

## **Abduction for Accessing Information Sources**

**Carlo Meghini\***

*Consiglio Nazionale delle Ricerche  
Istituto della Scienza e delle Tecnologie della Informazione  
Via Giuseppe Moruzzi, 1 - 56124 Pisa, Italy  
carlo.meghini@isti.cnr.it*

**Yannis Tzitzikas**

*Department of Computer Science, University of Crete  
Heraklion, Greece*

**Nicolas Spyratos**

*Université Paris-Sud, Laboratoire de Recherche en Informatique  
Orsay, France*

---

**Abstract.** We consider a general form of information sources, consisting of a set of objects classified by terms arranged in a taxonomy. The query-based access to the information stored in sources of this kind, is plagued with uncertainty, due, among other things, to the possible linguistic mismatch between the user and the object classification. To overcome this uncertainty in all situations in which the user is not finding the desired information and is not willing or able to state a new query, the study proposes to extend the classification, in a way that is as reasonable as possible with respect to the original one. By equating reasonableness with logical implication, the sought extension turns out to be an explanation of the classification, captured by abduction. The problem of query evaluation on information sources extended in this way is studied and a polynomial time algorithm is provided for the general case, in which no hypothesis is made on the structure of the taxonomy. The algorithm is successively specialized on a most common kind of information sources, namely sources whose taxonomy can be represented as a directed acyclic graph. It is shown that query evaluation on extended sources is easier for this kind of sources. Finally, two applications of the method are presented, which capture very important aspects of information access: information browsing and query result ranking.

**Keywords:** Logic-based abduction, information systems, algorithms

---

\*Address for correspondence: CNR ISTI, Via Giuseppe Moruzzi, 1 - 56124 Pisa, Italy

## 1. Introduction

Taxonomies are probably the oldest and most widely used conceptual modeling tool: nowadays, they are employed in Web directories (e.g. in Google and Yahoo!), Content Management (hierarchical structures are used frequently to classify documents), 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 [20]) and in very large collections of objects (e.g. see [35]). The Open Directory Project is currently developing “the largest, most comprehensive human-edited directory of the Web”, based on a taxonomy (<http://dmoz.org>).

More sophisticated conceptual models (including concepts, attributes, relations and axioms) have been used in the information system area since the last three decades; similar models have lately emerged in the context of the Semantic Web [2] and Ontology [15, 1, 36] and have been employed even for meta-tagging in the Web [23, 19]. However, almost all of these models have a backbone consisting of a subsumption hierarchy, in other words, a taxonomy. Furthermore, the design of taxonomies can be done more systematically if done following a faceted approach (e.g. see [34, 33]) and thanks to techniques that have emerged recently [38], taxonomies of compound terms can be also defined in a flexible and systematic manner. The advantages of the taxonomy-based conceptual modeling approach for building large scale peer-to-peer systems that support semantic-based retrieval services have been analyzed and reported in [40, 39].

The extraction of information from an information source centered around a taxonomy of terms, is plagued with uncertainty. From the one hand, the indexing of objects, that is the assignment of a set of terms to each object of the source, presents many difficulties, whether it is performed manually by some expert or automatically by a computer programme. In the former case, subjectivity may play a negative role (e.g. see [43]); in the latter case, automatic classification methods may at best produce approximations. On the other hand, the query formulation process, being linguistic in nature, would require perfect attuning of the system and the user language, an assumption that simply does not hold in open settings such as the Web.

In many contexts, such as directory systems, the above mentioned uncertainty is simply ignored. In others, it is dealt with in a quantitative way, *i.e.* by means of numerical methods. This is for instance the case of textual content management systems, which support the access to collections of textual documents via natural language queries. In these systems, a document index is created for each document, in which a *weight* is assigned to each term; the weight expresses the extent to which the document is deemed to be about the term. The same treatment is applied to each query, deriving an index of the query. Document and query indices are then matched against each other in order to estimate the relevance of the document to the query (e.g. see [3]).

The quantitative approach is nowadays the prevailing one. Its popularity is essentially due to the fact that it provides efficient methods for computing object and query indices, and for matching them to produce a ranking of the objects with respect to their relevance to the query. This is not without a price, though: in these systems, it is very difficult (not to say impossible) for the user to understand the rationale behind the obtained ranking. In hypertext retrieval engines, such as Google, the ranking is driven by link analysis, and its principles reflect some intuition, even though the actual formula to compute page ranking is very complex. But in classical information retrieval systems such a justification is not possible. Moreover, after decades of research and development, it is commonly believed that significant improvements on quantitative methods are practically impossible.

In the present study, we cope with the above mentioned uncertainty by means of a *qualitative* approach, and propose a method that retains the advantages of the quantitative approach (efficiency and ranking), but is grounded on a logical basis. Moreover, our method does not require sophisticated representations of object contents or of lexical and domain knowledge as many semantic-based retrieval models do (see, e.g. [26]) and extends well beyond text retrieval systems, as it will be argued.

From a conceptual point of view, we view an information source as an agent that operates according to an open world philosophy. The agent knows some facts, but it does not interpret these facts as the only ones that hold; the agent is somewhat aware that there could be other facts, compatible with the known ones, that might hold as well, although they are not captured for lack of knowledge. These facts are, indeed, *possibilities*. One way of defining in logical terms the notion of possibility, is to equate it with the notion of *explanation*. That is, the set of terms associated to an object is viewed as a *manifestation* of a phenomenon, the indexing process, for which an explanation is sought, justifying why the index has come to be the way it is. In logic, the reasoning required to infer explanations from given theory and observations, is known as *abduction*. We will therefore resort to abduction in order to define precisely the possibilities that we want our system to be able to handle. In particular, we will define an operation that extends an information source by adding to it a set of (term, object) pairs capturing the sought possibilities, and then study the property of this operation from a computational complexity point of view, for different types of information sources. We will then show that the introduced operation not only helps in dealing with the information extraction in presence of uncertainty, but has also two important applications: browsing and ranking based on a *possibility*-based measure of relevance.

The paper is structured as follows. Before delving into the technical development, the rest of this Section motivates our study by presenting a real-life example and by relating our work to the literature. Section 2 introduces information sources and query-based information access. Section 3, the core of the present study, presents the extension of an information source by means of an abduction-based operator, and sound and complete algorithms to achieve this extension. The complexity of query evaluation on extended information sources is also derived. In Section 4, one important type of information sources is studied, namely hierarchical information sources, whose taxonomy is shaped as a directed acyclic graph (DAG). It is shown that query evaluation on extended hierarchical information sources is computationally easier, and even easier for the information sources whose taxonomy is shaped as a tree. Section 5 presents the above mentioned two applications of the abduction-based method. Section 6 concludes.

## 1.1. Motivation

Let us suppose that a user has issued a query against an information source (hereafter, IS) but the obtained answer does not contain objects that are relevant to the user information needs. Further, let us assume that the user is not willing to replace the query with another one, for instance because of lack of knowledge on the query language or on the taxonomy. In this type of situation, database systems offer practically no support, as they are based on the assumption that users can always articulate their information needs in the form of a query. In an information retrieval setting, a user in the above described situation could use relevance feedback to pinpoint interesting (relevant) objects among those returned by the system, and ask the system to re-evaluate the query taking into account this information; but what if all the displayed objects are not relevant?

Assuming that the IS indeed contains relevant objects, the cause of the problem could be a linguistic mismatch between the user and the indexer. That is, the latter attributes to the IS terms a different

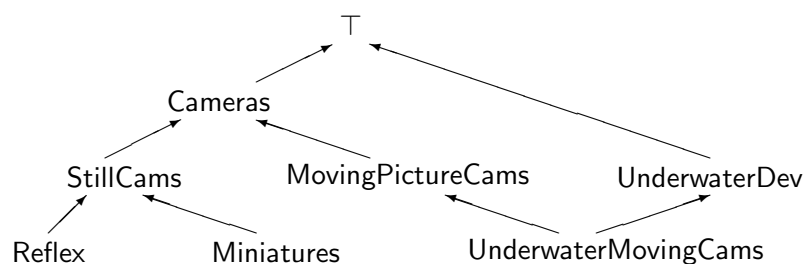


Figure 1. A taxonomy.

meaning than the former, or uses them in a different way. This kind of mismatch may be due to language, such as the case of an Italian user, non-fluent in English, looking for small cameras in the taxonomy in Figure 1; the user may be tempted to ask for **Miniatures**, decidedly an overkill. Or, the mismatch may be due to culture: a technologically less advanced culture may use the term **Miniature** in a different way from a more technologically advanced one. As a result, object classification is inconsistent with the user expectation. On the other hand, even if the user and the indexer were attuned to the same language, a situation as the one described above could occur due to an imprecise classification. Given the widespread usage of automatic indexers based on statistical techniques, these cases would not seem very unlikely to occur. It could also be the case that the sought concept is vague with respect to the considered taxonomy, as the notion of under-water cameras in the taxonomy shown in Figure 1. Clearly, **UnderwaterDev** is too general, and thus include (possibly many) uninteresting objects, while **UnderwaterMovingCams** is too specific, and thus include possibly no interesting object.

The present study aims at developing a method to support the user in all the cases, such as the above ones, in which the user is “close” to the sought objects and is not willing, or not able, to find them by refining the current query. From a user point of view, in these cases it is as if the IS suffers from *incompleteness*: it contains correct information, but not *all* the correct information. Whether this is really the case, or it is due to a lack of knowledge on the user side, it is irrelevant to our method.

To overcome this lack of knowledge, the idea is to expand the index in a way that is as *reasonable* as possible with respect to the original classification of objects. By “reasonable expansion” we mean a *logically grounded* expansion, that is an expansion that *logically implies* the index as it has been created in the first place. Then, the expansion we are talking about is in fact a *logical explanation* of the index. The most general form of explanation in logic is *abduction*, seen as the generation of causes to explain the observed effects of known laws. In our case, the known laws are the (sentences of the) IS taxonomy, the observed effects are the contents of the index, while the cause is the sought index expansion.

By exploiting abduction, we will therefore define a method for helping the user access the IS in a query-less way, addressing the situations described at the beginning of this Section. The method will turn out to be suitable to support also another important form of information access, namely browsing. In browsing, the user will only have to (a) select an initial object which possesses the features he is interested in, and (b) ask the system to browse similar objects. In response, the system will translate the user desire into a query, formed by the terms that best *explain* the terms describing the selected object. This query is then evaluated and the result offered to the user as an answer to his browsing request. The process may be iterated at will, until all objects have been browsed. Abduction is used in this process to compute the query, given the nature of explanation of the terms which make it up.

A fundamental limitation of qualitative methods for information access is their providing unstructured answers, as opposed to the ranking offered by quantitative methods. We can overcome this limitation by observing that ranking is complementary to browsing: in the latter, objects are displayed in time, while in the former they are displayed in space. In other words, during browsing the user discovers sets of objects as he proceeds in the navigation; by displaying the so discovered objects all at once, ordered according to the step at which they would be discovered, a rank is generated, which reflects closeness to the initial objects. By choosing as initial objects those returned by a given query, the rank reflects closeness to the query. On the basis of this observation, we then propose a method for ranking *in a qualitative way* the results of the queries posed to an IS. Since ranking is based on browsing, which is an application of abduction, also ranking turns out to be an application of abduction.

For generality, the problem of abduction-based expansion will first be tackled for arbitrary ISs; the derived results will successively be specialized to ISs having taxonomies like the one in Figure 1.

## 1.2. Related Work

From an information retrieval (IR) point of view, our method subscribes to *recall enhancing* mechanisms. Indeed, the ISs considered in this study resemble those managed by IR systems which employ the Boolean retrieval model (see [3] for a review) and exploit lexical ontologies or word-based thesauri [20] (like WordNet [7] and Roget's thesaurus) for expanding queries with synonyms, hyponyms and related terms in order to improve recall (e.g. see [32], [24], [25], [17]). However, IR techniques are applicable only if the objects of the domain have a textual content, while this is not a prerequisite of our approach. Another remarkable difference between IR systems and ISs is that the taxonomies employed by the former usually do not admit the semantic interpretation that we use for the latter. Lexical ontologies like WordNet [7] are structured using lexical relations (synonymy, hyponymy, antonymy) which are not semantic relations. For instance, according to Wordnet, *window* is subsumed by *opening* and by *panel*. However, every *window* is not a *panel* and an *opening*, thus extensional subsumption does not hold here (for more about this problem see [16]).

Logical models of IR have experienced some popularity some years ago [31]. These models are based on deduction, whether classical [27, 4] or non-classical [26, 14, 31] logics are employed, and are thus substantially different from the model assumed in this study. On the other hand, the usage of abduction in an IR setting is inextricably tied with IR models based on non-logical [18] or quantitative [30] information models, so also in this case there is no evident connection with our model.

In databases, abduction has been mainly employed as a tool for modelling database [21] or view [9] update. Logical approaches to information systems based on abduction have been proposed in the context of non-classical logics [6], or on various forms of classical logic, ranging from logic programming [42] to description logics [8]. Our model is based on classical logic, and the terms representing object contents are equivalent to unary predicates. However, since we will consider each object of the IS separately, the most suitable model of abduction for our purposes is the one presented in [11], based on propositional logic. This model is briefly recalled and related to that of an IS at the beginning of Section 3. To the best of our knowledge, no previous use of abduction for the purposes presented in this paper, exists.

A preliminary, short version of this work has been reported in [28].

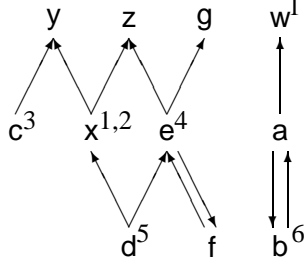


Figure 2. An information source

## 2. Information Sources

The basic notion of information source (IS) is defined as follows.

**Definition 1:** An *information source*  $S$  is a 4-tuple  $S = \langle T, K, Obj, I \rangle$  where

- $T$ , the *terminology*, is a finite set of symbols called *terms*;
- $K$ , the *taxonomy*, is a set of conditionals on  $T$ , *i.e.* formulas of the form  $p \rightarrow q$  where  $p$  and  $q$  are different terms;
- $Obj$ , the *domain*, is a finite set of objects disjoint from  $T$ ;
- $I$ , the *interpretation*, is a relation from  $T$  to  $Obj$ , that is  $I \subseteq T \times Obj$ . □

From an intuitive point of view, an IS must be viewed as a set of propositional theories, one for each object in the domain, which share the same language (terminology) and the same underlying theory (taxonomy). The role of the interpretation is to identify the propositional letters that are true of each object.

As customary, we will treat the relation  $I$  as a function from terms to sets of objects and, where  $t$  is a term in  $T$ , will write  $I(t)$  to denote the set  $I(t) = \{o \in Obj \mid (t, o) \in I\}$ . We will call  $I(t)$  the *interpretation* of term  $t$ . Dually, given an object  $o \in Obj$ , the *index of  $o$  in  $S$* ,  $ind_S(o)$ , is given by the set of terms which have  $o$  in their interpretation:

$$ind_S(o) = \{t \in T \mid (t, o) \in I\}.$$

Finally, the *context of  $o$  in  $S$* ,  $C_S(o)$ , is defined as:  $C_S(o) = ind_S(o) \cup K$ . For any object  $o$ ,  $C_S(o)$  consists of terms and simple conditionals that collectively form all the knowledge about  $o$  that  $S$  has.

**Example 1:** Throughout the paper, we will use as an example the IS  $S$  given in the right-hand side of Figure 2. The left-hand side of the Figure graphically illustrates  $S$ ; the interpretation of each term is shown, without braces, on the top-right of the term, *i.e.*  $ind_S(1) = \{x, w\}$ . □

In this study, we focus on ISs which satisfy an intuitive minimality criterion. In order to introduce this criterion and keep the paper self-contained, a few basic notions from propositional logic [12] are now recalled. Given a set of propositional variables  $P$ , a *truth assignment for  $P$*  is a function mapping  $P$  to the true and false truth values, respectively denoted by  $\mathbf{T}$  and  $\mathbf{F}$ . A truth assignment  $V$  *satisfies* a sentence

$\sigma$  of the propositional calculus,  $V \models \sigma$ , if  $\sigma$  is true in  $V$ , according to the classical truth valuation rules. A set of sentences  $\Sigma$  *logically implies* the sentence  $\alpha$ ,  $\Sigma \models \alpha$ , iff every truth assignment which satisfies every sentence in  $\Sigma$  also satisfies  $\alpha$ .

The *instance set* of an object  $o$  in an IS  $S$ , denoted as  $N_S(o)$ , is the set of terms that are logically implied by the context of  $o$  in  $S$

$$N_S(o) = \{t \in T \mid C_S(o) \models t\}.$$

For each term  $t$  in  $N_S(o)$ , we will say that  $o$  is an *instance* of  $t$ . Clearly,  $ind_S(o) \subseteq N_S(o)$ , therefore  $o$  is an instance of each term in  $ind_S(o)$ . We can now define the minimality criterion mentioned above.

**Definition 2:** The index of an object  $o$  in an IS  $S$ ,  $ind_S(o)$ , is *non-redundant* iff for all  $A \subseteq Obj$ ,

$$A \subset ind_S(o) \text{ implies } \{v \in T \mid A \cup K \models v\} \subset N_S(o).$$

An IS is *non-redundant* if all its indices are non-redundant. □

In practice, the index of an object is non-redundant if no term in it can be removed without loss of information. It can be easily verified that the IS introduced in the previous example is non-redundant. From now on, we will consider “IS” as a synonym of “non-redundant IS”.

## 2.1. Querying Information Sources

We next introduce the query language for extracting information from an IS in the traditional question-answering way.

**Definition 3:** Given a terminology  $T$ , the *query language for  $T$* ,  $\mathcal{L}_T$ , is defined by the following grammar, where  $t$  is a term in  $T$ :

$$q, q' ::= t \mid q \wedge q' \mid q \vee q' \mid \neg q \mid (q)$$

Any expression in  $\mathcal{L}_T$  is termed a *query*. □

The answer to queries is defined in terms of satisfaction in the model of an object context, the *truth model*, realizing a closed-world reading of an IS.

**Definition 4:** Given an IS  $S = (O, U)$ , for every object  $o \in Obj$ , the *truth model of  $o$  in  $S$* ,  $V_{o,S}$ , is the truth assignment for  $T$  defined as follows, for each term  $t \in T$  :

$$V_{o,S}(t) = \begin{cases} \mathbf{T} & \text{if } t \in N_S(o) \\ \mathbf{F} & \text{otherwise} \end{cases}$$

Given a query  $\varphi$  in  $\mathcal{L}_T$ , the *answer of  $\varphi$  in  $S$*  is the set of objects whose truth model satisfies the query:

$$ans(\varphi, S) = \{o \in Obj \mid V_{o,S} \models \varphi\}.$$

□

**Example 2:** Table I shows the truth model the objects in the IS introduced in Example 1. The answer to the query  $z$  in this IS,  $ans(z, S)$ , includes object 1, since  $V_{1,S}(z) = \mathbf{T}$ , as well as objects 2, 4 and 5, for the same reason. □

$t$	a	b	c	d	e	f	g	x	y	z	w
$V_{1,S}(t)$	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>
$V_{2,S}(t)$	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>F</b>
$V_{3,S}(t)$	<b>F</b>	<b>F</b>	<b>T</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>F</b>	<b>F</b>
$V_{4,S}(t)$	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>F</b>
$V_{5,S}(t)$	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>F</b>
$V_{6,S}(t)$	<b>T</b>	<b>T</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>T</b>

Table I. The truth model of each object in the information source  $S$ 

Following Definition 4, query evaluation requires the computation of the truth model of each object  $o$ , which in turn requires deciding whether each query term is logically implied by the object context  $C_S(o)$ . Computing propositional logical implication is in general a difficult task. However, the specific form of the propositional theories considered in this study, makes this computation much simpler, as the remainder of this Section shows. In order to devise an efficient query evaluation procedure, we will resort to graph theoretic concepts.

The *term graph* of an IS  $S = \langle T, K, Obj, I \rangle$  is the directed graph  $G_S = \langle T, E \rangle$ , such that  $(t, t') \in E$  iff  $t \rightarrow t'$  is in  $K$ . The left-hand side of Figure 2 shows indeed the term graph of the example IS  $S$ . For simplicity, we will use “term” also to refer to a vertex of the term graph. A path of length  $k \geq 0$  from a vertex  $u$  to a vertex  $u'$  in  $G_S$  is a sequence  $\langle v_0, v_1, \dots, v_k \rangle$  of vertices such that  $u = v_0$  and  $u' = v_k$ . There is always a 0-length path from  $u$  to  $u$ . The *tail of a term*  $t$  in  $G_S$ ,  $tail(t)$ , is the set of terms that can be reached from  $t$  by walking the graph edges backward, that is:

$$tail(t) = \{u \in T \mid \text{there exists a path from } u \text{ to } t \text{ in } G_S\}$$

**Lemma 1:** Given an IS  $S$ , a set of terms  $A \subseteq T$  and a term  $t \in T$ ,  $A \cup K \models t$  iff there is a path in  $G_S$  from a term in  $A$  to  $t$ .

*Proof:* ( $\rightarrow$ ) By induction on the length  $l$  of the path. For  $l = 0$ , the path is  $(t, t)$  which means that  $t \in A$ , so  $A \models t$  and by the monotonicity of  $\models$ ,  $A \cup K \models t$ . Suppose the Proposition holds for  $l = k > 0$ . Let  $\langle u_1, \dots, u_k, t \rangle$  be a path in  $G_S$  with  $u_1 \in A$ , and  $M$  be a model of  $A \cup K$ . By definition of  $G_S$ ,  $u_k \rightarrow t$  is in  $K$ . Since  $M$  is a model of  $A \cup K$  and by the induction hypothesis,  $M(u_k) = \mathbf{T}$ , it follows that  $M(t) = \mathbf{T}$ . Hence  $M$  is a model of  $t$ , so  $A \cup K \models t$ .

( $\leftarrow$ ) Suppose  $A \cup K \models t$  yet  $G_S$  contains no path from a term in  $A$  to  $t$ . Let  $M$  be the interpretation defined as follows:

$$M(v) = \begin{cases} \mathbf{T} & \text{if there is a path from a term in } A \text{ to } v \text{ in } G_S \\ \mathbf{F} & \text{otherwise} \end{cases}$$

By definition, any term in  $A$  is  $\mathbf{T}$  in  $M$ , so  $M$  is a model of  $A$ . Let  $p \rightarrow q$  be a conditional in  $K$ . If  $M(p)$  is  $\mathbf{F}$ , then the conditional is true in  $M$ . If, on the other hand,  $M(p) = \mathbf{T}$ , then there is a path from a term  $u$  in  $A$  to  $p$ ; since  $G_S$  contains the edge  $(p, q)$ , the path from  $u$  to  $p$  can be extended to  $q$ , and by definition  $M(q) = \mathbf{T}$ . Also in this case  $M$  satisfies the conditional. So  $M$  also satisfies every conditional in  $K$ , hence  $M$  is a model of  $A \cup K$ . Since  $A \cup K \models t$ ,  $M$  is also a model of  $t$ , that is  $M(t) = \mathbf{T}$ . However, by construction,  $M(t) = \mathbf{F}$ , a contradiction.  $\square$

We can now prove the following Proposition.

**Proposition 1:** For all ISs  $S$  and queries  $\varphi \in \mathcal{L}_T$ ,  $ans(\varphi, S) = \alpha_S(\varphi)$ , where  $\alpha_S$  is the *solver* of the IS  $S$ , defined as follows:

$$\begin{aligned}\alpha_S(t) &= \bigcup \{I(u) \mid u \in tail(t)\} \\ \alpha_S(q \wedge q') &= \alpha_S(q) \cap \alpha_S(q') \\ \alpha_S(q \vee q') &= \alpha_S(q) \cup \alpha_S(q') \\ \alpha_S(\neg q) &= Obj \setminus \alpha_S(q)\end{aligned}$$

*Proof:* By structural induction on the query  $\varphi$ .

**Case**  $\varphi = t$ . By definition,  $ans(t, S) = \{o \in Obj \mid V_{o,S} \models t\}$ .  $V_{o,S} \models t$  iff  $C_S(o) \models t$  iff  $ind_S(o) \cup K \models t$  iff (by the previous Lemma) there is a path in  $G_S$  from a term  $u \in ind_S(o)$  to  $t$ , which is the same to say that  $o \in I(u)$  for some  $u \in tail(t)$ . So,  $ans(t, S) = \{o \in Obj \mid \exists u \in tail(t) \text{ such that } o \in I(u)\}$ , therefore

$$ans(t, S) = \bigcup \{I(u) \mid u \in tail(t)\}.$$

**Case**  $\varphi = q \wedge q'$ .  $ans(q \wedge q', S) = \{o \in Obj \mid V_{o,S} \models q \wedge q'\} = \{o \in Obj \mid V_{o,S} \models q \text{ and } V_{o,S} \models q'\} = \{o \in Obj \mid V_{o,S} \models q\} \cap \{o \in Obj \mid V_{o,S} \models q'\} = ans(q, S) \cap ans(q', S)$ .

**Case**  $\varphi = q \vee q'$ .  $ans(q \vee q', S) = \{o \in Obj \mid V_{o,S} \models q \vee q'\} = \{o \in Obj \mid V_{o,S} \models q \text{ or } V_{o,S} \models q'\} = \{o \in Obj \mid V_{o,S} \models q\} \cup \{o \in Obj \mid V_{o,S} \models q'\} = ans(q, S) \cup ans(q', S)$ .

**Case**  $\varphi = \neg q$ .  $ans(\neg q, S) = \{o \in Obj \mid V_{o,S} \models \neg q\} = \{o \in Obj \mid V_{o,S} \not\models q\} = Obj \setminus \{o \in Obj \mid V_{o,S} \models q\} = Obj \setminus ans(q, S)$ .  $\square$

**Example 3:** In the example IS  $S$ , the term  $z$  can be reached in the term graph by each of the following terms:  $z, x, d, e, f$ . Hence,  $tail(z) = \{z, x, d, e, f\}$ . According to the last Proposition, then:  $ans(z, S) = \alpha_S(z) = I(z) \cup I(x) \cup I(d) \cup I(e) \cup I(f) = \{1, 2, 4, 5\}$ .  $\square$

**Proposition 2:** For all ISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $\alpha_S(t)$  can be computed in  $O(|T| \cdot |Obj| \cdot \lg |Obj|)$  time.

*Proof:* From Proposition 1,  $\alpha_S(t)$  can be computed by the following steps: (a) to derive  $tail(t)$  by searching the term graph in order to identify every vertex that is backward reachable from  $t$ ; (b) to access the interpretation of each term in  $tail(t)$ ; and (c) to compute the union of the involved interpretations. The time complexity of step (a) is  $O(|T|^2)$ , corresponding to the case in which every term is backward reachable from  $t$  in the term graph. We assume that step (b) can be performed in constant time, which is negligible with respect to the other values at stake. Let us now consider step (c). By adopting a merge-sort strategy, the union between two interpretations can be performed in  $n \lg n$  time, where  $n$  is the total size of the two interpretations. Since in the worst case the union of  $|T|$  interpretations must be computed, and each interpretation is the whole set  $Obj$ , we have a  $O(|T| \cdot |Obj| \cdot \lg(|Obj| \cdot |T|))$  time complexity for step (c). Overall, the upper bound for evaluating single-term queries is therefore  $O(|T|^2 + |T| \cdot |Obj| \cdot \lg(|Obj| \cdot |T|))$ . Since the size of the domain is expected to be significantly larger than the size of the terminology, *i.e.*  $|T| \ll |Obj|$ , the sought upper bound for singles-term queries evaluation is  $O(|T| \cdot |Obj| \cdot \lg |Obj|)$ .  $\square$

### 3. Extended Information Sources

This Section presents the main technical developments of the paper, and is divided in four main parts: In the first part, extended information sources are defined on the basis of abduction. In the second part, the problem of querying such sources is tackled, and the algorithm  $\beta 1$  is derived. In the third part, optimization is considered, leading to the derivation of the algorithm  $\beta 2$ . In the fourth and last part, the query-less information access mechanism which motivated our study is outlined and illustrated by an example.

#### 3.1. Propositional Abduction Problems

As announced in Section 1.2, the model of abduction that we adopt is the one presented in [11], now briefly recalled to make the paper self-contained. Let  $\mathcal{L}_V$  be the language of propositional logic over a finite alphabet  $V$  of propositional variables. A *propositional abduction problem* is a tuple  $\mathcal{A} = \langle V, H, M, Th \rangle$ , where  $H \subseteq V$  is the set of hypotheses,  $M \subseteq V$  is the manifestation, and  $Th \subseteq \mathcal{L}_V$  is a consistent theory.  $S \subseteq H$  is a solution (or explanation) for  $\mathcal{A}$  iff  $Th \cup S$  is consistent and  $Th \cup S \models M$ .  $Sol(\mathcal{A})$  denotes the set of the solutions to  $\mathcal{A}$ .

In the context of an IS  $S$ , we will consider each object separately. Thus,

- the terms in  $T$  play both the role of the propositional variables  $V$  and of the hypotheses  $H$ , as there is no reason to exclude *a priori* any term from an explanation;
- the taxonomy  $K$  plays the role of the theory  $Th$ ;
- the role of manifestation is played by the index of the object.

Consequently, we have the following:

**Definition 5:** Given an IS  $S = \langle T, K, Obj, I \rangle$  and object  $o \in Obj$ , the *propositional abduction problem for  $o$  in  $S$* ,  $\mathcal{A}_S(o)$ , is the propositional abduction problem  $\mathcal{A}_S(o) = \langle T, T, ind_S(o), K \rangle$ .  $\square$

The solutions to  $\mathcal{A}_S(o)$  are given by:

$$Sol(\mathcal{A}_S(o)) = \{A \subseteq T \mid K \cup A \models ind_S(o)\}$$

where the consistency requirement on  $K \cup A$  has been omitted since for no taxonomy  $K$  and set of terms  $A$ ,  $K \cup A$  can be inconsistent. Usually, certain explanations are preferable to others, a fact that is formalized in [11] by defining a preference relation  $\preceq$  over  $Sol(\mathcal{A})$ . Letting  $a \prec b$  stand for  $a \preceq b$  and  $b \not\preceq a$ , the set of preferred solutions is given by:

$$Sol_{\preceq}(\mathcal{A}) = \{S \in Sol(\mathcal{A}) \mid \nexists S' \in Sol(\mathcal{A}) : S' \prec S\}.$$

Also in the present context a preference relation is desirable, satisfying criteria that reflect the goals of our framework. Here are these criteria, in order of decreasing importance:

1. explanations including only terms in the manifestation are less preferable, as they do not provide any additional information;

2. explanations altering the behavior of the IS to a minimal extent are preferable; this requirement acts in the opposite direction of the previous one, by making preferable solutions that, if incorporated in the IS, minimize the differences in behavior between the so extended IS and the original one;
3. between two explanations that equally alter the behavior of the IS, the simpler one is to be preferred. As explanations are sets, it is natural to equate simplicity with smallness in size.

All the above criteria can be expressed in terms of the effects produced by the extension of an IS. In order to indicate these effects, we will use the term “perturbation”, defined next.

**Definition 6:** Given an IS  $S = \langle T, K, Obj, I \rangle$ , an object  $o \in Obj$  and a set of terms  $A \subseteq T$ , the *perturbation of  $A$  on  $S$  with respect to  $o$* ,  $pert_o(A)$  is given by:

$$pert_o(A) = \{t \in T \mid (C_S(o) \cup A) \models t \text{ and } C_S(o) \not\models t\}$$

that is,  $pert_o(A)$  is the set of additional terms in the instance set of  $o$  once the index of  $o$  is extended with the terms in  $A$ .  $\square$

We notice that the *pert* operator is monotonic, that is  $A \subseteq B$  implies  $pert_o(A) \subseteq pert_o(B)$ , for any object  $o \in Obj$ . We can now define the preference relation over solutions of the above stated abduction problem.

**Definition 7:** Given an IS  $S = \langle T, K, Obj, I \rangle$ , an object  $o \in Obj$  and two solutions  $A$  and  $A'$  to the problem  $\mathcal{A}_S(o)$ ,  $A \preceq A'$  if either of the following holds:

1.  $pert_o(A') = \emptyset$
2.  $0 < |pert_o(A)| < |pert_o(A')|$
3.  $0 < |pert_o(A)| = |pert_o(A')|$  and  $A \subseteq A'$ .  $\square$

The strict correspondence between the clauses in the last Definition and the criteria previously set for the preference relation should be evident. Solutions having an empty perturbation are obviously subsets of the instance set of the object, therefore the first condition of the last Definition captures the first of the three criteria. The second condition establishes preference for solutions that minimize the number of terms that change their truth value from **F** to **T** in the truth model of the object, and thus alter the behavior of the IS *with respect to query answering* to a minimal extent. Between two solutions producing the same alteration, the third condition makes preferable the smaller in size, and so simplicity, criterion number three, is implemented.

$(Sol(\mathcal{A}_S(o)), \preceq)$  is a pre-order: reflexivity and transitivity can be easily verified, while to see that antisymmetry does not hold it suffices to consider any two different subsets  $A$  and  $A'$  of the instance set of  $o$ , i.e.  $A, A' \subseteq N_S(o)$ ,  $A \neq A'$ ; we have:  $pert_o(A) = pert_o(A') = \emptyset$ , hence by 1 in Definition 7,  $A \preceq A'$  and  $A' \preceq A$ . If subsets of the instance set are excluded from the solutions, then  $\preceq$  induces a partial order, as clause 1 in Definition 7 never applies; this is not a total order, as the examples below will show.

We now introduce the notion of *extension* of an IS. The idea is that an extended IS (EIS for short) includes, for each object, the terms of the original index plus those added through the abduction process illustrated above. However, in so doing, non-redundancy may be compromised, due to the following casuses:

1. As it will be shown later (Proposition 7), a solution to an abduction problem may contain taxonomic cycles, and including a whole cycle in the index of an object clearly violates non-redundancy (all terms in the cycle but one can be removed without losing information).
2. A term in a solution may be a direct descendant of a term in the index. The coexistence of these two terms in the new index violates redundancy, thus the latter can be added only if the former is removed.

In order to cope with the former problem, we introduce the operator  $\rho$ , which takes as input a set of terms and replaces each cycle occurring in it by any term in the cycle. To this end, we assume that the terminology  $T$  of an IS is totally ordered by a binary relation  $<_T$ . Given any set of terms  $A \subseteq T$ ,  $l(A)$  denotes the least element of  $A$  according to the total order  $<_T$ .  $\rho$  is then defined as follows: Given an IS  $S = \langle T, K, Obj, I \rangle$ , let  $A \subseteq T$  be any set of terms containing  $n \geq 0$  disjoint sets  $C_1, \dots, C_n$ , such that the terms in  $C_i$  make up a cycle in the term graph  $G_S$ , for all  $1 \leq i \leq n$ . Then,

$$\rho(A) = (A \setminus \bigcup \{C_i \mid 1 \leq i \leq n\}) \cup \{l(C_1), \dots, l(C_n)\}.$$

The assumption on the disjointness of the  $C_i$ 's is without loss of generality, because if two cycles intersect they can be replaced by a single cycle including both.

In order to cope with the latter problem, we introduce a special union operator  $\sqcup$  which takes as input two interpretations and adds each pair  $(t, o)$  of the second interpretation to the first interpretation after removing any pair  $(u, o)$  in the first interpretation such that  $t \rightarrow u \in K$ . Formally,

$$I_1 \sqcup I_2 = I_2 \cup \{(t, o) \in I_1 \mid \text{for no pair } (v, o) \in I_2, v \rightarrow t \in K\}.$$

We can now define EISs.

**Definition 8:** Given an IS  $S = \langle T, K, Obj, I \rangle$  and an object  $o \in Obj$ , the *abduced index* of  $o$ ,  $abind_S(o)$ , is given by:

$$abind_S(o) = \bigcup Sol_{\leq}(\mathcal{A}_S(o)) \setminus ind_S(o).$$

The *abduced interpretation* of  $S$ ,  $I^+$ , is given by

$$I^+ = I \sqcup \{(t, o) \mid o \in Obj \text{ and } t \in \rho(abind_S(o))\}.$$

Finally, the *extended IS*,  $S^e$ , is given by  $S^e = \langle T, K, Obj, I^+ \rangle$ . □

### 3.2. Querying Extended Information Sources

A key role in solving propositional abduction problems is played by single-term solutions (STSs, for short) introduced next.

**Definition 9:** Given an IS  $S = \langle T, K, Obj, I \rangle$ , an object  $o \in Obj$  and a term  $t \in T \setminus N_S(o)$ , the *single-term solution* of  $t$  is the set  $\mu_o(t)$  given by:

$$\mu_o(t) = \{t\} \cup (ind_S(o) \setminus \sigma(t))$$

where  $\sigma(t) = \{u \in T \mid \text{there is a path in } G_S \text{ from } t \text{ to } u\}$ . □

The next Lemma states some useful properties of STSs.

**Lemma 2:** For any IS  $S = \langle T, K, Obj, I \rangle$ , object  $o \in Obj$  and term  $t \in T \setminus N_S(o)$  :

1.  $\mu_o(t)$  is a solution to  $\mathcal{A}_S(o)$
2.  $pert_o(\mu_o(t)) = \sigma(t) \setminus N_S(o)$
3.  $\mu_o(t)$  has the smallest perturbation among the solutions to  $\mathcal{A}_S(o)$  including  $t$
4. for any term  $v \in T \setminus N_S(o)$ ,  $\{v\} \cup K \models t$  implies  $pert_o(\mu_o(t)) \subseteq pert_o(\mu_o(v))$ .

*Proof:* (1) By definition of  $\models$ ,  $(ind_S(o) \setminus \sigma(t)) \models (ind_S(o) \setminus \sigma(t))$ . By Lemma 1 and the definition of  $\sigma(t)$ ,  $\{t\} \cup K \models \sigma(t)$ . By combining the last two relationships, we have:

$$(ind_S(o) \setminus \sigma(t)) \cup \{t\} \cup K \models (ind_S(o) \setminus \sigma(t)) \cup \sigma(t) \supseteq ind_S(o).$$

Hence  $\mu_o(t) \cup K \models ind_S(o)$ , and therefore  $\mu_o(t) \in Sol(\mathcal{A}_S(o))$ .

(2) We have:

$$\begin{aligned} pert_o(\mu_o(t)) &= \{u \in T \mid (C_S(o) \cup \mu_o(t)) \models u \text{ and } C_S(o) \not\models u\} \\ &= \{u \in T \mid (ind_S(o) \cup K \cup \{t\} \cup (ind_S(o) \setminus \sigma(t))) \models u\} \setminus N_S(o) \\ &= \{u \in T \mid (ind_S(o) \cup K \cup \{t\}) \models u\} \setminus N_S(o) \\ &= \{u \in T \mid (K \cup \{t\}) \models u\} \setminus N_S(o) \\ &= \sigma(t) \setminus N_S(o) \end{aligned}$$

(3) Let  $A \in Sol(\mathcal{A}_S(o))$ ,  $A \neq \mu_o(t)$ , such that  $t \in A$ . It follows that  $\{t\} \subseteq A$ . We must prove that  $pert_o(\mu_o(t)) \subseteq pert_o(A)$ . It can be verified that, for any set of terms  $X \subseteq T$ ,  $pert_o(X) = pert_o(X \setminus N_S(o))$ , and that  $pert_o$  is monotonic in  $T \setminus N_S(o)$ , that is  $A, A' \subseteq T \setminus N_S(o)$  and  $A \subseteq A'$  imply  $pert_o(A) \subseteq pert_o(A')$ . Putting all together,

$$pert_o(\mu_o(t)) = pert_o(\mu_o(t) \setminus N_S(o)) = pert_o(\{t\}) \subseteq pert_o(A)$$

(4) There can be two cases: (i)  $t \in N_S(o)$ . Then,  $pert_o(\mu_o(t)) = \emptyset \subseteq pert_o(\mu_o(v))$ . (ii)  $t \notin N_S(o)$ . Then, by the hypothesis,  $t \in \sigma(v)$ , therefore  $\sigma(t) \subseteq \sigma(v)$ . It follows  $\sigma(t) \setminus N_S(o) \subseteq \sigma(v) \setminus N_S(o)$ , so by the monotonicity of  $pert_o$  in  $T \setminus N_S(o)$ ,  $pert_o(\mu_o(t)) \subseteq pert_o(\mu_o(v))$ .  $\square$

**Example 4:** Let us consider again the IS  $S$  introduced in Example 1, and the problem  $\mathcal{A}_S(1)$ . The manifestation is given by  $ind_S(1) = \{w, x\}$  while  $N_S(1) = \{x, y, z, w\}$ . Table II gives, for each term in  $T \setminus N_S(1)$ , the  $\sigma$  value, the corresponding STS and its perturbation.  $\square$

The next Proposition gives the solver  $\alpha_{S^e}$  for an EIS  $S^e$ , playing for EISs the same role that Proposition 1 plays for ISs. The Proposition lays the foundations for the further developments of the paper.

**Proposition 3:** For all ISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,

$$ans(t, S^e) = \alpha_{S^e}(t) = \alpha_S(t) \cup \beta(t)$$

where  $\beta(t) = \{o \in Obj \mid t \in abind_S(o)\}$ .

*Proof:* By definition,  $ans(t, S^e) = \{o \in Obj \mid V_{o, S^e} \models t\}$ .  $V_{o, S^e} \models t$  iff  $C_{S^e}(o) \models t$  iff  $ind_{S^e}(o) \cup K \models t$  iff  $\{t \in T \mid (t, o) \in I^+\} \cup K \models t$  iff  $\{t \in T \mid (t, o) \in I \text{ or } t \in abind_S(o)\} \cup K \models t$

$t$	$\sigma(t)$	$\mu_o(t)$	$pert_o(\mu_o(t))$
a	{w, a, b}	{a, x}	{a, b}
b	{w, a, b}	{b, x}	{a, b}
c	{c, y}	{c, x, w}	{c}
d	{d, x, y, z, e, f, g}	{d, w}	{d, e, f, g}
e	{e, f, z, g}	{e, x, w}	{e, f, g}
f	{f, e, z, g}	{f, x, w}	{e, f, g}
g	{g}	{g, x, w}	{g}

Table II. The single-term solutions of  $\mathcal{A}_S(1)$  and their perturbations

iff  $ind_S(o) \cup abind_S(o) \cup K \models t$ . Hence,  $ans(t, S^e) = \{o \in Obj \mid ind_S(o) \cup K \models t\} \cup \{o \in Obj \mid abind_S(o) \cup K \models t\} = \alpha_S(t) \cup \{o \in Obj \mid abind_S(o) \cup K \models t\}$ . We claim that

$$\alpha_S(t) \cup \{o \in Obj \mid abind_S(o) \cup K \models t\} = \alpha_S(t) \cup \{o \in Obj \mid t \in abind_S(o)\}.$$

We first show that the left-hand side is contained in the right-hand side, *i.e.* for all objects  $o \in Obj$ ,  $o \in \alpha_S(t) \cup \{o \in Obj \mid abind_S(o) \cup K \models t\}$  implies  $o \in \alpha_S(t) \cup \{o \in Obj \mid t \in abind_S(o)\}$ . This is obvious if  $o \in \alpha_S(t)$ . Let  $o$  be such that  $abind_S(o) \cup K \models t$ . By Lemma 1, for some term  $v \in T$ ,  $v \in abind_S(o)$  and there exists a path in  $G_S$  from  $v$  to  $t$ . If  $v = t$ , then  $t \in abind_S(o)$  and the claim holds. Now let us suppose  $v \neq t$ . There can be two cases: (1)  $v \in N_S(o)$ . By definition of  $N_S(o)$  and Lemma 1, there exists a path in  $G_S$  from a term  $w \in ind_S(o)$  to  $v$ . But then there is a path in  $G_S$  from  $w$  to  $t$ , and by the same Lemma,  $ind_S(o) \cup K \models t$ . This implies  $o \in \alpha_S(t)$ . (2)  $v \notin N_S(o)$ . If  $t \in N_S(o)$ ,  $o \in \alpha_S(t)$ , so let us assume  $t \notin N_S(o)$ . Since  $v \in abind_S(o)$ , by Lemma 2.(3)  $\mu_o(v) \in Sol_{\leq}(\mathcal{A}_S(o))$ , hence  $|pert_o(\mu_o(v))| \leq |pert_o(\mu_o(t))|$ . Since  $\{v\} \cup K \models t$ , by Lemma 2.(4)  $|pert_o(\mu_o(v))| \geq |pert_o(\mu_o(t))|$ . It follows that  $|pert_o(\mu_o(v))| = |pert_o(\mu_o(t))|$  and therefore  $\mu_o(t) \in Sol_{\leq}(\mathcal{A}_S(o))$  too. By definition of  $abind_S(o)$  and since  $t \notin N_S(o)$ ,  $t \in abind_S(o)$ .

We now show that the right-hand side is contained in the left-hand side. Again, this is obvious if  $o \in \alpha_S(t)$ . Let  $o$  be such that  $t \in abind_S(o)$ . Then,  $abind_S(o) \models t$ . By the monotonicity of  $\models$ ,  $abind_S(o) \cup K \models t$ .  $\square$

The last Proposition indicates that, in order to answer queries against an EIS, one needs to compute the set  $\beta(t)$ . The properties of STSs stated in Lemma 2 allows us to derive  $abind_S(o)$  for any propositional abduction problem.

**Proposition 4:** Given an IS  $S = \langle T, K, Obj, I \rangle$  and an object  $o \in Obj$ , let  $d_o$  be the smallest size of non-empty perturbations of the solutions to  $\mathcal{A}_S(o)$ , that is:

$$d_o = \min\{|pert_o(A)| \mid A \in Sol(\mathcal{A}_S(o)) \text{ and } |pert_o(A)| > 0\}$$

Then

$$abind_S(o) = \{t \in T \setminus N_S(o) \mid |\sigma(t) \setminus N_S(o)| = d_o\}$$

*Proof:* ( $\rightarrow$ ) We first show that the left-hand side is contained in the right-hand side. By Definition of  $abind_S(o)$ ,  $t \in abind_S(o)$  implies  $t \notin N_S(o)$ . Moreover, it implies that there exists a solution  $A$  in

$Sol(\mathcal{A}_S(o))$  containing  $t$ , such that  $|pert_o(A)| = d_o$ . By Lemma 2.(3),  $A = \mu_o(t)$ . By Lemma 2.(2),  $pert_o(\mu_o(t)) = \sigma(t) \setminus N_S(o)$ , hence  $|\sigma(t) \setminus N_S(o)| = d_o$ .

( $\leftarrow$ )By Lemma 2.(1),  $\mu_o(t) \in Sol(\mathcal{A}_S(o))$ . By the hypothesis  $|\sigma(t) \setminus N_S(o)| = d_o$  and Lemma 2.(2), it follows that  $|pert_o(\mu_o(t))| = d_o$  and therefore  $\mu_o(t) \in Sol_{\leq}(\mathcal{A}_S(o))$ . By the hypothesis  $t \notin N_S(o)$  and the definition of  $abind_S(o)$ ,  $t \in abind_S(o)$ .  $\square$

**Example 5:** Let us consider again the problem  $\mathcal{A}_S(1)$  of the last Example. From Table II and the last Proposition, it follows that  $abind_S(1) = \{c, g\}$  and that  $I^+ = I \cup \{(c, 1), (g, 1)\}$ , *i.e.* the abduced interpretation extends the index of object 1 with two terms, **c** and **g**. Clearly, nothing guarantees that object 1 is indeed truly an instance of these two terms; this is just a “guess”, as, for instance, when making a diagnosis. However, since the selection of these terms is based on abduction, we are guaranteed that the original index of object 1 is “preserved” in the EIS (*i.e.* the diagnosis implies the symptoms); moreover, the minimization of perturbation is compliant with Occam’s razor, guaranteeing that no simpler extension (diagnosis) exists. More specifically, term **c** is selected because object 1 is an instance of **y**, and this could be caused by its being an instance of **c**. In addition, the inclusion of **c** in the *indx* of 1 only causes **c** itself to be added to 1 instance set. In this respect, term **e**, another sibling of **x**, is not selected because **e** has more substantial implications than **c**, namely **g** and **f**; at this stage, these implications make **e** a less simple choice than **c**. On the other hand, term **g** is selected because in so doing a minimal alteration to the IS is made; as a guess, this latter is somewhat more hazardous than the former, but as grounded from a logical point of view. From a purely diagnostic point of view, a guess of this kind does not make any sense, since term **g** does not give any contribution to the explanation of  $ind_S(1)$ . However, in the present setting guesses of this kind may represent genuine discoveries, whose excusion would confine the effectiveness of the method to the obvious. Furthermore, as it will be noted in Section 5.1, this behavior guarantees a fundamental property of the method, akin to completeness: by iterating the extension all terms will eventually be selected (for each object). More on this in Section 5.1.  $\square$

Proposition 4 suggests the following algorithm for computing  $\beta(t)$ : for each object  $o \in Obj$ , select the terms  $u$  in  $T \setminus N_S(o)$  that minimize the size of  $\sigma(u) \setminus N_S(o)$ . If  $t$  is amongst these terms,  $o \in \beta(t)$ , otherwise  $o \notin \beta(t)$ . Procedure  $\beta 1$ , shown in Figure 3, implements this algorithm.  $\beta 1$  uses  $beta$ ,  $N_S$ ,  $d$ ,  $ABIND$  and  $\pi$  to compute, respectively,  $\beta(t)$ ,  $N_S(o)$ , the minimum size of  $\pi(u)$ ,  $abind_S(o)$  and  $\pi(u)$ .  $beta$  is initialized to the empty set, then the main loop is entered. In it, for each object  $o \in Obj$ ,  $\beta 1$  first computes  $N_S(o)$  by the *succ* function. This function visits the successors in  $G_S$  of each element in the input set, and is not reported as it implements an algorithm in elementary graph theory which can be found on any textbook. Then,  $d$  is initialized to  $|T|+1$ , which is greater than any value  $d$  may assume, and so is equivalent to infinite, while  $ABIND$  is initialized to the empty set. For each term  $u \in T \setminus N_S(o)$ ,  $\beta 1$  computes in  $\pi$  the set  $\sigma(u) \setminus N_S(o)$  and collects in  $ABIND$  the terms which minimize the size of  $\pi$ . Finally, only if  $t$  is amongst these terms,  $\beta 1$  adds  $o$  to  $beta$ . Soundness and completeness of  $\beta 1$  directly follow from Proposition 4, and allows us to obtain an upper bound on the complexity of computing  $\beta(t)$ .

**Proposition 5:** For all ISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $\alpha_{S^e}(t)$  can be computed in  $O(|T| \cdot |Obj|^2 \cdot \lg |Obj|)$  time.

*Proof:*  $\beta 1$  loops on  $Obj$ , and, for each object  $o$ , it first loops on  $T \setminus N_S(o)$  (lines 8-17) and then updates the variable  $beta$  (line 18). Let us consider the loop on  $T \setminus N_S(o)$ . In the worst case,  $N_S(o)$  is empty, so that this loop is executed  $|T|$  times. At each iteration,  $\beta 1$  invokes the *succ* function, which visits the term graph in order to find the terms reachable from the current term. The worst case is when the term graph is fully connected, so that, at each iteration, the number of edges to be visited equals  $|T|^2$ , and the

```

procedure  $\beta_1$  ( $S : \mathbf{IS}$  ;  $t : \mathbf{term}$  ;  $beta : \mathbf{set\ of\ objects}$ )
1. begin
2.  $beta \leftarrow \emptyset$ 
3. for each object  $o \in Obj$  in  $S$  do
4.   begin
5.      $N_S \leftarrow succ(ind_S(o))$ 
6.      $d \leftarrow |T| + 1$ 
7.      $ABIND \leftarrow \emptyset$ 
8.     for each term  $u \in T \setminus N_S$  do
9.       begin
10.         $\pi \leftarrow succ(\{u\}) \setminus N_S$ 
11.        if  $|\pi| < d$  then
12.          begin
13.             $d \leftarrow |\pi|$ 
14.             $ABIND \leftarrow \pi$ 
15.          end
16.        else if  $|\pi| = d$  then  $ABIND \leftarrow ABIND \cup \pi$ 
17.        end
18.        if  $t \in ABIND$  then  $beta \leftarrow beta \cup \{o\}$ 
19.        end
20.   end

```

Figure 3. The  $\beta_1$  procedure

variable  $\pi$  is assigned the whole terminology. In this case,  $d$  is always equal to  $|T|$  and the test at line 16 is always passed, except at the first iteration, when the test at line 11 is passed. The time complexity of each iteration of the loop, different from the first, is then given by  $|T|^2 + |Obj| \lg |Obj|$ , and that of the entire loop is  $O(|T|^3 + |T| \cdot |Obj| \lg |Obj|)$ . The test on line 18 is always passed in the worst case, and every object  $o$  is inserted into  $beta$ , which grows by one element at each iteration, until it equals  $Obj$ . By keeping  $\beta$  ordered, this insertion has time complexity  $\lg k$ , where  $k$  is the current size of  $beta$ . It can be proved that the dominant term in

$$\sum_{j=1}^{|Obj|} \lg j$$

is  $|Obj| \lg |Obj|$ , thus the overall complexity of  $\beta_1$  is  $O(|Obj| \cdot |T|^3 + |T| \cdot |Obj|^2 \cdot \lg |Obj|)$ . Since, as already remarked, the size of the terminology is expected to be significantly smaller than that of the domain, the upper bound on  $\beta(t)$  is  $O(|T| \cdot |Obj|^2 \cdot \lg |Obj|)$ . By combining Proposition 3, stating that  $\alpha_{S^e}(t) = \alpha_S(t) \cup \beta(t)$ , and Proposition 2 giving the complexity of computing  $\alpha_S(t)$ , it follows that  $O(|T| \cdot |Obj|^2 \cdot \lg |Obj|)$  is also an upper bound on  $\alpha_{S^e}(t)$ .  $\square$

The last result establishes that query evaluation on EISs has a higher complexity than query evaluation on ISs, by a factor equal to the size of the domain. Since the latter is based on deduction while the former is based on abduction, this result is consistent with the complexity results for propositional abduction problems presented in [11].

Next Subsection is devoted to devise a method for computing  $\beta(t)$  that is in practice more efficient than the rather naive one sketched in the proof of the last Proposition. The framework developed to

this end will be useful in establishing complexity upper bounds for the special kind of ISs considered in Section 4.

### 3.3. Minimal Terminal Sets

We first observe that we do not need to compute  $abind_S(o)$ , *i.e.* solve an abduction problem, for each object  $o \in Obj$ . Objects having the same index give rise to the same propositional abduction problem and therefore have the same abduced index. Then, we only need to solve an abduction problem for each different index. Let  $\approx$  be a binary relation on objects, defined as follows:

$$o \approx o' \text{ if and only if } ind_S(o) = ind_S(o').$$

$\approx$  is an equivalence relation. Let  $[o]$  be the equivalence class of object  $o$ . We call each  $[o]$  a *class*. The above consideration is captured in the following Proposition:

**Proposition 6:** For all ISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $\beta(t) = \cup \{ [o] \mid t \in abind_S(o) \}$ .

*Proof:* For each object  $o' \in [o]$ ,  $o' \in \beta(t)$  iff  $t \in abind_S(o')$ . But  $ind_S(o) = ind_S(o')$  implies  $abind_S(o) = abind_S(o')$ . Hence  $t \in abind_S(o')$  iff  $t \in abind_S(o)$ .  $\square$

In the worst case, the number of classes equals the number of objects, but in the average case, the former can be much smaller than the latter. In the typical database application, the size of the schema is a few orders of magnitude smaller than that of the data. For the Web, these numbers are harder to obtain, due to the highly dynamic nature of the information system. According to [37], the size of the taxonomies employed in this context is of the order of few hundreds of thousands, and the indexed objects (*i.e.* Web pages) range from millions up. In general, considering classes should result in a significant optimization, at least in some types of applications.

Another major source of inefficiency of algorithm  $\beta 1$  is that it loops on the set  $T \setminus N_S(o)$ . This set can be very large, and it may be very time consuming to compute the perturbation of each STS for each term in it. Fortunately, as it will be shown soon,  $\beta(t)$  can be computed by exploring only a part of the set  $T \setminus N_S(o)$ , by visiting (a subgraph of) the term graph in the appropriate way.

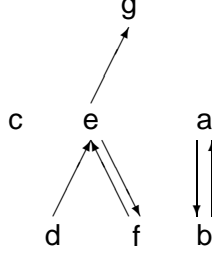
**Definition 10:** Given a graph  $G = (V, E)$ , a *terminal set in G* is either a singleton  $\{v\}$ , where  $v$  is a vertex with no outgoing edges, or is a set of vertices  $\{v_1, v_2, \dots, v_k\}$ ,  $k \geq 2$ , such that  $E(v_i) = \{v_{i+k}1\}$  for all  $1 \leq i \leq k$ , where  $+_k$  is sum modulo  $k$ . A terminal set  $A$  in  $G$  is *minimal* if no other terminal set  $A'$  in  $G$  exists such that  $|A'| < |A|$ . Any element of a minimal terminal set is called a *minterm*.  $\square$

We can now state the basic result on which the method for answering queries on EISs relies. Given a graph  $G = (V, E)$  and a set  $V' \subseteq V$ , the subgraph of  $G$  induced by  $V'$  is the graph  $G' = (V', E')$ , where  $E' = \{(u, v) \in E \mid u, v \in V'\}$ .

**Proposition 7:** Given an IS  $S = \langle T, K, Obj, I \rangle$ , an object  $o$  and a term  $t \in T$ ,  $t \in abind_S(o)$  iff  $t$  is a minterm in the subgraph  $G_S^o$  of the term graph  $G_S$  induced by  $T \setminus N_S(o)$ .

*Proof:* ( $\rightarrow$ )  $t \in abind_S(o)$  implies that the perturbation of  $\mu_o(t)$  is non-empty and has smallest size. Suppose  $t$  is not a minterm in  $G_S^o$ . Then there exists a term  $v \in T \setminus N_S(o)$  such that  $(t, u) \in E$  and  $(u, t) \notin E$ . But then  $pert_o(\mu_o(v)) < pert_o(\mu_o(t))$ , a contradiction.

( $\leftarrow$ ) Let  $t$  be a minterm. There are two cases: (1)  $\{t\}$  is a minimal terminal set. In this case,  $pert_o(\mu_o(t)) = 1$  and there cannot be a smaller positive perturbation in  $Sol(\mathcal{A}_S(o))$ ; so  $\mu_o(t) \in Sol_{\geq}(\mathcal{A}_S(o))$  and  $t \in abind_S(o)$ . (2)  $A = \{v_1, v_2, \dots, v_k\}$  is a minimal terminal set and  $t = v_j$  for some  $1 \leq j \leq k$ .

Figure 4. The graph  $G_S^1$ 

It follows that  $\text{pert}_o(\mu_o(v_1)) = \dots = \text{pert}_o(\mu_o(v_k)) = k$ . By the minimality of  $A$  no smaller terminal set exists. Then, for all  $1 \leq i \leq k$ ,  $\mu_o(v_i) \in \text{Sol}_{\leq}(\mathcal{A}_S(o))$ , hence  $v_i \in \text{abind}_S(o)$ , and in particular  $t \in \text{abind}_S(o)$ .  $\square$

**Example 6:** Figure 4 shows  $G_S^1$ , that is the subgraph of the term graph in Figure 2 induced by  $N_S(1)$ .  $G_S^1$  has 3 terminal sets:  $\{g\}$ ,  $\{c\}$  and  $\{a, b\}$ , the first two of which are minimal. Therefore  $\text{abind}_S(1) = \{c, g\}$ .  $\square$

The last Proposition reduces the computation of  $\beta(t)$  to that of the minterms in the graph  $G_S^o$ . In general,  $G_S^o$  will have two kinds of terminal sets:

- the terminal sets in  $G_S$  not involving terms in  $N_S(o)$ ; we call these *genuine* terminal sets;
- the terminal sets created by the pruning of  $G_S$  giving rise to  $G_S^o$ ; we call these *induced* terminal sets.

We assume that the terminal sets of the term graph are computed at system start-up, then stored, and revised every time the term graph is modified. Since they play a crucial role in answering queries on EIS, we think this is a safe assumption, from a database point of view. Then, genuine terminal sets are found by identifying the stored terminal sets which are free from elements in  $N_S(o)$ . Induced terminal sets can be found by exploring the borders of  $N_S(o)$  in  $G_S$  as done by procedure  $\beta 2$ , presented in Figure 5.

$\beta 2$  takes as input an IS  $S$  and a term  $t$ , and computes, in *beta*,  $\beta(t)$ . It operates under the following assumptions, for a given IS  $S$ ,

1. the classes of  $S$  are known;
2. the components  $G_1, \dots, G_{n_S}$  of the term graph  $G_S$  are known;
3.  $G_S$  is represented through adjacency sets, *i.e.* for each term  $t \in T$ ,  $\text{Adj}[t]$  is a set containing all terms  $u$  such that there is an edge from  $t$  to  $u$ , *i.e.*  $(t, u) \in E$ ;
4. inverse adjacency sets are also available, *i.e.* for each term  $t \in T$ ,  $\text{Adj}^{-1}(t)$  gives the nodes  $u \in T$  such that  $(u, t) \in E$ .

$\beta 2$  uses the same variables as  $\beta 1$ . It loops on the classes of the IS  $S$ , and for each such class  $[o]$ , it computes in  $N_S$  the set  $N_S(o)$  via the *succ* function. Then, a loop is entered on each component  $G_i$  of the term graph  $G_S$ ; within this loop, genuine terminal sets are first identified, by checking each stored

```

procedure  $\beta_2$  ( $S : \mathbf{IS} ; t : \mathbf{term} ; \mathit{beta} : \mathbf{set\ of\ objects}$ )
1. begin
2.  $\mathit{beta} \leftarrow \emptyset$ 
3. for each class  $[o]$  in  $S$  do begin
4.    $N_S \leftarrow \mathit{succ}(\mathit{ind}_S(o))$ 
5.    $d \leftarrow |T| + 1$ 
6.    $ABIND \leftarrow \emptyset$ 
7.   for each component  $G_i = (T_i, E_i)$  of  $G_S$  do begin
8.     for each terminal set  $TS$  of  $G_i$  do
9.       if  $TS \cap N_S = \emptyset$  then
10.        if  $|TS| < d$  then begin
11.           $d \leftarrow |TS|$ 
12.           $ABIND \leftarrow TS$ 
13.        end
14.        else if  $|TS| = d$  then  $ABIND \leftarrow ABIND \cup TS$ 
15.       $B \leftarrow \emptyset$ 
16.      for each term  $t \in N_S$  do
17.        for each term  $u \in \mathit{Adj}^{-1}[t] \setminus N_S$  do
18.          if  $u \notin B$  then begin
19.             $B \leftarrow B \cup \{u\}$ 
20.            if  $\mathit{Adj}[u] \subseteq N_S$  then
21.              if  $d > 1$  then begin
22.                 $d \leftarrow 1$ 
23.                 $ABIND \leftarrow \{u\}$ 
24.              end
25.              else  $ABIND \leftarrow ABIND \cup \{u\}$ 
26.            else  $\mathit{explore}(u)$ 
27.          end
28.        end
29.      if  $t \in ABIND$  then  $\mathit{beta} \leftarrow \mathit{beta} \cup [o]$ 
30.    end
31.  end

procedure  $\mathit{explore}$  ( $\mathit{path} : \mathbf{sequence\ of\ terms}$ )
1. begin
2.  $t \leftarrow \mathit{last}(\mathit{path})$ 
3.  $A \leftarrow \mathit{Adj}[t] \setminus N_S$ 
4.  $u \leftarrow \mathit{get}(A)$ 
5. if  $|A| = 1$  then
6.   if  $u \in \mathit{path}$  then
7.     if  $|\mathit{path}| < d$  then
8.       begin
9.          $d \leftarrow |\mathit{path}|$ 
10.         $ABIND \leftarrow \mathit{path}$ 
11.       end
12.     else if  $|\mathit{path}| = d$  then  $ABIND \leftarrow ABIND \cup \mathit{path}$ 
13.   else if  $u \notin B$  and  $\mathit{Adj}[u] \not\subseteq N_S$  then
14.     begin
15.        $B \leftarrow B \cup \{u\}$ 
16.        $\mathit{explore}(\mathit{append}(\mathit{path}, u))$ 
17.     end
18.  end

```

Figure 5. The  $\beta_2$  and  $\mathit{explore}$  procedures

terminal set of the component for intersection with  $N_S$  (line 9). If the terminal set  $TS$  does not contain any term in  $N_S$ , then it is a potential minimal terminal set, thus, its cardinality is compared to the current value of  $d$ . If the former is smaller than the latter, then  $TS$  is smaller than any terminal set found so far, therefore  $ABIND$  is set to it, and  $d$  is set to its cardinality (lines 11 and 12). If the cardinality of  $TS$  is the same as the current value of  $d$ , then  $TS$  is another minimal solution according to the current  $d$ , and as such it is added to  $ABIND$  (line 14). If neither is the case,  $TS$  is just ignored as not minimal.  $\beta_2$  then proceeds to compute induced terminal sets. To this end, it needs to visit the term graph. The set  $B$  is used to remember the terms already visited; initially,  $B$  is empty (line 15). A loop is then entered, in which, for each term  $t$  in  $N_S$ , each predecessor  $u$  not in  $N_S$  is considered (line 17). If  $u$  is not yet visited (line 18), it is inserted in  $B$  and considered in the present iteration. There can be two cases:

1. All direct successors of  $u$ ,  $Adj[u]$ , are in  $N_S$  (line 20). In this case,  $\{u\}$  is a terminal set in  $G_S^o$ , which implies that the smallest size of non-empty perturbations  $d_o$  is 1. Hence, if the current value of  $d$  is greater than 1, all solutions found so far are not minimal:  $d$  is set to 1 and  $ABIND$  is initialized with  $\{u\}$  (lines 22 and 23). If, on the other hand, the current value of  $d$  is 1, then  $u$  is simply added to  $ABIND$  (line 25).
2. There is some direct successors of  $u$  not in  $N_S$ . In this case,  $u$  may still be a minterm, in case it belongs to a terminal set with more than one element. In order to check whether this is the case, the procedure *explore* is invoked, with the sequence consisting of just  $u$  as input (line 26).

When all the minterms of the current class have been identified,  $\beta_2$  checks whether  $t$  is amongst these. If yes (line 29), then  $[o]$  is added to *beta*.

The procedure *explore*, also presented in Figure 5, takes as input a sequence *path* of terms outside  $N_S$  that is a path in  $G_S^o$ ; each element of this path, except the last one, has been found to have the next element in *path* as its only successor outside  $N_S$ , as required by Definition 10. The last element in *path* is known to have at least one successor outside  $N_S$ . *path* is thus a potential terminal set, and an effective one just in case *path* will at some point become a cycle, consisting of elements having exactly one successor outside  $N_S$ . The procedure *explore* explores the graph  $G_S$  in order to make sure it detects such a terminal set, if one exists. To this end, it focuses on the last element of *path*,  $t$ , which is also the first one, upon the first invocation. The variable  $A$  is assigned the successors of  $t$  outside  $N_S$  (line 3), and  $u$  is anyone of these (line 4). If  $t$  has more than one successor outside  $N_S$ , *i.e.*  $|A| > 1$ , it violates the condition of Definition 10 requiring that each element in a terminal set have exactly one successor, so *explore* is exited with no further action. If, on the other hand,  $|A| = 1$ , then there can be two cases:

1.  $u$  is an element in *path*. If yes, then a cycle has been detected, and an induced terminal set. Consequently, if the size of the cycle is smaller than the current  $d$  (line 7), all current solutions are not minimal, so both  $d$  and  $ABIND$  are re-initialized, respectively with the size of *path* and with its elements (lines 9 and 10). If, instead,  $d$  equals the size of *path*, then *path* is just added to  $ABIND$  (line 11).
2.  $u$  is not already in *path*. In this case,  $u$  may have been already visited, *i.e.*  $u \in B$ , which means that *path* cannot be a cycle, otherwise all its element would be in  $B$ . Or, being newly discovered,  $u$  may have no successor at all, which means that it is a genuine terminal set, or may have all successors in  $N_S$ , which means that it is an induced terminal set. If it is a genuine terminal set, then it has already been identified in the first stage of  $\beta_2$  and need not be considered; if it is an induced

terminal set, then, being not already visited, will be identified as soon as one of its successors in  $N_S$  is considered. Therefore, in all these cases the current *path* is no solution, and no action needs to be done. The only interesting case is if  $u$  is not visited and has some successor outside  $N_S$ . In this case, *explore* adds  $u$  to the visited terms and recursively invokes itself on the current path extended with  $u$  (line 16).

Soundness and completeness of the  $\beta_2$  procedure directly follow from the above considerations and from Proposition 7.

### 3.4. Query-less Information Access

Having established an efficient procedure for extending an IS, we can now show how this procedure can be applied in order to implement the query-less information access mechanism alluded to in Section 1.1. To this end, a superscript will be used in order to indicate the iteration at which an IS  $S$  is generated, that is,  $S = S^0$ ,  $S^e = S^1$ ,  $(S^e)^e = S^2$  and so on.

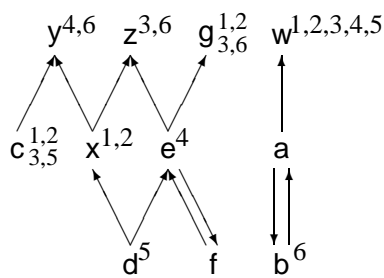
The interface of the access mechanism consists of:

- the query  $q$  whose answer leaves to the user much to desire; this query will not change during the access process and is displayed only for the user convenience;
- a set  $O$  of objects; and
- an *extend* button, which the user may push at his will in order to execute another step.

At the beginning,  $O = O^0 = ans(q, S^0)$ . At step  $k \geq 0$ , pushing the *extend* button makes the interface change as follows:

$$O^{k+1} = ans(q, S^{k+1}). \tag{1}$$

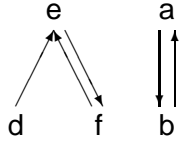
In words, by pushing the *extend* button, the user implicitly extends the current IS  $S^k$  to  $S^{k+1}$ , and observes in  $O$  the effect of this extension on the answer to the query  $q$ . An illustration of this mechanism on the running example follows.



$o$	$abind_{S^0}(o)$	$ind_{S^0}(o)$	$ind_{S^1}(o)$
1	{c, g}	{w, x}	{c, g, w, x}
2	{c, g, w}	{x}	{c, g, w, x}
3	{g, w, z}	{c}	{c, g, w, z}
4	{w, y}	{e}	{e, w, y}
5	{c, w}	{d}	{c, d, w}
6	{g, y, z}	{b}	{b, g, y, z}

Figure 6. Extending the information source  $S$

**Example 7:** Let us consider again source  $S$  given in Figure 2. Figure 6, right, shows a table reporting the abduced index of each object in the IS and the resulting index in the EIS  $S^1$ ; for



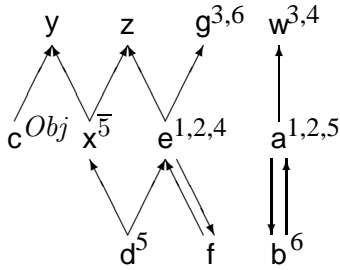
convenience, also the index of each objects in  $S$  is shown. The same figure, left, shows  $S^1$ . Let us now extend  $S^1$ , starting from object 1. The problem we must consider is  $\mathcal{A}_{S^1}(1)$ . The graph  $G_{S^1}^1$  is shown left. This graph has two terminal sets  $\{a, b\}$  and  $\{e, f\}$ , both minimal. It follows that  $abind_{S^1}(1) = \{a, b, e, f\}$ . Assuming the terms are ordered lexicographically,  $\rho(abind_{S^1}(1)) = \{a, e\}$ , so the abduced interpretation of  $S^1$  is given by:

$$\{(c, 1), (x, 1), (g, 1), (w, 1)\} \sqcup \{(a, 1), (e, 1)\}$$

In computing this expression, the pair  $(a, 1)$  “rules out” the pair  $(w, 1)$ , while  $(e, 1)$  rules out  $(g, 1)$ . The abduced index is therefore:

$$\{(c, 1), (x, 1), (e, 1), (a, 1)\}.$$

Figure 7 shows the IS  $S^2$ . For readability, the interpretation of term  $x$ , which is  $\{1, 2, 3, 4, 6\}$ , is denoted  $\bar{5}$ . The effect of two consecutive extensions on the answer to three different queries is displayed in Table III. It can be observed that the first two queries result in successive enlargements of the set  $O$ , while the third query breaks monotonicity since it contains negation.  $\square$



$o$	$abind_{S^1}(o)$	$ind_{S^1}(o)$	$ind_{S^2}(o)$
1,2	$\{a, b, e, f\}$	$\{c, g, w, x\}$	$\{a, c, e, x\}$
3	$\{x\}$	$\{c, g, w, z\}$	$\{c, g, w, x\}$
4	$\{c, f, x\}$	$\{e, w, y\}$	$\{c, e, w, x\}$
5	$\{a, b\}$	$\{c, d, w\}$	$\{a, c, d\}$
6	$\{c, x\}$	$\{b, g, y, z\}$	$\{b, c, g, x\}$

Figure 7. Extending the information source  $S^1$

### 4. Hierarchical Information Sources

In this Section, we apply the ideas developed so far to a special class of ISs, namely those whose term graphs are directed acyclic graphs (DAGs) with a maximal element  $\top$ . We call these ISs “hierarchical”. Hierarchical taxonomies are common in object-oriented models, where subsumption is a partial ordering

$q$	$O^0 = ans(q, S)$	$O^1 = ans(q, S^1)$	$O^2 = ans(q, S^2)$
$c$	$\{3\}$	$\{1,2,3,5\}$	$\{1,2,3,4,5,6\}$
$c \wedge f$	$\{\}$	$\{5\}$	$\{1,2,4,5\}$
$c \wedge \neg f$	$\{3\}$	$\{1,2,3\}$	$\{3,6\}$

Table III. Successive extensions of the example IS

relation amongst classes and maximal elements are introduced in order to tie classes up, thus making each class reachable from the top.

The following Proposition defines hierarchical ISs and addresses query evaluation on their extensions.

**Proposition 8:** Let a *hierarchical* information source (HIS) be an IS whose term graph is a DAG with a greatest element  $\top$ . Then, for all HISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $t \neq \top$ ,

$$ans(t, S^e) = \bigcap \{ \alpha_S(u) \mid t \rightarrow u \in K \}.$$

*Proof:* From Proposition 7, a term  $t \in abind_S(o)$  iff  $t$  is a minterm in the subgraph  $G_S^o$ . Since the term graph  $G_S^o$  of a HIS is a DAG,  $G_S^o$  is a DAG too, hence its terminal sets, according to Definition 10, are those singletons  $\{u\}$  such that  $u$  is a term with no outgoing edges. Each such terminal set is obviously minimal, being a singleton. Thus, an object  $o \in \beta(t)$  iff  $t$  has no outgoing edges in  $G_S^o$ . There may be two cases. (1)  $N_S(o) = \emptyset$ . In this case  $G_S = G_S^o$ , and  $\top$  is the only minterm. Thus  $t = \top$  and the Proposition holds vacuously. (2)  $N_S(o) \neq \emptyset$ . In this case, since  $\top$  is the greatest element of the taxonomy,  $\top \in N_S(o)$ , so  $t \neq \top$ . Then,  $t$  is a minterm iff all immediate successors of  $t$  in  $G_S$  are in  $N_S(o)$ , but  $t$  is not. In this way, when  $N_S(o)$  is “erased” from  $G_S$  to produce  $G_S^o$ ,  $t$  is left with no outgoing edges. It follows that:

$$\begin{aligned} \beta(t) &= \{o \in Obj \mid t \notin N_S(o), \text{ and } t \rightarrow u \in K \text{ implies } u \in N_S(o)\} \\ &= \{o \in Obj \mid C_S(o) \not\models t, \text{ and } t \rightarrow u \in K \text{ implies } C_S(o) \models u\} \\ &= ans(\bigwedge \{u \mid t \rightarrow u \in K\} \wedge \neg t, S) \\ &= \bigcap \{ \alpha_S(u) \mid t \rightarrow u \in K \} \setminus \alpha_S(t) \end{aligned}$$

From Proposition 3,  $ans(t, S^e) = \alpha_S(t) \cup \beta(t)$ , therefore  $\alpha_S(t) \cup (\bigcap \{ \alpha_S(u) \mid t \rightarrow u \in K \} \setminus \alpha_S(t))$  and the Proposition follows.  $\square$

The last Proposition establishes that an object  $o$  is in the result of query  $t$  against the EIS  $S^e$  just in case it is an instance of all the immediate generalizations of  $t$  in  $S$ . Indeed, if  $o$  were an instance of  $t$ , it would, as a consequence, be an instance of all  $t$ 's generalizations, thus the explanation of the current index offered by the system is the most reasonable one can conceive. As a result, the behavior of the query mechanism turns out to be compliant with intuition. Notice that asking the query  $\top$  on the extended IS, a case not dealt with by the last Proposition, does not make much sense, since already  $ans(\top, S) = Obj$ .

**Example 8:** Let us consider the HIS  $H$  shown in Figure 8, whose taxonomy has been already used in Section 1.1. In that Section, it has been pointed out that a user looking for small cameras via the query **Miniatures** could obtain a disappointing answer due to the fact the IS reserves the latter term for much smaller cameras than those meant by the user. Let 1 be such a camera. From the user point of view the index lacks the knowledge that 1 is an instance of **Miniatures**. Note that this knowledge explains the classification of 1 as a **StillCams**. By pushing the *extend* button, the problems  $\mathcal{A}_H(1)$ ,  $\mathcal{A}_H(2)$ , and  $\mathcal{A}_H(3)$  are considered.

- $\mathcal{A}_H(1)$  has four minimal solutions, given by  $\mu_1(\text{Reflex})$ ,  $\mu_1(\text{Miniatures})$ ,  $\mu_1(\text{MovingPictureCams})$ , and  $\mu_1(\text{UnderwaterDev})$ . Indeed, **Reflex**, **Miniatures**, **MovingPictureCams** and **UnderwaterDev** are the minterms of the graph  $G_H^1$ .

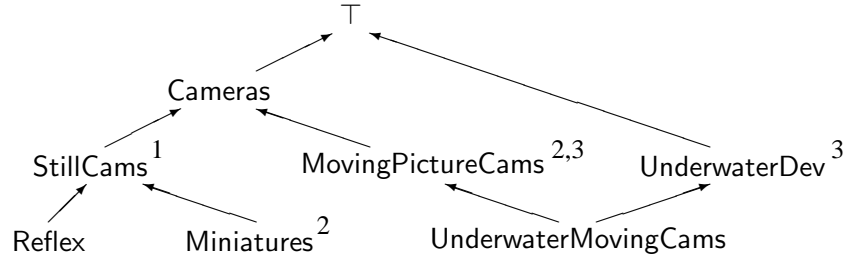


Figure 8. A hierarchical information source

- The minterms of the graph  $G_H^2$  are **Reflex** and **UnderwaterDev**, thus problem  $\mathcal{A}_H(2)$  has two minimal solutions, given by  $\mu_2(\text{Reflex})$  and  $\mu_2(\text{UnderwaterDev})$ .
- Finally, the problem  $\mathcal{A}_H(3)$  has two minimal solutions, given by  $\mu_3(\text{StillCams})$  and  $\mu_3(\text{UnderwaterMovingCams})$ .

The EIS  $H^1$  is shown in Figure 9. We have

$$\begin{aligned}
 O^0 &= \text{ans}(\text{Miniatures}, H) = \{2\} \\
 O^1 &= \text{ans}(\text{Miniatures}, H^1) = \cap \{ \alpha_H(u) \mid \text{Miniatures} \rightarrow u \in K \} = \alpha_H(\text{StillCams}) \\
 &= I(\text{StillCams}) \cup I(\text{Miniatures}) \cup I(\text{Reflex}) = \{1, 2\}
 \end{aligned}$$

Thus, object 1 is retrieved after one extension.  $\square$

From a complexity point of view, the last Proposition permits to compute an upper bound on the evaluation of queries on an extended HIS.

**Proposition 9:** For all HISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $\alpha_{S^e}(t)$  can be computed in  $O(|T|^2 \cdot |Obj| \cdot \lg |Obj|)$  time.

*Proof:* The query evaluation procedure derived from the last Proposition consists of 3 steps: (a) to compute each immediate successors  $u$  of  $t$  in the term graph; (b) to evaluate the query  $u$  on the HIS; (c) to take the intersection of the results. In order to derive the worst case, let  $x$  be the number of terms that are immediate successors of  $t$ , and  $y$  the terms that are reachable backward from each immediate successor of  $t$ . In the worst case,  $y = |T| - x$ . Using the result established in Proposition 2 for the

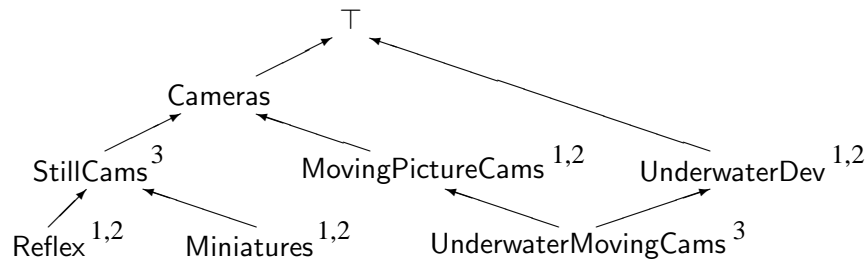


Figure 9. A hierarchical information source

IS type	Complexity
Simple	$O( T  \cdot  Obj  \cdot \lg  Obj )$
Extended	$O( T  \cdot  Obj ^2 \cdot \lg  Obj )$
Extended Hierarchical	$O( T ^2 \cdot  Obj  \cdot \lg  Obj )$
Extended Tree	$O( T  \cdot  Obj  \cdot \lg  Obj )$

Table IV. Summary of complexity results for query evaluation

complexity of step (b), the cost function has a maximum for  $x = \frac{|T|-1}{2}$ . Using this value, step (a) requires  $\frac{|T|-1}{2}$  time; step (b)  $O(|T|^2 \cdot |Obj| \cdot \lg |Obj|)$ ; step (c)  $O(|T| \cdot |Obj| \cdot \lg |Obj|)$ . The cost of step (b) is clearly the largest of the three, so the Proposition follows.  $\square$

Last Proposition shows that the evaluation of queries on extended HISs is a much simpler problem than the general one, studied in the previous part of the paper.

HISs include the ISs whose term graphs are trees. This kind of taxonomies, which we call *tree* information sources (TISs), are common in catalogs, directories and Web search engines, and therefore deserve some attention. As a corollary of Proposition 8, and of the observation that a TIS is a HIS in which every term different from  $\top$  has exactly one immediate successor in the term graph, we have that, for all TISs  $S = \langle T, K, Obj, I \rangle$  and terms  $t \in T$ ,  $t \neq \top$ ,

$$ans(t, S^e) = \{\alpha_S(u) \mid t \rightarrow u \in K\}.$$

The complexity of query evaluation on extended TISs is clearly the same as that on ISs.

Table IV summarizes the complexity results obtained for query evaluation on the classes of IS examined in this study. The last result indicates that the evaluation of queries on extended HISs is worse than that on ISs by a factor proportional to the size of the terminology. This is a significant improvement over the general case, where the factor is proportional to the size of the domain (Proposition 5). This difference reflects the fact that in the general case,  $\alpha_{S^e}(t)$  must be computed in an object-based (or class-based) fashion, *i.e.* by considering one object (class) at a time, while the evaluation of  $\alpha_{S^e}(t)$  on a HIS proceeds in a term-based fashion, *i.e.* by considering the terms that are immediate successors of  $t$  in the term graph. This also simplifies the implementation, as it avoids to compute and keep track of classes.

## 5. Applications

The basic application of the information source extension introduced in the previous part of the paper is the query-less information access mechanism presented in Section 3.4. As pointed out in Section 1, this is not the only possible exploitation. In this Section, we consider two important, complementary, applications. The first application is semantic-based information *browsing*, by which the user initially selects one object, and then navigates through a series of sets of objects, related by a semantic similarity criterion. The second application is query result *ranking*, in which all the objects in the IS that would be returned at some point by the query-less access mechanism are shown to the user at once, ordered in decreasing degree of closeness to the query. The rest of this Section is devoted to illustrating in detail each one of these two mechanisms, starting from browsing.

### 5.1. Browsing by Semantic Similarity

IS browsing is realized via an iterative application of the IS extension operator, which is used to capture a notion of semantic similarity. The interface of the browsing mechanism is given by:

- a set  $O$  of objects; and
- a *browse* button, which the user may push at his will in order to obtain from the system the objects that are most similar to those in  $O$ . The similarity in question is clearly semantic in nature, since all the system knows about objects is their interpretation.

Initially,  $O$  is set by the user to  $O^0 = \{o\}$  where  $o$  is a prototype of the sought objects. In order to start browsing, the user then pushes the *browse* button. In response, the interface changes as follows:

$$O^1 = \text{ans}(q^1, S)$$

where

$$q^1 = \bigvee \{t \mid t \in \text{ind}_S(o)\} \wedge \bigwedge \{\neg u \mid u \in T \setminus N_S(o)\}$$

$O$  now comprises the objects whose index shares at least one term with the index of  $o$ , and are therefore semantically similar to it. Clearly, a different similarity criterion could be selected, resulting in a different query. The one we have chosen is rather loose; however, as it will be proved, it guarantees that the browsing process does indeed allow the user to explore the whole IS, and as such satisfies a basic intuitive requirement of browsing. If the user is unsatisfied by the current set of objects, he pushes the *browse* button again and, in response:

$$O^2 = \text{ans}(q^2, S)$$

where

$$q^2 = \bigvee \{t \mid t \in \text{abind}_S(o)\} \wedge \bigwedge \{\neg u \mid u \in T \setminus N_{S^e}(o)\}$$

The objects obtained in this way share at least one term with the abduced index of  $o$ , and are therefore the most similar to it, except for those in  $O^1$ , which have been already browsed at the previous iteration. If these new objects satisfy the user, the process is over, otherwise another step takes place, resulting in the replacement of  $O^2$  by the set of objects  $O^3 = \text{ans}(q^3, S)$  where

$$q^3 = \bigvee \{t \mid t \in \text{abind}_{S^1}(o)\} \wedge \bigwedge \{\neg u \mid u \in T \setminus N_{S^2}(o)\}$$

and so on, until either the user is satisfied or no more expansion is possible. The semantics of the browsing mechanism is described by the following equation (for  $k \geq 0$ ):

$$O^{k+1} = \text{ans}(q^{k+1}, S). \quad (2)$$

where  $q^1$  is as above, while for  $m \geq 2$ :

$$q^m = \bigvee \{t \mid t \in \text{abind}_{S^{m-2}}(o)\} \wedge \bigwedge \{\neg u \mid u \in T \setminus N_{S^{m-1}}(o)\}$$

The complementarity between browsing and query-less access can be appreciated by contrasting equations (1) and (2). By using the browsing mechanism, the user can progressively retrieve all the objects

of the IS, starting from an object that is an ostensive representation of his information needs, and then moving away, step after step, in a centrifugal process, where the center is represented by the initial object. A generalization of this interaction mechanism is studied in [29].

In order to ground this mechanism on mathematical basis, the properties of the  $\cdot^+$  function, on which the abduced index is based, are now investigated. Intuitively, we would expect that the application of  $\cdot^+$  to an interpretation, produce an equal or a larger interpretation, the former case corresponding to the situation in which the taxonomy of the IS does not enable to find any explanations for each object index. Technically, this amounts to say that  $\cdot^+$  is a monotonic function, which is in fact the case. Then, by iterating the  $\cdot^+$  operator, we expect to move from an interpretation to a larger one, until an interpretation is reached which cannot be extended any more. Also this turns out to be true, and in order to show it, we will model the domain of the  $\cdot^+$  operator as a complete partial order, and use the notion of fixed point in order to capture interpretations that are no longer extensible.

**Proposition 10:** Given an IS  $S = \langle T, K, Obj, I \rangle$ , let the *domain* of  $S$  be the set  $\mathcal{D}$  given by  $\mathcal{D} = \{I \cup A \mid A \in \mathcal{P}(T \times Obj)\}$ . Then,  $\cdot^+$  is a continuous function on the complete partial order  $(\mathcal{D}, \subseteq)$ .

*Proof:*  $(\mathcal{D}, \subseteq)$  is obviously a partial order; moreover, it has  $I$  as least element, while the least upper bound of every chain  $K$  in  $\mathcal{D}$  is  $\cup K$ . This proves that  $(\mathcal{D}, \subseteq)$  is a complete partial order. We next show that  $\cdot^+$  is monotonic, that is, that given any two ISs  $S_1 = \langle T, K, Obj, I_1 \rangle$  and  $S_2 = \langle T, K, Obj, I_2 \rangle$  such that  $I_1, I_2 \in \mathcal{D}$ ,  $I_1 \subseteq I_2$  implies  $I_1^+ \subseteq I_2^+$ . Let  $(t, o) \in I_1^+$ . We must prove that  $(t, o) \in I_2^+$ . There are two cases. (1)  $(t, o) \in I_1$ . By the hypothesis,  $(t, o) \in I_2$  and by definition of  $I_2^+$ ,  $(t, o) \in I_2^+$ . (2)  $(t, o) \notin I_1$ . If  $(t, o) \in I_2$  then again by definition of  $I_2^+$ ,  $(t, o) \in I_2^+$ . So, let us assume  $(t, o) \notin I_2$ . Then,  $t \in abind_{S_1}(o)$  and therefore  $\mu_o(t) \cup K \models ind_{S_1}(o)$ . By hypothesis,  $ind_{S_1}(o) \subseteq ind_{S_2}(o)$ , hence  $\mu_o(t) \cup K \models ind_{S_2}(o)$ , and therefore  $\mu_o(t)$  is a solution of the abduction problem  $\mathcal{A}_{S_2}(o)$ . Now, from  $ind_{S_1}(o) \subseteq ind_{S_2}(o)$ , it follows that  $N_{S_1}(o) \subseteq N_{S_2}(o)$ , and therefore the perturbation of any STS  $\mu_o(a)$  in  $S_1$ ,  $\sigma(a) \setminus N_{S_1}(o)$ , is going to be larger than that in  $S_2$ , given by  $\sigma(a) \setminus N_{S_1}(o)$ . This implies that  $\mu_o(t)$  has minimal perturbation in  $S_2$  too. So,  $t \in abind_{S_2}(o)$ . If  $t$  has been selected by the function  $\rho$  to be in  $I_1^+$ , it will also be selected by the same function to be in  $I_2^+$ , hence  $(t, o) \in I_2^+$  and the monotonicity of  $\cdot^+$  is proved. Its continuity follows from monotonicity and from the fact that in the considered complete partial order all chains are finite, hence the class of monotonic functions coincides with the class of continuous functions (see, e.g. [13]).  $\square$

As a corollary of the previous Proposition and of the Knaster-Tarski fixed point theorem, we have that the function  $\cdot^+$  has a least fixed point that is the least upper bound of the chain  $\{I, I^+, (I^+)^+, \dots\}$ . As already remarked, all chains are finite, so the fixed point will be reached after a finite number of iterations. According to Proposition 4, for each object  $o$ , the terms added at each iteration are amongst those in  $T \setminus N_S(o)$ . It follows that the fixed point is reached when, for each object  $o$ ,  $N_S(o) = T$ . This guarantees that the extension process always covers the entire terminology, if pushed to its extreme consequences. Moreover, the same Proposition allows to derive an upper bound on the number of iterations required to reach the fixed point. In fact, assuming that at each iteration  $abind_S(o)$  consists only of one term, in at most

$$\max\{|T \setminus N_S(o)| \mid o \in Obj\}$$

iterations the fixed point is reached.

**Example 9:** Let us consider again the IS  $S$  shown in Figure 2, and let us suppose that the user has requested a browsing starting from object 1. We have:

$$\begin{aligned} O^0 &= \{1\} \\ q^0 &= x \vee w \wedge \neg a \wedge \neg b \wedge \neg c \wedge \neg d \wedge \neg e \wedge \neg f \wedge \neg g \\ O^1 &= \{1, 2\} \end{aligned}$$

Suppose the user is not satisfied and requests another *browse* step. The set  $abind_S(1)$  must be computed. From the table in Figure 6, we have  $abind_S(1) = \{c, g\}$ . As a consequence:

$$\begin{aligned} q^1 &= c \vee g \wedge \neg a \wedge \neg b \wedge \neg d \wedge \neg e \wedge \neg f \\ O^2 &= \{3\} \end{aligned}$$

The user not being yet satisfied, the set  $abind_{S^1}(1)$  is computed. From the table in Figure 7 we have  $abind_{S^1}(1) = \{a, b, e, f\}$ . Consequently:

$$\begin{aligned} q^2 &= a \vee b \vee e \vee f \wedge \neg d \\ O^3 &= \{4, 6\} \end{aligned}$$

If the user asks for another browsing step, the set  $abind_{S^2}(1)$  must be computed. The graph  $G_{S^2}^1$  consists just of the vertex  $d$ , and the only minimal terminal set is  $\{d\}$ , which is also  $abind_{S^2}(1)$ . Therefore:

$$\begin{aligned} q^3 &= d \\ O^4 &= \{5\} \end{aligned}$$

The index of object 1 in  $S^3$  is given by:

$$ind_{S^3}(1) = \{a, c, d\}.$$

Since  $N_{S^3}(1) = T$  no further extension is possible and the browsing process is over having displayed all IS objects to the user.  $\square$

## 5.2. Query Result Ranking

As mentioned in the Introduction, an important feature of information retrieval systems is their ability to provide the user with a ranking of the IS objects as a result of a query evaluation. Ranking is a useful structuring device, and can be obtained also in the context of our qualitative approach, by exploiting the abduction framework in a dual way to the browsing mechanism described above.

The set of objects that the user will get in response to a query  $q$  by applying the query-less information access mechanism outlined in Section 3.4, is given by:

$$answer_S(q) = \bigcup_{i=0}^N ans(q, S^i)$$

where  $N$  is the iteration at which the fixed point is reached, *i.e.*  $S^{N-1} \subset S^N = S^{N+1} = S^{N+2} = \dots$ . Notice that if  $q$  is non-monotonic, *i.e.* it contains negation, certain objects may not appear in  $answer_S(q)$ .

We can order these objects with respect to the iteration at which each object would start appearing in the answer. In particular, we can define the *rank* of an object  $o \in \text{answer}_S(q)$ , denoted by  $\text{rank}_S(o, q)$ , as follows:

$$\text{rank}_S(o, q) = \min\{k \mid o \in \text{ans}(q, S^k)\}$$

The *ranked answer* to  $q$  on  $S$  is an ordering of sets, defined as:

$$\text{rans}(q, S) = \langle \{o \mid \text{rank}_S(o, q) = 1\}, \{o \mid \text{rank}_S(o, q) = 2\}, \dots, \{o \mid \text{rank}_S(o, q) = N\} \rangle.$$

**Example 10:** Consider the HIS presented in Figure 8 and the query  $q = \text{UnderwaterMovingCams}$ . In this case we have:

$$\begin{aligned} \text{ans}(\text{UnderwaterMovingCams}, H^0) &= \emptyset \\ \text{ans}(\text{UnderwaterMovingCams}, H^1) &= \{3\} \\ \text{ans}(\text{UnderwaterMovingCams}, H^2) &= \{1, 2, 3\} \end{aligned}$$

so the ranked answer to  $\text{UnderwaterMovingCams}$  on  $H$  is

$$\text{rans}(\text{UnderwaterMovingCams}, H) = \langle \{3\}, \{1, 2\} \rangle.$$

□

## 6. Conclusions

Indexing accuracy and consistency are difficult to maintain. It has been known for a long time that there are large variations between the index terms assigned by different indexers, or by the same indexer at different times [43]. Imposing a standard indexing language (like the terminology of a taxonomy) tends to improve consistency, but it does not improve accuracy. The same holds for the automatic classification methods which may at best produce approximations. To alleviate the effects of this problem in query-based information access, we have proposed a mechanism which allows liberating the index of a source in a gradual manner. This mechanism is governed by the notion of explanation, logically captured by abduction, and permits to support query-less information access. In this type of access, the user, after posing an initial query to which an unsatisfactory answer has been given by the system, iteratively relaxes the index of the IS, thereby retrieving other objects, which are rougher and rougher approximations of the query.

The proposed method has been shown to have two other important applications:

- similarity-based browsing, in which the user starts from a selected object and navigates to the semantically most similar objects not yet discovered;
- query result ranking according to an explanation-based measure of relevance. Recall that in statistics-based Information Retrieval the notion of relevance is founded only experimentally; what relevance is, in other words, is defined by the user from time to time and from experiment to experiment, and is then heavily dependent on judgments where highly subjective and hardly reproducible factors are brought to bear. Our approach can be used for ordering the objects of the answer according to a more well-founded notion of relevance.

The proposed approach can be applied on collections of manually indexed objects (e.g. Web Catalogs), as well as on automatically indexed collections of texts whose index is derived using automatic classification methods. The only prerequisite is that the employed taxonomy should admit of a semantic interpretation such as the one given in this paper. Specifically the proposed operation can be applied to taxonomies whose relationships can be seen as conditionals. Taxonomies that fall to this category include:

- Taxonomies which conceptualize the domain as a denumerable set of objects, and consist of terms that denote sets of objects and relationships that denote extensional subsumption<sup>1</sup>. The subject hierarchies of Web catalogs fit in this case.
- Taxonomies which consist of terms that denote geographical areas structured by a spatial inclusion relation (e.g. Crete  $\rightarrow$  Greece).
- Taxonomies (for products, living species, fields of knowledge) in which the terms correspond to categories structured by a subclass relation (e.g. Canaries  $\rightarrow$  Birds). For example many existing thesauri fit in this case.

It can also be applied on secondary (i.e. mediator) taxonomy-based sources [41] for alleviating not only the uncertainty that originates from primary sources, but also the uncertainty and inexactness of the mediator mappings.

Also notice that by restricting the set of solutions  $A$  in Definition 5 we can limit the scope of the extension operation to the parts of the index that we want to relax. This restriction can be applied on taxonomies designed according to a faceted approach [34, 33] in order to relax the index with respect to the desired facet(s).

Finally, the proposed operation can be applied not only to manually constructed taxonomies but also to taxonomies derived automatically on the basis of an inference service. For instance, it can be applied on sources indexed using taxonomies of *compound terms* which are defined algebraically [38], or concept hierarchies formed using Description Logics (DL) [10]. Specifically, we can view a DL knowledge base  $\Sigma = (TBox, ABox)$  as an IS  $S = \langle T, K, Obj, I \rangle$  where

- $T$  consists of all concepts that appear in  $TBox$  and  $ABox$ ;
- $K$  contains a conditional  $C \rightarrow D$  ( $C, D \in T$ ) iff  $\Sigma \models C \sqsubseteq D$ , i.e.  $C$  subsumes  $D$  in the knowledge base  $\Sigma$ ;
- $Obj$  is the set of individuals that appear in the concept assertions of  $ABox$ ; and
- $I$  is defined by the concept assertions of the  $ABox$ , namely  $(C, o) \in I$  iff  $C(o) \in ABox$ .

For example consider a query  $Q$  posed to the knowledge base  $\Sigma$ . Let  $S_Q$  be the source obtained by adding the query  $Q$  to the source  $S$  as defined earlier, i.e. by adding  $Q$  to  $T$  and placing  $Q$  in its correct place (with respect to  $\sqsubseteq$ ) in  $K$ . The extension operator  $\cdot^+$  can then be applied to the source  $S_Q$ .

<sup>1</sup>In contrast to the intensional meaning of terms (i.e. see [22], [5]) and to intensional subsumption.

## References

- [1] OWL Web Ontology Language Overview, W3C Recommendation, February 2004, [Http://www.w3.org/TR/owl-features/](http://www.w3.org/TR/owl-features/).
- [2] RDF Primer, W3C Recommendation, February 2004, [Http://w3c.org/TR/rdf-primer/](http://w3c.org/TR/rdf-primer/).
- [3] Baeza-Yates, R., Ribeiro-Neto, B.: “*Modern Information Retrieval*”, ACM Press, Addison-Wesley, 1999.
- [4] Bruza, P., Huibers, T.: Investigating Aboutness Axioms using Information Fields, *Proceedings of SIGIR-94, 17th ACM Conference on Research and Development in Information Retrieval*, Dublin, July 1994.
- [5] Bunge, M.: *Scientific Research I. The Search for Systems*, 3(1), 1967.
- [6] Cialdea Mayer, M., Pirri, F.: Propositional Abduction in Modal Logic, *Bulletin of the Interest Group in Pure and Applied Logic*, 1995.
- [7] Cognitive Science Laboratory, P. U.: “WordNet: A Lexical Database for the English Language”, (<http://www.cogsci.princeton.edu/wn>).
- [8] Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M.: A Uniform Tableaux-Based Method for Concept Abduction and Contraction in Description Logics, *Proc. of 17th European Conference on Artificial Intelligence*, 2004.
- [9] Console, L., Sapino, M. L., Dupr, D. T.: The role of abduction in database view updating, *Journal of Intelligent Information Systems*, 4(3), May 1995, 261–280.
- [10] Donini, F., Lenzerini, M., Nardi, D., Schaerf, A.: Reasoning in Description Logics, in: *Principles of Knowledge Representation* (G. Brewka, Ed.), Studies in Logic, Language and Information, CSLI Publications, 1996, 193–238.
- [11] Eiter, T., Gottlob, G.: The complexity of logic-based abduction, *Journal of the ACM*, 42(1), January 1995, 3–42.
- [12] Enderton, H.: *A mathematical introduction to logic*, Academic Press, N. Y., 1972.
- [13] Fejer, P., Simovici, D.: *Mathematical Foundations of Computer Science. Volume 1: Sets, Relations, and Induction*, Springer-Verlag, 1991.
- [14] Fuhr, N., Rölleke, T.: Information Retrieval with Probabilistic Datalog, in: *Logic and Uncertainty in Information Retrieval: Advanced models for the representation and retrieval of information* (F. Crestani, M. Lalmas, C. J. van Rijsbergen, Eds.), Kluwer Academic Publishing, Dordrecht, NL, 1998, Forthcoming.
- [15] Grüninger, M., Lee, J.: Ontology Applications and Design - Introduction, *Communication of the ACM*, 45(2), February 2002.
- [16] Guarino, N.: “Some Ontological Principles for Designing Upper Level Lexical Resources”, *Proceedings of first int. Conf. on Language Resources and Evaluation*, Granada, Spain, May 1998.
- [17] Guarino, N., Masolo, C., Vetere, G.: “OntoSeek: Content-based Access to the Web”, *IEEE Intelligent Systems*, May, June 1999, 70–80.
- [18] Gulla, J. A., Stein, A.: Mixed-Initiative Retrieval Dialogues Using Abductive Reasoning, *Computational Models for Mixed Initiative Interaction. Papers from the AAAI Spring Symposium, Stanford University.*, number SS-97-04 in AAAI Technical Report, 1997.
- [19] van Harmelen, F., Fensel, D.: “Practical Knowledge Representation for the Web”, *Workshop on Intelligent Information Integration, IJCAI’99*, 1999.

- [20] International Organization For Standardization: “Documentation - Guidelines for the establishment and development of monolingual thesauri”, 1986, Ref. No ISO 2788-1986.
- [21] Kakas, A. C., Mancarella, P.: Database updates through abduction, *Proceedings of VLDB'90, the 16th International Conference on Very Large Databases*, Morgan Kaufmann, 1990.
- [22] Kauppi, R.: “Einführung in die Theorie der Begriffssysteme”, *Acta Universitatis Tampereensis, Ser A, Vol 15, University of Tampere*, 1967.
- [23] Luke, S., Spector, L., Rager, D., Hendler, J.: “Ontology-based Web Agents”, *Proceedings of First International Conference on Autonomous Agents*, 1997, (<http://www.cs.umd.edu/projects/plus/SHOE/>).
- [24] Mazur, Z.: “Models of a Distributed Information Retrieval System Based on Thesauri with Weights”, *Information Processing and Management*, **30**(1), 1994, 61–77.
- [25] McGuinness, D. L.: “Ontological Issues for Knowledge-Enhanced Search”, *Proceedings of FOIS'98*, Amsterdam, IOS Press, Trento, Italy, June 1998.
- [26] Meghini, C., Sebastiani, F., Straccia, U.: A model of multimedia information retrieval, *Journal of the ACM*, **48**(5), 2001, 909–970.
- [27] Meghini, C., Straccia, U.: A relevance terminological logic for information retrieval, *Proceedings of SIGIR-96, the 19th ACM Conference on Research and Development in Information Retrieval*, Zurich, August 1996.
- [28] Meghini, C., Tzitzikas, Y., Spyrtos, N.: An Abduction-based Method for Index Relaxation in Taxonomy-based Sources, *Proceedings of MFCS 2003, 28th International Symposium on Mathematical Foundations of Computer Science*, number 2747 in Lecture notes in computer science, Springer Verlag, Bratislava, Slovak Republic, August 2003.
- [29] Meghini, C., Tzitzikas, Y., Spyrtos, N.: A Unifying Framework for Flexible Information Access in Taxonomy-based Sources, *Proceedings of the 6th International Conference On Flexible Query Answering Systems (FQAS)*, number 3055 in Lecture notes in artificial intelligence, Springer Verlag, Lyon, France, June 2004.
- [30] Müller, A.: A Flexible Framework for Multimedia Retrieval, in: *Information Retrieval: Uncertainty and Logics. Advanced Models for the Representation and Retrieval of Information* (K. van Rijsbergen, F. Crestani, M. Lalmas, Eds.), Kluwer, Dordrecht, NL, 1998.
- [31] Nie, J.: A General Logical Approach to Inferential Information Retrieval, in: *Encyclopedia of Computer Science and Technology* (A. Kent, J. Williams, Eds.), vol. 44, 2001, 203–226.
- [32] Paice, C.: “A Thesaural Model of Information Retrieval”, *Information Processing and Management*, **27**(5), 1991, 433–447.
- [33] Prieto-Diaz, R.: “Implementing Faceted Classification for Software Reuse”, *Communications of the ACM*, **34**(5), 1991.
- [34] Ranganathan, S. R.: “The Colon Classification”, in: *Vol IV of the Rutgers Series on Systems for the Intellectual Organization of Information* (S. Artandi, Ed.), New Brunswick, NJ: Graduate School of Library Science, Rutgers University, 1965.
- [35] Sacco, G. M.: “Dynamic Taxonomies: A Model for Large Information Bases”, *IEEE Transactions on Knowledge and Data Engineering*, **12**(3), May 2000.
- [36] Sowa, J.: Building, Sharing, and Merging Ontologies, Aug 2001, <Http://www.jfsowa.com/ontology/ontoshar.htm>.
- [37] Sullivan, D.: Web Directory Sizes, January 2003, <Http://searchenginewatch.com/reports/article.php/2156411>.

- [38] Tzitzikas, Y., Analyti, A., Spyratos, N., Constantopoulos, P.: "An Algebra for Specifying Compound Terms for Faceted Taxonomies", *13th European-Japanese Conf. on Information Modelling and Knowledge Bases*, Kitakyushu, J, June 2003.
- [39] Tzitzikas, Y., Meghini, C.: "Query Evaluation in Peer-to-Peer Networks of Taxonomy-based Sources", *Proceedings of 19th Int. Conf. on Cooperative Information Systems, CoopIS'2003*, Catania, Sicily, Italy, November 2003.
- [40] Tzitzikas, Y., Meghini, C., Spyratos, N.: "Taxonomy-based Conceptual Modeling for Peer-to-Peer Networks", *Proceedings of 22th Int. Conf. on Conceptual Modeling, ER'2003*, Chicago, Illinois, October 2003.
- [41] Tzitzikas, Y., Spyratos, N., Constantopoulos, P.: "Mediators over Ontology-based Information Sources", *Second Int. Conf. on Web Information Systems Engineering, WISE 2001*, Kyoto, Japan, Dec. 2001.
- [42] Wang, K.: Argumentation-based abduction in disjunctive logic programming, *Journal of Logic Programming*, **45**(1-3), 2000, 105–141.
- [43] Zunde, P., Dexter, M.: "Indexing Consistency and Quality", *American Documentation*, **20**(3), July 1969, 259–267.