Brighton, UK, August 1998.
13. Giovanni M. Sacco. “Dynamic Taxonomies: A Model for Large Information Bases”. *IEEE Transactions on Knowledge and Data Engineering*, 12(3), May 2000.
14. S. Sanner. “Towards practical taxonomic classification for description logics on the Semantic Web”. Technical Report KSL-03-06, Stanford University, Knowledge Systems Lab, 2003.
15. Yannis Tzitzikas and Carlo Meghini. “Ostensive Automatic Schema Mapping for Taxonomy-based Peer-to-Peer Systems”. In *Seventh International Workshop on Cooperative Information Agents, CIA-2003*, pages 78–92, Helsinki, Finland, August 2003. (Best Paper Award).
16. Yannis Tzitzikas, Carlo Meghini, and Nicolas Spyrtos. “Towards a Generalized Interaction Scheme for Information Access”. In *Procs of the 3rd Intern. Symposium on Foundations of Information and Knowledge Systems, FoIKS’2004*, Vienna Austria, February 2004.
17. Jeffrey D. Ullman. “*Principles of Database and Knowledge-Base Systems, Vol. I*”. Computer Science Press, 1988.

## A Proofs

**Prop.**  $\bigvee_{o \in A} D_I(o) \sim n^+(A)$

*Proof:*

It suffices to prove the following two:

(a) The query  $\bigvee_{o \in A} D_I(o)$  is a lower bound of  $\{q \mid S(q) \supseteq A\}$ .

(b)  $\bigvee_{o \in A} D_I(o) \in \{q \mid S(q) \supseteq A\}$

(proof of (a))

Let  $x$  denote the query  $\bigvee_{o \in A} D_I(o)$ . We will prove that  $x$  is a lower bound of the set  $\{q \mid S(q) \supseteq A\}$ , i.e. we will prove  $x \leq y$  for each  $y \in \{q \mid S(q) \supseteq A\}$ . Since  $A \subseteq \bar{I}(y)$ , for each  $o \in A$ ,  $o \in \bar{I}(y)$ . Recall that each  $o \in A$  is indexed under the set of terms  $D_I(o)$ . This implies that it must be  $y \geq D_I(o)$  otherwise  $o$  would not be an element of  $\bar{I}(y)$ . Thus  $y \geq \bigvee_{o \in A} D_I(o)$ , i.e.  $y \geq x$ .

(proof of (b))

Each  $o \in A$  is an element of  $I(t)$  for each  $t \in D_I(o)$ . Thus  $o$  is an element of  $I(D_I(o))$ . This implies that  $A \subseteq \bigcup\{I(D_I(o)) \mid o \in A\} = I(\bigvee_{o \in A} D_I(o))$ . Since  $I \sqsubseteq \bar{I}$ , we infer that  $I(\bigvee_{o \in A} D_I(o)) \subseteq \bar{I}(\bigvee_{o \in A} D_I(o))$ , thus  $\bigvee_{o \in A} D_I(o) \in \{q \mid S(q) \supseteq A\}$ .

Since (according to (a)) the query  $\bigvee_{o \in A} D_I(o)$  is a lower bound of  $\{q \mid S(q) \supseteq A\}$  and (according to (b)) it is an element of  $\{q \mid S(q) \supseteq A\}$ , it follows that this query is the glb of  $\{q \mid S(q) \supseteq A\}$ .

◇

**Prop.**  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\} \sim n^-(A)$

*Proof:*

It suffices to prove the following two:

(a) The query  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\}$  is an upper bound of  $\{q \mid S(q) \subseteq A\}$ .

(b)  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\} \in \{q \mid S(q) \subseteq A\}$

(proof of (a))

Let  $x$  denote the query  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\}$ . We will prove that  $x$  is an upper bound of the set  $\{q \mid S(q) \subseteq A\}$ , i.e. we will prove  $x \geq y$  for each  $y \in \{q \mid S(q) \subseteq A\}$ . Suppose that there is an object  $o \in \bar{I}(y)$  such that  $o \notin \bar{I}(x)$ . Since  $o$  is indexed under the set of terms  $D_I(o)$ , it must be  $y \geq D_I(o)$  otherwise  $o$  would not be an element of  $\bar{I}(y)$ . If  $S(D_I(o)) \subseteq A$  then certainly  $o$  would be an element of  $\bar{I}(x)$  by the definition of  $x$ . So, let us suppose that  $S(D_I(o)) \not\subseteq A$ . In this case  $y \geq D_I(o) \Leftrightarrow \bar{I}(y) \supseteq S(D_I(o))$ . As  $S(D_I(o)) \not\subseteq A$  we infer that  $\bar{I}(y) \not\subseteq A$  which is a contradiction. Thus the hypothesis  $o \notin \bar{I}(x)$  is not valid, hence  $x$  is an upper bound of  $\{q \mid S(q) \subseteq A\}$ .

(proof of (b))

If  $S(D_I(o)) \subseteq A$  then  $\bigcup\{S(D_I(o)) \mid o \in A, S(D_I(o)) \subseteq A\} \subseteq A$ . Thus  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\} \in \{q \mid S(q) \subseteq A\}$ .

Since (according to (a)) the query  $\bigvee\{D_I(o) \mid o \in A, S(D_I(o)) \subseteq A\}$  is an upper bound of  $\{q \mid S(q) \subseteq A\}$  and (according to (b)) it is an element of  $\{q \mid S(q) \subseteq A\}$ , it follows that this query is the lub of  $\{q \mid S(q) \subseteq A\}$ .

◇